6510 Assembly Instructions Cheat Sheet

Logical and Arithmetic

1 byte | 2 bytes | 3 bytes

* add one cycle if page boundary crossed

	(implied)	#00	$00	$00,X	($00,X)	($00),Y	$0000	$0000,X	$0000,Y	Flags	Function
ORA		09 2	05 3	15 4	01 6	11 5*	0D 4	1D 4*	19 4*	NZ	A or $ ⇒ A
AND		29 2	25 3	35 4	21 6	31 5*	2D 4	3D 4*	39 4*	NZ	A and $ ⇒ A
EOR		49 2	45 3	55 4	41 6	51 5*	4D 4	5D 4*	59 4*	NZ	A xor $ ⇒ A
ADC		69 2	65 3	75 4	61 6	71 5*	6D 4	7D 4*	79 4*	NZCV	A + $ + C ⇒ A
SBC		E9 2	E5 3	F5 4	E1 6	F1 5*	ED 4	FD 4*	F9 4*	NZCV	A − $ + C − 1 ⇒ A
CMP		C9 2	C5 3	D5 4	C1 6	D1 5*	CD 4	DD 4*	D9 4*	NZC	A − $
CPX		E0 2	E4 3				EC 4			NZC	X − $
CPY		C0 2	C4 3				CC 4			NZC	Y − $
DEC			C6 5	D6 6			CE 6	DE 7		NZ	$ − 1 ⇒ $
DEX	CA 2									NZ	X − 1 ⇒ X
DEY	88 2									NZ	Y − 1 ⇒ Y
INC			E6 5	F6 6			EE 6	FE 7		NZ	$ + 1 ⇒ $
INX	E8 2									NZ	X + 1 ⇒ X
INY	C8 2									NZ	Y + 1 ⇒ Y
ASL	0A 2		06 5	16 6			0E 6	1E 7		NZC	$×2 ⇒ $
ROL	2A 2		26 5	36 6			2E 6	3E 7		NZC	$×2 + C ⇒ $
LSR	4A 2		46 5	56 6			4E 6	5E 7		NZC	$/2 ⇒ $
ROR	6A 2		66 5	76 6			6E 6	7E 7		NZC	$/2 + C×128 ⇒ $

Loads and Stores

2 bytes | 3 bytes

* add one cycle if page boundary crossed

	#00	$00	$00,X	$00,Y	($00,X)	($00),Y	$0000	$0000,X	$0000,Y	Flags	Function
LDA	A9 2	A5 3	B5 4		A1 6	B1 5*	AD 4	BD 4*	B9 4*	NZ	$ ⇒ A
STA		85 3	95 4		81 6	91 6	8D 4	9D 5	99 5		A ⇒ $
LDX	A2 2	A6 3		B6 4			AE 4		BE 4*	NZ	$ ⇒ X
STX		86 3		96 4			8E 4				X ⇒ $
LDY	A0 2	A4 3	B4 4				AC 4	BC 4*		NZ	$ ⇒ Y
STY		84 3	94 4				8C 4				Y ⇒ $

Jumps, Calls and NOP

1 byte | 3 bytes

† BRK pushes PC+2, padding with NOP recommended

	(implied)	$0000	($0000)	Flags	Function
BRK	00 7†			BI	PC+2 ⇒ (−−S), P⇒(S), $FFFE⇒PC
RTI	40 6			NZCVDI	(S++)⇒P, (S++)⇒PC
JSR		20 6			PC ⇒ (−−S), $ ⇒ PC
RTS	60 6				(S++) ⇒ PC
JMP		4C 3	6C 5		$ ⇒ PC
NOP	EA 2				No operation

Branches

2 bytes

	±00	Branch on
BPL	10 2*	N = 0
BMI	30 2*	N = 1
BVC	50 2*	V = 0
BVS	70 2*	V = 1
BCC	90 2*	C = 0
BCS	B0 2*	C = 1
BNE	D0 2*	Z = 0
BEQ	F0 2*	Z = 1

* add a cycle if branch is taken, another if page is boundary crossed

Transfers and Stack

1 byte

	(implied)	Flags	Function
TAX	AA 2	NZ	A ⇒ X
TXA	8A 2	NZ	X ⇒ A
TAY	A8 2	NZ	A ⇒ Y
TYA	98 2	NZ	Y ⇒ A
TSX	BA 2	NZ	S ⇒ X
TXS	9A 2		X ⇒ S
PLA	68 4	NZ	(S++) ⇒ A
PHA	48 3		A ⇒ (−−S)
PLP	28 4	NZCVDI	(S++) ⇒ P
PHP	08 3		P ⇒ (−−S)

Flag Manipulation

1 byte | 2 bytes | 3 bytes

	(implied)	$00	$0000	Flags set
BIT		24 3	2C 4	$ bit 7 ⇒ N, $ bit 6 ⇒ V, $ and A ⇒ Z
CLC	18 2			0 ⇒ C
SEC	38 2			1 ⇒ C
CLD	D8 2			0 ⇒ D
SED	F8 2			1 ⇒ D
CLI	58 2			0 ⇒ I
SEI	78 2			1 ⇒ I
CLV	B8 2			0 ⇒ V

size → 1 byte; opcode → EA; mnemonic → NOP; cycles → 2

P-register	Flags:
bit 0	C = Carry
bit 1	Z = Zero
bit 2	I = Interrupt disable
bit 3	D = Decimal
bit 4	B = Break (BRK hit)
bit 6	V = Overflow (sign changed)
bit 7	N = Negative (bit 7 set)

Illegal NOPs

Like LDA but discard result

	(implied) 1 byte	also known as DOP 2 bytes #00	$00	$00,X	also known as TOP 3 bytes $0000	$0000,X
	1A 3A 5A 7A DA FA 2	80 82 89 C2 E2 2	04 44 64 3	14 34 54 74 D4 F4 4	0C 4	1C 3C 5C 7C DC FC 4*

* add one cycle if page boundary crossed

Illegal JAMs

Freeze the CPU

02 12 22 32 42 52
62 72 92 B2 D2 F2

Illegal Logical and Arithmetic

2 bytes | 3 bytes

	#00	$00	$00,X	$00,Y	($00,X)	($00),Y	$0000	$0000,X	$0000,Y	Flags	Function
SLO ASO		07 5	17 6		03 8	13 8	0F 6	1F 7	1B 7	NZC	ASL $, ORA $
RLA		27 5	37 6		23 8	33 8	2F 6	3F 7	3B 7	NZC	ROL $, AND $
SRE LSE		47 5	57 6		43 8	53 8	4F 6	5F 7	5B 7	NZC	LSR $, EOR $
RRA		67 5	77 6		63 8	73 8	6F 6	7F 7	7B 7	NZCV	ROR $, ADC $
SAX		87 3		97 4	83 6		8F 4				A and X ⇒ $
LXA AB, LAX	2†	A7 3		B7 4	A3 6	B3 5*	AF 4		BF 4*	NZ	LDA $, LDX $
DCP DCM		C7 5	D7 6		C3 8	D3 8	CF 6	DF 7	DB 7	NZC	DEC $, CMP $
ISB, ISC INS		E7 5	F7 6		E3 8	F3 8	EF 6	FF 7	FB 7	NZCV	INC $, SBC $
ANC 0B,2B	2									NZC	AND $, N ⇒ C
ASR ALR	4B 2									NZC	AND $, LSR
ARR	6B 2									NZCV	AND $, ROR
ANE XAA	8B 2†									NZ	TXA, AND $
SBX AXS	CB 2									NZC	[A and X] − $ ⇒ X
SBC2 SBC	EB 2									NZCV	A − $ + C − 1 ⇒ A
AHX, SHA AXA						93 6‡			9F 5‡		A and X and H ⇒ $
SHY SAY								9C 5‡			Y and H ⇒ $
SHX XAS									9E 5‡		X and H ⇒ $
SHS TAS									9B 5‡		A and X ⇒ S, S and H
LAE, LAR LAS									BB 4*	NZ	S and $ ⇒ [A,X,S]

* add one cycle if page boundary crossed
† highly unstable ‡ partially unstable H = high byte of the address + 1

v0.2.0 (c) 2018-2024 Veikko Sariola, CC BY-SA 4.0
https://github.com/vsariola/c64-cheat-sheets/

Sources:
https://csdb.dk/release/?id=226987
http://www.oxyron.de/html/opcodes02.html
http://unusedino.de/ec64/technical/aay/c64/
http://codebase64.org/doku.php?id=base:6502_6510_coding