Macro Library Resource Study

Peter Camilleri Version 1.0.0 January 18, 2020

This spreadsheet contains the results on a study of the 13 macro files in the CX16 repository.

For each macro, the affects of addressing modes, constants, and data values modify how the macros generate assembly code.

For each condition/path, this is expressed in terms of clock cycles (θ) and bytes (B) used.

adj_16 - zero page

	Α		В		С		D		Ε		F	
.macro _adj_var_16 var,step	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В
.ifconst step												
.if step												
clc			2	1	2	1	2	1	2	1		
.endif												
.if .lobyte(step)												
lda var					3	2	3	2	3	2		
adc #.lobyte(step)					2	2	2	2	2	2		
sta var					3	2	3	2	3	2		
.endif												
.if .hibyte(step)												
lda var+1			3	2					3	2		
adc #.hibyte(step)			2	2					3 2 3	2 2 2		
sta var+1			3	2					3	2		
.elseif .lobyte(step)												
bcc no_carry					3	2	2	2				
inc var+1					0	2	4	2				
.endif												
.else											_	
clc											2	1
lda var											3 2	2
adc #.lobyte(step) sta var											3	2 2
Ida var+1											3	2
adc #.hibyte(step)											2	2
sta var+1											3	2
.endif											J	_
no carry:												
.endmacro												
-	0	0	10	7	13	11	16	11	18	13	18	13

adj_16 - absolute

	Α		В		С		D		Е		F	
.macro _adj_var_16 var,step	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В
.ifconst step												
.if step												
clc			2	1	2	1	2	1	2	1		
.endif												
.if .lobyte(step)												
lda var					4	3	4	3	4	3		
adc #.lobyte(step)					2	2	2	2	2	2		
sta var					4	3	4	3	4	3		
.endif												
.if .hibyte(step)												
lda var+1			4	3					4	3		
adc #.hibyte(step)			2	2					2	2		
sta var+1			4	3					4	3		
.elseif .lobyte(step)												
bcc no_carry					3	2	2	2				
inc var+1					0	3	6	3				
.endif												
.else											_	
clc											2	1
lda var											4	3
adc #.lobyte(step)											2	2
sta var											4	3
lda var+1											4	3 2
adc #.hibyte(step)											2 4	3
sta var+1											4	3
.endif												
no_carry: .endmacro												
.criamació												

adj_16 - zero page,x

	Α		В		С		D		Е		F	
.macro _adj_var_16 var,step	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В
.ifconst step												
.if step												
clc			2	1	2	1	2	1	2	1		
.endif												
.if .lobyte(step)												
lda var,x					4	2	4	2	4	2		
adc #.lobyte(step)					2	2	2	2	2	2		
sta var,x					4	2	4	2	4	2		
.endif												
.if .hibyte(step)												
lda var+1,x			4	2					3	2		
adc #.hibyte(step)			2	2					2	2		
sta var+1,x			4	2					3	2		
.elseif .lobyte(step)												
bcc no_carry					3	2	2	2				
inc var+1,x					0	2	6	2				
.endif												
.else												
clc											2	1
lda var,x											4	2
adc #.lobyte(step)											2	2
sta var,x											4	2
lda var+1,x											4	2
adc #.hibyte(step)											2	2
sta var+1,x											4	2
.endif												
no_carry:												
.endmacro												
	0	0	12	7	15	11	20	11	20	13	22	13

adj_16 - absolute,x (or y) - no page cross

	Α		В		С		D		Ε		F	
.macro _adj_var_16 var,step	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В
.ifconst step												
.if step												
clc			2	1	2	1	2	1	2	1		
.endif												
.if .lobyte(step)												
lda var,x					4	3	4	3	4	3		
adc #.lobyte(step)					2	2	2	2	2	2		
sta var,x					4	3	4	3	4	3		
.endif												
.if .hibyte(step)												
lda var+1,x			4	3					4	3		
adc #.hibyte(step)			2	2					2	3 2 3		
sta var+1,x			4	3					4	3		
.elseif .lobyte(step)												
bcc no_carry					3	2	2	2				
inc var+1,x					0	3	6	3				
.endif												
.else											_	
clc											2	1
lda var,x											4	3
adc #.lobyte(step)											2 4	2
sta var,x												3
Ida var+1,x											4 2	3 2
adc #.hibyte(step) sta var+1,x											4	3
.endif											7	5
no carry:												
.endmacro												
	0	0	12	9	15	14	20	14	22	17	22	17
				-			-					

A step is const 0
B step is const \$xx00
C step is const \$00xx, no carry
D step is const \$00xx, with carry
E step is const

F step is not const

adj_16 - absolute,x (or y) with page cross

, , , , , , , , ,	Α		В		С		D		Е		F	
.macro _adj_var_16 var,step	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В
ifconst step												
.if step												
clc			2	1	2	1	2	1	2	1		
.endif												
.if .lobyte(step)												
lda var,x					5	3	5	3	5	3		
adc #.lobyte(step)					2	2	2	2	2	2		
sta var,x					5	3	5	3	5	3		
.endif												
.if .hibyte(step)												
lda var+1,x			5	3					5	3		
adc #.hibyte(step)			2	2					5 2	2		
sta var+1,x			5	3					5	3		
.elseif .lobyte(step)												
bcc no_carry					3	2	2 7	2				
inc var+1,x					0	3	7	3				
.endif												
.else												
clc											2	1
lda var,x											5	3
adc #.lobyte(step)											2	2
sta var,x											5	3
lda var+1,x											5	3
adc #.hibyte(step)											2	2
sta var+1,x											5	3
.endif												
no_carry:												
.endmacro												
	0	0	14	9	17	14	23	14	26	17	26	17

adj_16 - zero page indirect

<i>'</i> –	. 0	Α		В		С		D	
.macro a	adj_var_16 var,step	θ	В	θ	В	θ	В	θ	В
	st step								
.if	step								
	clc			2	1	2	1		
.er	ndif								
.if	.lobyte(step)								
	lda (var)					5	2		
	adc #.lobyte(step)					2	2		
	sta (var)					5	2		
.er	ndif								
.if	step								
	ldy #1			2	2	2	2		
	lda (var),y			5	2	5	2		
	adc #.hibyte(step)			2	2	2	2		
	sta (var),y			5	2	5	2		
.er	ndif								
.else									
clo	:							2	1
	ı (var)							5	2
	c #.lobyte(step)							2	2
	ı (var)							5	2
-	[,] #1							2	2
	ı (var),y							5	2
	c #.hibyte(step)							2	2
	ı (va),y							5	2
.endif									
no_carry:									
.endmacr	0								
		0	0	16	9	28	15	28	15

A step is const 0
B step is const \$xx00
C step is const
D step is not const

adj_16 - zero page indirect,y

, ,	Α		В		С		D	
.macro _adj_var_16 var,step	θ	В	θ	В	θ	В	θ	В
ifconst step								
.if step								
clc			2	1	2	1		
.endif								
.if .lobyte(step)								
lda (var),y					5	2		
adc #.lobyte(step)					2	2		
sta (var),y					5	2		
.endif								
.if step								
iny			2	1	2	1		
lda (var),y			5	2	5	2		
adc #.hibyte(step)			2	2	2	2		
sta (var),y			5	2	5	2		
dey			2	1	2	1		
.endif								
.else								
clc							2	1
lda (var)							5	2
adc #.lobyte(step)							2	2
sta (var)							5	2
iny							2	1
lda (var),y							5	2
adc #.hibyte(step)							2	2
sta (va),y							5	2
dey							2	1
.endif								
no_carry:								
.endmacro								
	0	0	18	9	30	15	30	15

A step is const 0
B step is const \$xx00
C step is const

cmp_16 - zero page

	Α		В		С		D		Е		F	
.macro _cmp_var_16 var, value	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В
sec	2	1	2	1	2	1	2	1	2	1	2	1
lda var	3	2	3	2	3	2	3	2	3	2	3	2
.if (!.const(value)) .lobyte(value)												
sbc #.lobyte(value)							2	2	2	2	2	2
.endif												
bne not_equal	2	2	3	2	3	2	2	2	3	2	2	2
lda var+1	3	2	0	2	0	2	3	2	0	2	0	2
sbc #.hibyte(value)	2	2	0	2	0	2	2	2	0	2	0	2
bra all_done	3	2	0	2	0	2	3	2	0	2	0	2
not_equal:												
lda var+1	0	2	3	2	3	2	0	2	3	2	3	2
sbc #.hibyte(value)	0	2	2	2	2	2	0	2	2	2	2	2
bne all_done	0	2	3	2	2	2	0	2	3	2	2	2
inc	0	1	0	1	2	1	0	1	0	1	2	1
all_done:												
.endmacro	45	40	40	40	47	40	47		40		40	00
	15	18	16	18	17	18	17	20	18	20	18	20
cmn 16 - abcoluto												
cmp_16 – absolute	٨		В		C		_		_		_	
-	A	_	В	Б	C	_	D	D	E	_	F	Б
.macro _cmp_var_16 var, value	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В
.macro _cmp_var_16 var, value sec	θ 2	1	θ 2	1	θ 2	1	θ 2	1	θ 2	1	θ 2	1
.macro _cmp_var_16 var, value sec lda var	θ		θ		θ		θ		θ		θ	
.macro _cmp_var_16 var, value sec lda var .if (!.const(value)) .lobyte(value)	θ 2	1	θ 2	1	θ 2	1	θ 2 4	1 3	θ 2 4	1 3	θ 2 4	1
.macro _cmp_var_16 var, value sec Ida var .if (!.const(value)) .lobyte(value) sbc #.lobyte(value)	θ 2	1	θ 2	1	θ 2	1	θ 2	1	θ 2	1	θ 2	1
.macro _cmp_var_16 var, value sec Ida var .if (!.const(value)) .lobyte(value) sbc #.lobyte(value) .endif	θ 2 4	1 3	θ 2 4	1	θ 2 4	1	θ 2 4	1 3 2	θ 2 4	1 3 2	θ 2 4 2	1 3 2
.macro _cmp_var_16 var, value sec lda var .if (!.const(value)) .lobyte(value) sbc #.lobyte(value) .endif bne not_equal	θ 2 4	1 3 2	θ 2 4	1 3 2	θ 2 4	1 3 2	θ 2 4 2	1 3 2 2	θ 2 4 2	1 3 2 2	θ 2 4 2	1 3 2 2
.macro _cmp_var_16 var, value sec da var .if (!.const(value)) .lobyte(value) sbc #.lobyte(value) .endif bne not_equal da var+1	θ 2 4 2 4	1 3 2 3	θ 2 4 3 0	1 3 2 3	θ 2 4 3 0	1 3 2 3	θ 2 4 2 2 4	1 3 2 2 3	θ 2 4 2 3 0	1 3 2 2 2 3	θ 2 4 2 2 0	1 3 2 2 3
.macro _cmp_var_16 var, value sec Ida var .if (!.const(value)) .lobyte(value) sbc #.lobyte(value) .endif bne not_equal Ida var+1 sbc #.hibyte(value)	θ 2 4 2 4 2	1 3 2 3 2	θ 2 4 3 0 0	1 3 2 3 2	θ 2 4 3 0 0	1 3 2 3 2	θ 2 4 2 2 4 2	1 3 2 2 3 2	θ 2 4 2 3 0 0	1 3 2 2 2 3 2	θ 2 4 2 2 0 0	1 3 2 2 3 2
.macro _cmp_var_16 var, value sec da var .if (!.const(value)) .lobyte(value) sbc #.lobyte(value) .endif bne not_equal da var+1 sbc #.hibyte(value) bra all_done	θ 2 4 2 4	1 3 2 3	θ 2 4 3 0	1 3 2 3	θ 2 4 3 0	1 3 2 3	θ 2 4 2 2 4	1 3 2 2 3	θ 2 4 2 3 0	1 3 2 2 2 3	θ 2 4 2 2 0	1 3 2 2 3
.macro _cmp_var_16 var, value sec Ida var .if (!.const(value)) .lobyte(value)	θ 2 4 2 4 2 3	1 3 2 3 2 2	θ 2 4 3 0 0	1 3 2 3 2 2	θ 2 4 3 0 0	1 3 2 3 2 2	θ 2 4 2 2 4 2 3	1 3 2 2 3 2 2	θ 2 4 2 3 0 0 0	1 3 2 2 3 2 2	θ 2 4 2 2 0 0	1 3 2 2 3 2 2
.macro _cmp_var_16 var, value sec Ida var .if (!.const(value)) .lobyte(value)	θ 2 4 2 4 2 3	1 3 2 3 2 2	θ 2 4 3 0 0 0	1 3 2 3 2 2	θ 2 4 3 0 0 0	1 3 2 3 2 2	θ2424230	1 3 2 2 3 2 2	θ 2 4 2 3 0 0 0	1 3 2 2 3 2 2	θ 2 4 2 2 0 0 0	1 3 2 2 3 2 2 3
.macro _cmp_var_16 var, value sec Ida var .if (!.const(value)) .lobyte(value)	θ 2 4 2 4 2 3 0 0	1 3 2 3 2 2 3 2	θ 2 4 3 0 0 0 0	1 3 2 3 2 2 3 2	θ 2 4 3 0 0 0	1 3 2 3 2 2 2	θ24242300	1 3 2 2 3 2 2 3 2	θ 2 4 2 3 0 0 0	1 3 2 2 3 2 2 3 2	θ 2 4 2 2 0 0 0 4 2	1 3 2 2 3 2 2 3 2 2
.macro _cmp_var_16 var, value sec Ida var .if (!.const(value)) .lobyte(value)	θ 2 4 2 4 2 3	1 3 2 3 2 2	θ 2 4 3 0 0 0	1 3 2 3 2 2	θ 2 4 3 0 0 0	1 3 2 3 2 2	θ2424230	1 3 2 2 3 2 2	θ 2 4 2 3 0 0 0	1 3 2 2 3 2 2	θ 2 4 2 2 0 0 0	1 3 2 2 3 2 2 3

all_done: .endmacro

17 21 18 21 19 21 19 23 20 23 20 23

A value is const \$xx00, low bytes equal B value is const \$xx00, low bytes not equal, high not zero

C value is const \$xx00, low bytes not equal, high zero

D value is not const or not \$xx00, low bytes equal

E value is not const or not \$xx00, low bytes not equal, high not zero

F value is not const or not \$xx00, low bytes not equal, high zero

cmp_16 - zero page,x

	Α		В		С		D		Е		F	
.macro _cmp_var_16 var, value	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В
sec	2	1	2	1	2	1	2	1	2	1	2	1
lda var,x	4	2	4	2	4	2	4	2	4	2	4	2
.if (!.const(value)) .lobyte(value)	7	_	7	_	7	_	7	_	7	_	7	_
sbc #.lobyte(value)							2	2	2	2	2	2
.endif							_	_	_	_	_	_
bne not_equal	2	2	3	2	3	2	2	2	3	2	2	2
lda var+1,x	4	2	0	2	0	2	4	2	0	2	0	2
sbc #.hibyte(value)	2	2	0	2	0	2	2	2	0	2	0	2
bra all done	3	2	0	2	0	2	3	2	0	2	0	2
not equal:												
 lda var+1,x	0	2	4	2	4	2	0	2	4	2	4	2
sbc #.hibyte(value)	0	2	2	2	2	2	0	2	2	2	2	2
bne all_done	0	2	3	2	2	2	0	2	3	2	2	2
inc	0	1	0	1	2	1	0	1	0	1	2	1
all_done:												
.endmacro												
	17	18	18	18	19	18	19	20	20	20	20	20
cmp_16 - absolute,x (or y)												
	Α		В		С		D		E		F	
<pre>cmp_16 - absolute,x (or y) .macro _cmp_var_16 var, value</pre>	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В
	θ 2	1	θ 2	1	θ 2	1	θ 2	1	θ 2	1	θ 2	1
.macro _cmp_var_16 var, value sec lda var,x	θ		θ	_	θ		θ	_	θ		θ	
.macro _cmp_var_16 var, value sec lda var,x .if (!.const(value)) .lobyte(value)	θ 2	1	θ 2	1	θ 2	1	θ 2 4	1	θ 2 4	1 3	θ 2 4	1 3
.macro _cmp_var_16 var, value sec lda var,x .if (!.const(value)) .lobyte(value) sbc #.lobyte(value)	θ 2	1	θ 2	1	θ 2	1	θ 2	1	θ 2	1	θ 2	1
.macro _cmp_var_16 var, value sec lda var,x .if (!.const(value)) .lobyte(value) sbc #.lobyte(value) .endif	θ 2 4	1 3	θ 2 4	1 3	θ 2 4	1	θ 2 4	1 3 2	θ 2 4	1 3 2	θ 2 4	1 3 2
.macro _cmp_var_16 var, value sec lda var,x .if (!.const(value)) .lobyte(value) sbc #.lobyte(value) .endif bne not_equal	θ 2 4	1 3 2	θ 2 4	1 3 2	θ 2 4	1 3	θ 2 4 2	1 3 2 2	θ 2 4 2 3	1 3 2 2	θ 2 4 2	1 3 2 2
.macro _cmp_var_16 var, value sec Ida var,x .if (!.const(value)) .lobyte(value) sbc #.lobyte(value) .endif bne not_equal Ida var+1,x	θ 2 4 2 4	1 3 2 3	θ 2 4 3 0	1 3 2 3	θ 2 4 3 0	1 3 2 3	θ 2 4 2 2 4	1 3 2 2 3	θ 2 4 2 3 0	1 3 2 2 2 3	θ 2 4 2 2 0	1 3 2 2 2 3
.macro _cmp_var_16 var, value sec da var,x .if (!.const(value)) .lobyte(value) sbc #.lobyte(value) .endif bne not_equal da var+1,x sbc #.hibyte(value)	θ 2 4 2 4 2	1 3 2 3 2	θ 2 4 3 0 0	1 3 2 3 2	θ 2 4 3 0 0	1 3 2 3 2	θ 2 4 2 2 4 2	1 3 2 2 3 2	θ 2 4 2 3 0 0	1 3 2 2 3 2	θ 2 4 2 2 0 0	1 3 2 2 3 2
.macro _cmp_var_16 var, value sec Ida var,x .if (!.const(value)) .lobyte(value) sbc #.lobyte(value) .endif bne not_equal Ida var+1,x sbc #.hibyte(value) bra all_done	θ 2 4 2 4	1 3 2 3	θ 2 4 3 0	1 3 2 3	θ 2 4 3 0	1 3 2 3	θ 2 4 2 2 4	1 3 2 2 3	θ 2 4 2 3 0	1 3 2 2 2 3	θ 2 4 2 2 0	1 3 2 2 2 3
.macro _cmp_var_16 var, value sec lda var,x .if (!.const(value)) .lobyte(value) sbc #.lobyte(value) .endif bne not_equal lda var+1,x sbc #.hibyte(value) bra all_done not_equal:	θ 2 4 2 4 2 3	1 3 2 3 2 2	θ 2 4 3 0 0	1 3 2 3 2 2	θ 2 4 3 0 0	1 3 2 3 2 2	θ 2 4 2 2 4 2 3	1 3 2 2 3 2 2	θ 2 4 2 3 0 0 0	1 3 2 2 3 2 2	θ 2 4 2 2 0 0 0	1 3 2 2 3 2 2
.macro _cmp_var_16 var, value sec lda var,x .if (!.const(value)) .lobyte(value)	θ 2 4 2 4 2 3	1 3 2 3 2 2	θ 2 4 3 0 0 0	1 3 2 3 2 2	θ 2 4 3 0 0 0	1 3 2 3 2 2	θ 2 4 2 2 4 2 3	1 3 2 2 3 2 2	θ 2 4 2 3 0 0 0	1 3 2 2 3 2 2 3	θ 2 4 2 2 0 0 0	1 3 2 2 3 2 2 3
.macro _cmp_var_16 var, value sec lda var,x .if (!.const(value)) .lobyte(value)	θ 2 4 2 4 2 3 0 0	1 3 2 3 2 2 3 2	θ 2 4 3 0 0 0	1 3 2 3 2 2 3 2	θ 2 4 3 0 0 0	1 3 2 3 2 2 3 2	θ 2 4 2 2 4 2 3 0 0	1 3 2 2 3 2 2 3 2	θ 2 4 2 3 0 0 0	1 3 2 2 3 2 2 3 2	θ 2 4 2 2 0 0 0 4 2	1 3 2 2 3 2 2 3 2
.macro _cmp_var_16 var, value sec Ida var,x .if (!.const(value)) .lobyte(value) sbc #.lobyte(value) .endif bne not_equal Ida var+1,x sbc #.hibyte(value) bra all_done not_equal: Ida var+1,x sbc #.hibyte(value) bne all_done	θ 2 4 2 4 2 3 0 0 0	1 3 2 3 2 2 3 2 2	θ24 3 0 0 0 4 2 3	1 3 2 3 2 2 2	θ24 3 0 0 0 4 2 2	1 3 2 3 2 2 2	θ2422423000	1 3 2 2 3 2 2 3 2 2	θ2423000423	1 3 2 2 3 2 2 3 2 2	θ2422000422	1 3 2 2 3 2 2 2 3 2 2
.macro _cmp_var_16 var, value sec da var,x .if (!.const(value)) .lobyte(value)	θ 2 4 2 4 2 3 0 0	1 3 2 3 2 2 3 2	θ 2 4 3 0 0 0	1 3 2 3 2 2 3 2	θ 2 4 3 0 0 0	1 3 2 3 2 2 3 2	θ 2 4 2 2 4 2 3 0 0	1 3 2 2 3 2 2 3 2	θ 2 4 2 3 0 0 0	1 3 2 2 3 2 2 3 2	θ 2 4 2 2 0 0 0 4 2	1 3 2 2 3 2 2 3 2
.macro _cmp_var_16 var, value sec Ida var,x .if (!.const(value)) .lobyte(value) sbc #.lobyte(value) .endif bne not_equal Ida var+1,x sbc #.hibyte(value) bra all_done not_equal: Ida var+1,x sbc #.hibyte(value) bne all_done	θ 2 4 2 4 2 3 0 0 0	1 3 2 3 2 2 3 2 2	θ24 3 0 0 0 4 2 3	1 3 2 3 2 2 2	θ24 3 0 0 0 4 2 2	1 3 2 3 2 2 2	θ2422423000	1 3 2 2 3 2 2 3 2 2	θ2423000423	1 3 2 2 3 2 2 3 2 2	θ2422000422	1 3 2 2 3 2 2 2 3 2 2

Add two clock cycles if a page boundary is crossed.

A value is const \$xx00, low bytes equal

17 21 18 21 19 21 19 23 20 23 20 23

B value is const \$xx00, low bytes not equal, high not zero

C value is const \$xx00, low bytes not equal, high zero

D value is not const or not \$xx00, low bytes equal

E value is not const or not \$xx00, low bytes not equal, high not zero

F value is not const or not \$xx00, low bytes not equal, high zero

cmp_16 - zero page indirect

	Α		В		С		D		Е		F	
.macro _cmp_var_16 var, value	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В
sec	2	1	2	1	2	1	2	1	2	1	2	1
ldy #1	2	2	2	2	2	2	2	2	2	2	2	2
lda (var)	5	2	5	2	5	2	5	2	5	2	5	2
.if (!.const(value)) .lobyte(value)							2	2	2	2	2	2
bne not_equal	2	2	3	2	3	2	2	2	3	2	2	2
lda (var),y	5	2	0	2	0	2	5	2	0	2	0	2
sbc #.hibyte(value)	2	2	0	2	0	2	2	2	0	2	0	2
bra all_done	3	2	0	2	0	2	3	2	0	2	0	2
not equal:												
Ida (var),y	0	2	5	2	5	2	0	2	5	2	5	2
sbc #.hibyte(value)	0	2	2	2	2	2	0	2	2	2	2	2
bne all_done	0	2	3	2	2	2	0	2	3	2	2	2
inc	0	1	0	1	2	1	0	1	0	1	2	1
all_done:												
.endmacro												
	21	20	22	20	23	20	23	22	24	22	24	22

cmp_16 - zero page indirect,y

	Α		В		С		D		Ε		F	
.macro _cmp_var_16 var, value	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В
sec	2	1	2	1	2	1	2	1	2	1	2	1
lda (var)	5	2	5	2	5	2	5	2	5	2	5	2
.if (!.const(value)) .lobyte(value)												
sbc #.lobyte(value)							2	2	2	2	2	2
.endif												
bne not_equal	2	2	3	2	3	2	2	2	3	2	2	2
iny	2	2	0	2	0	2	2	2	0	2	0	2
lda (var),y	5	2	0	2	0	2	5	2	0	2	0	2
dey	2	2	0	2	0	2	2	2	0	2	0	2
sbc #.hibyte(value)	2	2	0	2	0	2	2	2	0	2	0	2
bra all_done	3	2	0	2	0	2	3	2	0	2	0	2
not_equal:												
iny	0	2	2	2	2	2	0	2	2	2	2	2
lda (var),y	0	2	5	2	5	2	0	2	5	2	5	2
dey	0	2	2	2	2	2	0	2	2	2	2	2
sbc #.hibyte(value)	0	2	2	2	2	2	0	2	2	2	2	2
bne all_done	0	2	3	2	2	2	0	2	3	2	2	2
inc	0	1	0	1	2	1	0	1	0	1	2	1
all_done:												
.endmacro												
	23	26	24	26	25	26	25	28	26	28	26	28

A value is const \$xx00, low bytes equal

B value is const \$xx00, low bytes not equal, high not zero

C value is const \$xx00, low bytes not equal, high zero

D value is not const or not \$xx00, low bytes equal

E value is not const or not \$xx00, low bytes not equal, high not zero

F value is not const or not \$xx00, low bytes not equal, high zero

dec_16 - zero page

	Α		В	
.macro _dec_var_16 var	θ	В	θ	В
lda var	3	2	3	2
bne no_wrap	2	2	3	2
dec var+1	0	2	5	2
no_wrap:				
dec var	5	2	5	2
.endmacro				
	10	8	16	8

dec_16 - absolute

.macro _dec_var_16 var	θ	В	θ	В
lda var	3	3	3	3
bne no_wrap	2	2	3	2
dec var+1	0	3	6	3
no_wrap:				
dec var	6	3	6	3
.endmacro				
	11	11	18	11

dec_16 - zero page,x

	A		D	
.macro _dec_vax_16 var	θ	В	θ	В
lda var,x	4	2	4	2
bne no_wrap	2	2	3	2
dec var+1,x	0	2	6	2
no_wrap:				
dec var,x	6	2	6	2
.endmacro				
	12	8	19	R

dec_16 - absolute,x

	Α		В		C		ט	
.macro _dec_vax_16 var	θ	В	θ	В	θ	В	θ	В
lda var,x	4	2	4	2	5	2	5	2
bne no_wrap	3	2	2	2	2	2	3	2
dec var+1,x	0	2	6	2	0	2	7	2
no_wrap:								
dec var,x	6	2	6	2	7	2	7	2
.endmacro								
	13	8	18	8	14	8	22	8

A decrement with no wrap

B decrement with wrap

C decrement with no wrap, crosses page

D decrement with wrap, crosses page

dec_16 - absolute,y

Α		В		C		υ	
θ	В	θ	В	θ	В	θ	В
4	2	4	2	5	2	5	2
3	2	2	2	3	2	2	2
0	2	4	2	0	2	5	2
0	1	0	1	0	1	2	1
0	2	4	2	0	2	5	2
0	2	4	2	0	2	5	2
2	1	2	1	2	1	2	1
4	2	4	2	5	2	5	2
13	14	24	14	15	14	31	14
	θ 4 3 0 0 0 0 2 4	 θ B 4 2 3 2 0 2 0 1 0 2 0 2 2 1 4 2 	 θ β 4 2 4 3 2 2 4 0 1 0 0 2 4 0 2 4 2 4 2 4 2 4 	 θ B θ B 4 2 4 2 3 2 2 2 0 2 4 2 0 1 0 1 0 2 4 2 0 2 4 2 2 2 4 2 2 1 2 1 4 2 4 2 	 θ B θ B θ 4 2 4 2 5 3 2 2 2 3 0 2 4 2 0 0 1 0 1 0 0 2 4 2 0 0 2 4 2 0 0 2 4 2 0 2 4 2 0 2 1 2 1 2 4 2 4 2 5 	 θ B θ B θ B 4 2 4 2 5 2 3 2 2 2 3 2 0 2 4 2 0 2 0 1 0 1 0 1 0 2 4 2 0 2 0 2 4 2 0 2 0 2 4 2 0 2 2 4 2 5 2 	θ B θ B θ B θ 4 2 4 2 5 2 5 3 2 2 2 3 2 2 0 2 4 2 0 2 5 0 1 0 1 0 1 2 5 0 2 4 2 0 2 5 2 1 2 1 2 1 2 4 2 4 2 5 2 5

dec_16 - zero page indirect

	Α		В	
.macro _dec_zpp_16 var	θ	В	θ	В
lda (var)	5	2	5	2
bne no_wrap	3	2	2	2
ldy #\$1	0	2	2	2
lda (var),y	0	2	5	2
dec	0	1	2	1
sta (var),y	0	2	5	2
lda (var)	0	2	5	2
no_wrap:				
dec	2	1	2	1
sta (var)	4	2	5	2
.endmacro				
	14	16	33	16

dec_16 - zero page indirect,y

	Α		В	
.macro _dec_zpp_16 var	θ	В	θ	В
lda (var),y	5	2	5	2
bne no_wrap	3	2	2	2
iny	0	1	2	1
lda (var),y	0	2	5	2
dec	0	1	2	1
sta (var),y	0	2	5	2
dey				
lda (var)	0	2	5	2
no_wrap:				
dec	2	1	2	1
sta (var),y	5	2	5	2
.endmacro				
	15	15	33	15

A decrement with no wrap

B decrement with wrap

C decrement with no wrap, crosses page

D decrement with wrap, crosses page

eql_16 - zero page

	Α		В		С		D		Е		F		G		Н		ı	
.macro _eql_var_16 var, value	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В
.ifconst value																		
.if value																		
lda var			3	2	3	2	3	2	3	2	3	2	3	2				
.if .lobyte(value)																		
cmp #.lobyte(value)							2	2	2	2	2	2	2	2				
.endif																		
bne not_equal			2	2	3	2	2	2	3	2	2	2	3	2				
lda var+1			3	2	0	2	3	2	0	2	3	2	0	2				
.if .hibyte(value)																		
cmp #.hibyte(value)			2	2	0	2					2	2	0	2				
.endif																		
.else																		
lda var	3	2																
ora var+1	3	2																
.endif																		
.else																		
lda var															3	2	2	2
cmp #.lobyte(value)															2	2	2	2
bne not_equal															2	2	3	2
lda var+1															3	2	0	2
cmp #.hibyte(value)															2	2	0	2
not_equal:																		
.endmacro																		
	6	4	10	8	6	8	10	8	8	8	12	10	8	10	12	10	7	10

A value is const 0

B value is const xx00, low bytes equal

C value is const xx00, low bytes equal
D value is const 00xx, low bytes not equal
E value is const 00xx, low bytes equal
E value is const 00xx, low bytes not equal
F value is const, low bytes equal
G value is const, low bytes not equal

H value is not constant, low bytes equal

eql_16 - absolute

	Α		В		С		D		Е		F		G		Н		ı	
.macro _eql_var_16 var, value	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В
.ifconst value																		
.if value																		
lda var			4	3	4	3	4	3	4	3	4	3	4	3				
.if .lobyte(value)																		
cmp #.lobyte(value)							2	2	2	2	2	2	2	2				
.endif																		
bne not_equal			2	2	3	2	2	2	3	2	2	2	3	2				
lda var+1			4	3	0	3	4	3	0	3	4	3	0	3				
.if .hibyte(value)																		
cmp #.hibyte(value)			2	2	0	2					2	2	0	2				
.endif																		
.else																		
lda var	4	3																
ora var+1	4	3																
.endif																		
.else																		
lda var															4	3	4	3
cmp #.lobyte(value)															2	2	2	2
bne not_equal															2	2	3	2
lda var+1															4	3	0	3
cmp #.hibyte(value)															2	2	0	2
not_equal:																		
.endmacro																		
	8	6	12	10	7	10	12	10	9	10	14	12	9	12	14	12	9	12

A value is const 0

B value is const xx00, low bytes equal

C value is const xx00, low bytes equal D value is const 00xx, low bytes equal E value is const 00xx, low bytes not equal

F value is const, low bytes equal

G value is const, low bytes not equal

H value is not constant, low bytes equal

eql_16 - zero page,x

	Α		В		С		D		Е		F		G		Н		ı	
.macro _eql_var_16 var, value	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В
.ifconst value																		
.if value																		
lda var,x			4	2	4	2	4	2	4	2	4	2	4	2				
.if .lobyte(value)																		
cmp #.lobyte(value)							2	2	2	2	2	2	2	2				
.endif																		
bne not_equal			2	2	3	2	2	2	3	2	2	2	3	2				
lda var+1,x			4	2	0	2	4	2	0	2	4	2	0	2				
.if .hibyte(value)																		
cmp #.hibyte(value)			2	2	0	2					2	2	0	2				
.endif																		
.else																		
lda var,x	4	2																
ora var+1,x	4	2																
.endif																		
.else																		
lda var,x															4	2	4	2
cmp #.lobyte(value)															2	2	2	2
bne not_equal															2	2	3	2
lda var+1,x															4	2	0	2
cmp #.hibyte(value)															2	2	0	2
not_equal:																		
.endmacro																		
	8	4	12	8	7	8	12	8	9	8	14	10	9	10	14	10	9	10

A value is const 0

B value is const xx00, low bytes equal

C value is const xx00, low bytes not equal

D value is const 00xx, low bytes equal E value is const 00xx, low bytes not equal F value is const, low bytes equal

G value is const, low bytes not equal

H value is not constant, low bytes equal

eql_16 - absolute,x/y no page cross

	Α		В		С		D		Е		F		G		Н		ı		
.macro _eql_vax_16 var, value	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В	
.ifconst value																			
.if value																			
lda var,x			4	3	4	3	4	3	4	3	4	3	4	3					
.if .lobyte(value)																			
cmp #.lobyte(value)							2	2	2	2	2	2	2	2					
.endif																			
bne not_equal			2	2	3	2	2	2	3	2	2	2	3	2					
lda var+1,x			4	3	0	3	4	3	0	3	4	3	0	3					
.if .hibyte(value)																			
cmp #.hibyte(value)			2	2	0	2					2	2	0	2					
.endif																			
.else																			
lda var,x	4	3																	
ora var+1,x	4	3																	
.endif																			
.else																			
lda var,x															4	3	4	3	
cmp #.lobyte(value)															2	2	2	2	
bne not_equal															2	2	3	2	
lda var+1,x															4	3	0	3	
cmp #.hibyte(value)															2	2	0	2	
not_equal:																			
.endmacro																			
	8	6	12	10	7	10	12	10	9	10	14	12	9	12	14	12	9	12	

A value is const 0

B value is const xx00, low bytes equal

C value is const xx00, low bytes not equal

D value is const 00xx, low bytes equal E value is const 00xx, low bytes not equal

F value is const, low bytes equal

G value is const, low bytes not equal

H value is not constant, low bytes equal

eql_16 - absolute,x/y with page cross

	Α		В		С		D		Ε		F		G		Н			
.macro _eql_vax_16 var, value	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В
.ifconst value																		
.if value																		
lda var,x			5	3	5	3	5	3	5	3	5	3	5	3				
.if .lobyte(value)																		
cmp #.lobyte(value)							2	2	2	2	2	2	2	2				
.endif																		
bne not_equal			2 5	2	3 0	2	2 5	2	3	2	2 5	2	3	2				
lda var+1,x			5	3	0	3	5	3	0	3	5	3	0	3				
.if .hibyte(value)			_	_							_	_	_					
cmp #.hibyte(value)			2	2	0	2					2	2	0	2				
.endif																		
.else																		
lda var,x	5	3																
ora var+1,x	5	3																
.endif																		
.else																		
lda var,x															5	3	5	3
cmp #.lobyte(value)															2	2	2	2
bne not_equal															2	2	3	2
lda var+1,x															5	3	0	3
cmp #.hibyte(value)															2	2	0	2
not_equal:																		
.endmacro																		
	10	6	14	10	8	10	14	10	10	10	16	12	10	12	16	12	10	12

A value is const 0

B value is const xx00, low bytes equal

C value is const xx00, low bytes not equal

D value is const 00xx, low bytes equal E value is const 00xx, low bytes not equal

F value is const, low bytes equal

G value is const, low bytes not equal

H value is not constant, low bytes equal

eql_16 - zero page indirect

	Α		В		С		D		Е		F		G		Н		ı	
.macro _eql_zpp_16 var, value	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В
.ifconst value																		
.if value																		
lda (var)			5	2	5	2	5	2	5	2	5	2	5	2				
.if .lobyte(value)																		
cmp #.lobyte(value)							2	2	2	2	2	2	2	2				
.endif																		
bne not_equal			2	2	3	2	2	2	3	2	2	2	3	2				
ldy #1			2 5	2	0	2	2	2	0	2	2 5	2	0	2				
lda (var),y			5	2	0	2	5	2	0	2	5	2	0	2				
.if .hibyte(value)																		
cmp #.hibyte(value)			2	2	0	2					2	2	0	2				
.endif																		
.else																		
lda (var)	5	2																
ldy #1	2	2																
ora (var),y	5	2																
.endif																		
.else																		
lda (var)															5	2	5	2
cmp #.hibyte(value)															2	2	2	2
bne not_equal															2	2	3	2
ldy #1															2	2	0	2
lda (var),y															5	2	0	2
cmp #.lobyte(value)															2	2	0	2
not_equal:																		
.endmacro																		
	12	6	16	10	8	10	16	10	10	10	18	12	10	12	18	12	10	12

A value is const 0

B value is const xx00, low bytes equal

C value is const xx00, low bytes not equal

D value is const 00xx, low bytes equal

E value is const 00xx, low bytes not equal

F value is const, low bytes equal

G value is const, low bytes not equal

H value is not constant, low bytes equal

eql_16 - zero page indirect,y

	Α		В		С		F		G		Н		ı	
.macro _eql_zpy_16 var, value	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В	θ	В
.ifconst value														
.if value														
lda (var),y			5	2	5	2	5	2	5	2				
.if .lobyte(value)														
cmp #.lobyte(value)							2	2	2	2				
.endif														
bne not_equal			2	2	3	2	2	2	3	2				
iny			2	1	0	1	2	1	0	1				
ldy (var),y			5	2	0	2	5	2	0	2				
dey			2	1	0	1	2	1	0	1				
cmp #.hibyte(value)			2	2	0	2	2	2	0	2				
.else														
iny	2	2												
lda (var).y	5	2												
dey	2	2												
ora (var),y	5	2												
.endif														
.else														
iny											2	1	2	1
lda (var),y											5	2	5	2
dey											2	1	2	1
cmp #.hibyte(value)											2	2	2	2
bne not_equal											2	2	3	2
lda (var),y											5	2	0	2
cmp #.lobyte(value)											2	2	0	2
not_equal:														
.endmacro														
	14	8	18	10	8	10	20	12	10	12	20	12	14	12

A value is const 0

B value is const xx00, low bytes equal

C value is const xx00, low bytes not equal F value is const, low bytes equal

G value is const, low bytes not equal

H value is not constant, low bytes equal

gte_16 - zero page

	Α		В	
.macro _gte_var_16 var, value	θ	В	θ	В
sec	2	1	2	1
.if (!.const(value)) .lobyte(value)				
lda var			3	2
sbc #.lobyte(value)			2	2
.endif				
lda var+1	3	2	3 2	2
sbc #.hibyte(value)	2	2	2	2
endmacro				
	7	5	12	9

gte_16 - absolute

	Α		В	
.macro _gte_var_16 var, value	θ	В	θ	В
sec	2	1	2	1
.if (!.const(value)) .lobyte(value)				
lda var			4	3
sbc #.lobyte(value)			2	2
.endif				
lda var+1	4	2	4 2	3
sbc #.hibyte(value)	2	2	2	2
.endmacro				
	R	5	14	11

gte_16 - zero page,x

	Α		В	
.macro _gte_vax_16 var, value	θ	В	θ	В
sec	2	1	2	1
.if (!.const(value)) .lobyte(value)				
lda var,x			4	2
sbc #.lobyte(value)			2	2
.endif				
lda var+1,x	4	2	4	2
sbc #.hibyte(value)	2	2	2	2
.endmacro				
	8	5	14	9

gte_16 - absolute,x/y

	A		D		C		ע	
.macro _gte_vax_16 var, value	θ	В	θ	В	θ	В	θ	В
sec	2	1	2	1	2	1	2	1
.if (!.const(value)) .lobyte(value)								
lda var,x			4	3			5	3
sbc #.lobyte(value)			2	2			2	2
.endif								
lda var+1,x	4	3	4	3	5	3	5	3
sbc #.hibyte(value)	2	2	2	2	2	2	2	2
.endmacro								
	8	6	14	11	9	6	16	11

A const value xx00

B not const value or not xx00

C const value xx00, page cross

D not const value or not xx00, page cross

gte_16 - zero page indirect

	Α		В	
.macro _gte_zpp_16 var, value	θ	В	θ	В
sec	2	1	2	1
.if (!.const(value)) .lobyte(value)				
lda (var)			5	2
sbc #.lobyte(value)			2	2
.endif				
ldy #1	2	2	2	2
lda (var),y	5	2	5	2
sbc #.hibyte(value)	2	2	2	2
.endmacro				
	11	7	18	11

gte_16 - zero page indirect,y

gie_io - zero page maneci,y				
	Α		В	
.macro _gte_zpy_16 var, value	θ	В	θ	В
sec	2	1	2	1
.if (!.const(value)) .lobyte(value)				
lda (var),y			5	2
sbc #.lobyte(value)			2	2
.endif				
iny	2	1	2	1
lda (var),y	5	2	5	2
dey	2	1	2	1
sbc #.hibyte(value)	2	2	2	2
.endmacro				
	13	7	20	11

A const value xx00 B not const value or not xx00

inc_16 - zero page

.macro _inc_var_16 var
inc var
bne no_wrap
inc var+1
no_wrap:
.endmacro

Α θ 5 3 0	B 2 2 2	Β θ 5 2 5	B 2 2 2
8	6	12	6

inc_16 - absolute

.macro _inc_var_16 var
inc var
bne no_wrap
inc var+1
no_wrap:

Α		В	
θ	В	θ	В
6	3	6	3
3	2	2	2
0	3	6	3
9	8	14	8

inc_16 - zero page,x

.macro inc vax 16 var
inc var,x
bne no_wrap
inc var+1,x
no_wrap:
.endmacro

Α		В	
θ	В	θ	В
6	2	6	2
3	2	2	2
0	2	6	2
9	6	14	6

inc_16 - absolute,x

.macro _inc_vax_16 var
inc var,x
bne no_wrap
inc var+1,x
no_wrap:
.endmacro

Α		В		C		ט	
θ	В	θ	В	θ	В	θ	В
6	3	6			3	7	3
3	2	2	2	3	2	2	2
0	3	6		0	3	7	3
9	8	14	8	10	8	16	8
	θ 6 3 0	θ B 6 3 3 2 0 3	θ B θ 6 3 6 3 2 2 0 3 6	θBθB63632222363	θBθBθ36373222230	 θ B Θ Β Β Θ Θ	θ B θ B θ B θ 6 3 6 3 7 3 7 3 2 2 2 3 2 2 0 3 6 3 0 3 7

A decrement with no wrap

B decrement with wrap

C decrement with no wrap, crosses page

D decrement with wrap, crosses page

inc_16 - absolute,y

Α		В		C		ט	
θ	В	θ	В	θ	В	θ	В
4	3	4	3	5	3	5	3
2	1	2	1	2	1	2	1
4	3	4	3	5	3	5	3
3	2	2	2	3	2	2	2
0	3	4	3	5	3	5	3
0	1	2	1	0	1	2	1
0	3	4	3	5	3	5	3
9	13	18	13	20	13	21	13
	θ 4 2 4 3 0 0	θ B4 32 14 33 20 30 10 3	 θ β 4 3 4 2 4 3 4 3 2 2 0 3 4 0 1 2 0 3 4 	 θ B θ B 4 3 4 3 2 1 2 1 4 3 4 3 3 2 2 2 0 3 4 3 0 1 2 1 0 3 4 3 	 θ B θ B θ 4 3 4 3 5 2 1 2 1 2 4 3 4 3 5 3 2 2 2 3 0 3 4 3 5 0 1 2 1 0 0 3 4 3 5 	 θ B θ B θ B 4 3 4 3 5 3 2 1 2 1 2 1 4 3 4 3 5 3 3 2 2 2 3 2 0 3 4 3 5 3 0 1 2 1 0 1 0 3 4 3 5 3 	θ B θ B θ B θ 4 3 4 3 5 3 5 2 1 2 1 2 1 2 4 3 4 3 5 3 5 3 5 3 2 2 2 3 2 2 2 0 3 4 3 5 3 5 0 1 2 1 0 1 2 0 3 4 3 5 3 5

inc_16 - zero page indirect

	Α		В	
.macro _inc_zpp_16 var	θ	В	θ	В
lda (var)	5	2	5	2
inc	2	1	2	1
sta (var)	5	2	5	2
bne no_wrap	3	2	2	2
ldy #1	0	2	2	2
lda (var),y	0	3	4	3
inc	0	1	2	1
sta (var),y	0	3	4	3
no_wrap:				
.endmacro				
	10	14	21	14

inc_16 - zero page indirect,y

	Α		В	
.macro _inc_vax_16 var	θ	В	θ	В
lda (var),y	5	2	5	2
inc	2	1	2	1
sta (var),y	5	2	5	2
bne no_wrap	3	2	2	2
iny	0	1	2	1
lda (var),y	0	2	5	2
inc	0	1	2	1
sta (var),y	0	2	5	2
dey	0	1	2	1
no_wrap:				
.endmacro				
	10	12	25	12

A decrement with no wrap

- B decrement with wrap
- C decrement with no wrap, crosses page
- D decrement with wrap, crosses page

Ibra

	Α	
.macro Ibra target	θ	В
jmp target	3	3
.endmacro		
	3	3

Ibcc, also Ibcs, Ibne, Ibeq, Ibpl, Ibmi, Ibvc, and Ibvs

	A		D	
.macro lbcc target	θ	В	θ	В
bcs not_taken	2	2	3	2
jmp target	3	3	0	3
not_taken:				
.endmacro				
	5	5	3	5

A branch taken B branch not taken

ls_branches

Ibslt

.macro lbslt target bvs sign_flipped bpl not_taken
taken:
jmp target
sign_flipped:
bpl taken
not_taken:
.endmacro

Α θ 2 2	B 2 2	Β θ 2 3	B 2 2	C θ 3 0	B 2 2	D θ 3 0	B 2 2
3	3	0	3	3	3	0	3
0	2	0	2	3	2	2	2
7	9	5	9	9	9	5	9

A branch taken
B branch not taken
C branch taken, sign flipped
D branch not taken, signed flipped

Ibsle

.macro lbsle target beg taken
bvs sign_flipped
bpl not_taken
taken:
jmp target
sign_flipped:
bpl taken
not_taken:
.endmacro

Α		В		С		D		Ε	
θ	В	θ	В	θ	В	θ	В	θ	В
2	2	2	2	2	2	2	2	3	2
2	2	2	2	3	2	3	2	0	2
2	2	3	2	0	2	0	2	0	2
3	3	0	3	3	3	0	3	3	3
0	2	0	2	3	2	2	2	0	2
9	11	7	11	11	11	7	11	6	11

A branch taken
B branch not taken
C branch taken, sign flipped
D branch not taken, signed flipped
E branch taken, equal

Ibsge

.macro lbsge target bvs sign_flipped bmi not_taken
taken:
jmp target
sign_flipped:
bmi taken
not_taken:
.endmacro

Α		В		С		D	
θ	В	θ	В	θ	В	θ	В
2	2	2	2	3	2	3	2
2	2	3	2	0	2	0	2
3	3	0	3	3	3	0	3
0	2	0	2	3	2	2	2
7	9	5	9	9	9	5	9

A branch taken
B branch not taken
C branch taken, sign flipped
D branch not taken, signed flipped

ls_branches

lbsgt

.macro lbslt target beq not_taken bvs sign_flipped
bmi not_taken
taken:
jmp target
sign_flipped:
bmi taken
not_taken:
.endmacro

Α		В		С		D		Ε	
θ	В	θ	В	θ	В	θ	В	θ	В
2	2	2	2	2	2	2	2	3	2
2	2	2	2	3	2	3	2	0	2
2	2	3	2	0	2	0	2	0	2
3	3	0	3	3	3	0	3	0	3
0	2	0	2	3	2	2	2	0	2
9	11	7	11	11	11	7	11	3	11

A branch taken
B branch not taken
C branch taken, sign flipped
D branch not taken, signed flipped
E branch not taken, equal

lu_branches

В θ В

3 2

0 3

lbult

	Α		В
.macro Ibult target	θ	В	θ
bcs not taken	2	2	3
jmp target	3	3	0
not_taken:			
.endmacro			
	5	5	3

A branch taken B branch not taken

Ibule

	Α		В		С	
.macro Ibule target	θ	В	θ	В	θ	В
beq taken	2	2	2	2	3	2
bcs not taken	2	2	3	2	0	2
taken:						
jmp target	3	3	0	3	3	3
not_taken:						
.endmacro						
	7	7	5	7	6	7

A branch taken B branch not taken C branch taken, equal

Ibuge

.macro lbuge target	θ	В	θ	В
bcc not_taken	2	2	3	2
jmp target	3	3	0	3
not_taken:				
endmacro				
	5	5	3	5

A branch taken B branch not taken

lbugt

	Α		В		С	
.macro Ibult target	θ	В	θ	В	θ	В
bcc not taken	2	2	2	2	3	2
beq not taken	2	2	3	2	0	2
jmp target	3	3	0	3	0	3
not taken:						
.endmacro						
	7	7	5	7	3	7

A branch taken B branch not taken C branch not taken, equal

s_branches

bslt

.macro bslt target
bvs sign_flipped
bmi target
bra not_taken
sign_flipped:
bpl target
not_taken:
.endmacro

Α θ 2 3 0	B 2 2	Β θ 2 2 3	B 2 2	C θ 3 0 0	B 2 2	D θ 3 0	B 2 2 2
0	2	0 7	2		2	2	2

A branch taken
B branch not taken
C branch taken, sign flipped
D branch not taken, signed flipped

bsle

.macro bsle target
beq target
bvs sign_flipped
bmi target
bra not_taken
sign_flipped:
bpl target
not_taken:
.endmacro

Α		В		С		D		Ε	
θ	В	θ	В	θ	В	θ	В	θ	В
2	2	2	2	2	2	2	2	3	2
2	2	2	2	3				0	2
3	2	2	2	0	2	0	2	0	2
0	2	3	2	0	2	0	2	0	2
0	2	0	2	3	2	2	2	0	2
7	10	9	10	8	10	7	10	3	10

A branch taken
B branch not taken
C branch taken, sign flipped
D branch not taken, signed flipped
E branch taken, equal

bsge

.macro bsge target
bvs sign_flipped
bpl target
bra not_taken
sign_flipped:
bmi target
not_taken:
.endmacro

Α		В		С		D	
θ	В	θ	В	θ	В	θ	В
2	2	2	2	3	2	3	2
3	2	2	2	0	2	0	2
0	2	3	2	0	2	0	2
0	2	0	2	3	2	2	2
5	8	7	8	6	8	5	8

A branch taken
B branch not taken
C branch taken, sign flipped
D branch not taken, signed flipped

s_branches

bsgt

.macro bslt target beq not_taken
bvs sign_flipped
bpl target
bra not_taken
sign_flipped:
bmi target
not_taken:
.endmacro

Α		В		С		D		Ε	
θ	В	θ	В	θ	В	θ	В	θ	В
2	2	2	2	2	2	2	2	3	2
2	2	2	2	3	2	3	2	0	2
3	2	2	2	0	2	0	2	0	2
0	2	3	2	0	2	0	2	0	2
0	2	0	2	3	2	2	2	0	2
7	10	9	10	8	10	7	10	3	10

A branch taken
B branch not taken
C branch taken, sign flipped
D branch not taken, signed flipped
E branch not taken, equal

set_16 - zero page

	Α		В		С		D		Ε	
.macro _set_var_16 var,value	θ	В	θ	В	θ	В	θ	В	θ	В
.ifconst value										
.if .lobyte(value)										
lda #.lobyte(value)			2	2			2	2		
sta var			3	2			3	2		
.else										
stz var	3	2			3	2				
.endif										
.if .hibyte(value)										
lda #.hibyte(value)					2	2	2	2		
sta var+1					3	2	3	2		
.else										
stz var+1	3	2	3	2						
.endif										
.else										
lda #.lobyte(value)									2	2
sta var									3 2	2
lda #.hibyte(value)									2	2 2 2
sta var+1									3	2
.endif										
.endmacro										
	6	4	8	6	8	6	10	8	10	8

set_16 - absolute

	Α		В		С		D		Е	
.macro _set_var_16 var,value	θ	В	θ	В	θ	В	θ	В	θ	В
.ifconst value										
.if .lobyte(value)										
lda #.lobyte(value)			2	2			2	2		
sta var			4	3			4	3		
.else										
stz var	4	3			4	3				
.endif										
.if .hibyte(value)										
lda #.hibyte(value)					2	2	2	2		
sta var+1					4	3	4	3		
.else										
stz var+1	4	3	4	3						
.endif										
.else										
lda #.lobyte(value)									2	2
sta var									4	2 3 2
lda #.hibyte(value)									2	2
sta var+1									4	3
.endif										
.endmacro										
	8	6	10	8	10	8	12	10	12	10

A set const value 0000 B set const value 00xx C set const value xx00

D set const value xxxx

E set non-const value

set_16 - zero page,x

	Α		В		С		D		Ε	
.macro _set_vax_16 var,value	θ	В	θ	В	θ	В	θ	В	θ	В
.ifconst value										
.if .lobyte(value)										
lda #.lobyte(value)			2	2			2	2		
sta var,x			4	2			4	2		
.else										
stz var,x	4	2			4	2				
.endif										
.if .hibyte(value)										
lda #.hibyte(value)					2	2	2	2		
sta var+1,x					4	2	4	2		
.else										
stz var+1,x	4	2	4	2						
.endif										
.else										
lda #.lobyte(value)									2	2
sta var,x									4	2
lda #.hibyte(value)									2	2
sta var+1,x									4	2
.endif										
.endmacro										
	8	4	10	6	10	6	12	8	12	8

set_16 – absolute,x – no page cross

set_10 - absolute,x - 110 page c1055										
	Α		В		С		D		Ε	
.macro _set_vax_16 var,value	θ	В	θ	В	θ	В	θ	В	θ	В
.ifconst value										
.if .lobyte(value)										
lda #.lobyte(value)			2	2			2	2		
sta var,x			4	3			4	3		
.else										
stz var,x	4	3			4	3				
.endif										
.if .hibyte(value)										
lda #.hibyte(value)					2	2	2	2		
sta var+1,x					4	3	4	3		
.else										
stz var+1,x	4	3	4	3						
.endif										
.else										
lda #.lobyte(value)									2	2
sta var,x									4	3
lda #.hibyte(value)									2	2
sta var+1,x									4	3
.endif										
.endmacro										
	8	6	10	8	10	8	12	10	12	10

A set const value 0000 B set const value 00xx

C set const value xx00

D set const value xxxx

E set non-const value

set_16 - absolute,x - with page cross

set_io - absolute,x - with page cross											
	Α		В		С		D		Ε		
.macro _set_vax_16 var,value	θ	В	θ	В	θ	В	θ	В	θ	В	
.ifconst value											
.if .lobyte(value)											
lda #.lobyte(value)			2	2			2	2			
sta var,x			5	3			5	3			
.else											
stz var,x	5	3			5	3					
.endif											
.if .hibyte(value)											
lda #.hibyte(value)					2	2	2	2			
sta var+1,x					5	3	5	3			
.else											
stz var+1,x	5	3	5	3							
.endif											
.else											
lda #.lobyte(value)									2	2	
sta var,x									5	3 2	
lda #.hibyte(value)									2		
sta var+1,x									5	3	
.endif											
.endmacro											
	10	6	12	8	12	8	14	10	14	10	

A set const value 0000 B set const value 00xx C set const value xx00

D set const value xxxx

E set non-const value

set_16 - absolute,y

	Α		В	
.macro _set_vay_16 var,value	θ	В	θ	В
lda #.lobyte(value)	2	2	2	2
sta var,y	4	3	5	3
lda #.hibyte(value)	2	2	2	2
sta var+1,y	4	3	5	3
.endmacro				
	12	10	14	10

A no page cross B with page cross

set_16 - zero page indirect

	~	
.macro _set_zpp_16 var,value	θ	В
lda #.lobyte(value)	2	2
sta (var)	4	2
ldy #1	2	2
lda #.hibyte(value)	2	2
sta (var),y	4	2
.endmacro		
	14	10

set_16 - zero page indirect,y

page,,		
	Α	
.macro _set_zpy_16 var,value	θ	В
lda #.lobyte(value)	2	2
sta (var),y	4	2
iny	2	1
lda #.hibyte(value)	2	2
sta (var),y	4	2
dey	2	2
.endmacro		
	16	11

C

tst_16 - zero page

	Α		В		C	
.macro _tst_var_16 var	θ	В	θ	В	θ	В
lda var+1	3	2	3	2	3	2
bne all_done	3	2	2	2	2	2
ora var	0	2	3	2	3	2
bpl all_done	0	2	3	2	2	2
Isr	0	1	0	1	2	1
all_done:						
.endmacro						
	6	9	11	9	12	9

tst_16 - absolute

.macro _tst_var_16 var	
lda var+1	
bne all_done	
ora var	
bpl all_done	
Isr	
all_done:	
.endmacro	

tst_16 - zero page,x

.macro _tst_var_16 var
lda var+1,x
bne all_done
ora var,x
bpl all_done
Isr
all_done:
.endmacro

A high byte != 0 B high byte = 0, low byte < \$80 C high byte = 0, low byte >= \$80

Α		В		С	
θ	В	θ	В	θ	В
4	3	4	3	4	3
3	2	2	2	2	2
0	3	4	3	4	3
0	2	3	2	2	2
0	1	0	1	2	1
_					
7	11	13	11	14	11

Α		В		С	
θ	В	θ	В	θ	В
4	2	4	2	4	2
3	2	2	2	2	2
0	2	4	2	4	2
0	2	3	2	2	2
0	1	0	1	2	1
7	9	13	9	14	9

F

tst_16 - absolute,x/y

A		D		C		ע		_		Г	
θ	В	θ	В	θ	В	θ	В	θ	В	θ	В
4	3	4	3	4	3	5	3	5	3	5	3
3	2	2	2	2	2	3	2	2	2	2	2
0	3	4	3	4	3	0	3	5	3	5	3
0	2	3	2	2	2	0	2	2	2	3	2
0	1	0	1	2	1	0	1	0	1	2	1
7	11	13	11	14	11	8	11	14	11	17	11
	θ 4 3 0 0	θ B4 33 20 30 2	θ B θ4 3 43 2 20 3 40 2 3	θ B θ B4 3 4 33 2 2 20 3 4 30 2 3 2	θ B θ B θ 4 3 4 3 4 3 2 2 2 2 0 3 4 3 4 0 2 3 2 2	θ B θ B θ B 4 3 4 3 4 3 3 2 2 2 2 2 0 3 4 3 4 3 0 2 3 2 2 2	θ B θ B θ B θ 4 3 4 3 4 3 5 3 2 2 2 2 2 3 0 3 4 3 4 3 0 0 2 3 2 2 2 2 0	θ B θ B θ B θ B 4 3 4 3 4 3 5 3 3 2 2 2 2 2 3 2 0 3 4 3 4 3 0 3 0 2 3 2 2 2 0 2	θ B θ B θ B θ B θ 4 3 4 3 4 3 5 3 5 3 2 2 2 2 2 3 2 2 0 3 4 3 4 3 0 3 5 0 2 3 2 2 2 0 2 2	θ B θ B θ B θ B θ B 4 3 4 3 4 3 5 3 5 3 3 2 2 2 2 2 2 2 2 2 0 3 4 3 4 3 0 3 5 3 0 2 3 2 2 2 0 2 2 2	θ B θ B </td

A high byte != 0 B high byte = 0, low byte < \$80 C high byte = 0, low byte >= \$80 D high byte != 0 E high byte = 0, low byte < \$80 F high byte = 0, low byte >= \$80

u_branches

bult

.macro bult target
bcc target
not_taken:
.endmacro

A B Θ B 3 2 2 2 3

A branch taken B branch not taken

bule

.macro bult target
beq target
bcc target
.endmacro

	В		С	
В	θ	В	θ	В
2	2	2	3	2
2	2	2	0	2
4	4	4	3	4
	2	B θ 2 2 2 2	B θ B 2 2 2 2 2 2	B θ B θ 2 2 2 3 2 2 2 0

A branch taken B branch not taken C branch taken, equal

buge

.macro buge target
bcs target
.endmacro

A branch taken B branch not taken

bugt

.macro bult target beq not_taken bcs target .endmacro

Α θ 2 3	2	2	B 2 2	C θ 3 0	2
5	4	4	4	3	4

A branch taken B branch not taken C branch not taken, equal