Instruction; Cycles	C74 Cycles	
$9B TXY ; 2 cycles, Flags: NZ	2	
$BB TYX ; 2 cycles, Flags: NZ	2	
$FC JSR (absx) ; 6 cycles, no Flags	8	
$82 BRL #16 ; 4 cycles, no flags	4	
$F4 PEA #16 ; 5 cycles, no Flags	5	
$D4 PEI z ; 6 cycles, no Flags	6	
$62 PER #16 ; 6 cycles, no Flags	6	
$03 ORA d,s ; 4 cycles, Flags: NZ	5	
$23 AND d,s ; 4 cycles, Flags: NZ	5	
$43 EOR d,s ; 4 cycles, Flags: NZ	5	
$63 ADC d,s ; 4 cycles, Flags: NVZC	5	
$83 STA d,s ; 4 cycles, no Flags	5	
$A3 LDA d,s ; 4 cycles, Flags: NZ	5	
$C3 CMP d,s ; 4 cycles, Flags: NZC	5	
$E3 SBC d,s ; 4 cycles, Flags: NVZC	5	
$13 ORA (d,s),y ; 7 cycles, Flags: NZ	8	
$33 AND (d,s),y ; 7 cycles, Flags: NZ	8	
$53 EOR (d,s),y ; 7 cycles, Flags: NZ	8	
$73 ADC (d,s),y ; 7 cycles, Flags: NVZC	8	
$93 STA (d,s),y ; 7 cycles, no Flags	8	
$B3 LDA (d,s),y ; 7 cycles, Flags: NZ	8	
$D3 CMP (d,s),y ; 7 cycles, Flags: NZC	8	
$F3 SBC (d,s),y ; 7 cycles, Flags: NVZC	8	
$1B TCS ; A -> S ; 2 cycles, no flags	2	8 Bits only
$3B TSC ; S -> A ; 2 cycles, Flags: NZ	2	8 Bits only
$E2 SEP #imm; P := P OR #imm; 3 cycles	2	
$C2 REP #imm; P := P AND /#imm; 3 cycles	2	
BRK; 7 cycles	8	

COP / WDM / MVN / MVP / XCE

Addressing Modes

Interrupt vectors in bank 0

Immeditate

Implied

Absolute	LDA $FFD2	16 bit address in current bank
Absolute Long	LDA $ABFFD2	24 bit address (TBR)
Absolute Indexed X	LDA $FFD2,X	Crosses Banks
Ansolute Indexed Y	LDA $FFD2,Y	Crosses Banks
Absolute Long Indexed X	LDA $ABFFD2,X	Crosses Banks
Direct Page (Zero Page)	LDA $FF	8 bit address in DH
Direct Indexed X	LDA $FF,X	Wraps on page bounbdaries
Direct Indexed Y	LDA $FF,Y	Wraps on page bounbdaries
Direct Page Indirect	LDA ($00)	16 bit pointer in DH
Direct Page Indirect Long	LDA [$00]	24 bit Pointer in DH

Direct Page Indirect Indexed Y	LDA ($00),Y	Crosses Banks
Direct Page Indexed Indirect X	LDA ($00,X)	Wraps on page bounbdaries
Direct Page Long Indexed Y	LDA [$00],Y	24 bit Pointer in DH, no wrap on banks
Stack		Always in Bank 0
Stack Relative	LDA d,S	Always in Bank 0
Stack Relative Indirect Indexed	LDA (d,S),Y	Crosses Banks
PC Increment	PC += 1	Wraps on bank boundaries
Branch PC relative	BRA	256 byte range, Wraps on bank
Branch PC relative Long	BRL	64K range, Wraps on bank boundaries
JMP Absolute Indexed Indirect	JMP ($FFD2,X)	16 bit pointer in current program bank
JSR Absolute Indexed Indirect	JSR ($FFD2,X)	16 bit pointer in current program bank
JMP Absolute Indirect	JMP ($2000)	16 bit pointer in current Bank 0
JMP Absolute Indirect Long	JMP [$2000]	24 bit pointer in Bank 0 -> changes PBR
JMP/JSR Absolute Long	JMP $ABFFD2	24 bit address -> changes PBR

Note: 65C02 fixes JMP (abs) bug but adds a cycle
Note: 65816 fixes JMP (abs) and has the correct # of cycles

K24 Microcode differences to 65816 Emulation Mode:
Not cycle accurate
D register is 8 bits (called "DH"), points only to page boundaries
PHD and PLD still push and pull a 16 bit register (with $00 as the low byte)
A & B accumulators cannot be used as 16 bit register.
TCS, TSC transfer the A accumulator to the S register
TCD, TDC transter the B accumulator to the DH register
All opcodes which manipulate the stack wrap on page boundaries
ABORT and COP Interrupt Vectors not present
XCE opcode replaced by NOP
MVN and MVP opcodes not implemented
MVN Opcode ($44) replaced by CFG instruction:
  Exchange CFG register with A accumulator
  CFG [0..1]: 0 (6502), 1 (65C02), 2 (6502+NOPs), 3 (K24 - 65816 Emulation Mode)
COP Opcode ($02) replaced by SPI instruction:
  Transfers 8 bits between the A accumulator and external SPI devices
  SPI #<SS>, where SS = Bitwise Slave Select SS0, SS1, SS2; 1 = Select
  SS7 = 1 leaves slave selected at the end of instruction

6502/65C02 Microcode differences to 65816 Emulation Mode:
Same as K24 but PBR is set to 0 only on RESET.
This emables 6502 and 65C02 mode to run in 64k code segment "partitions"
Interrupt Vectors are expected on the current Program Bank, NOT Bank 0