

6510 Assembly Instructions Cheat Sheet

Logical and Arithmetic

	1 byte				2 bytes				3 bytes					
	(implied)	#00	\$00	\$00,X	(\$00,X)	(\$00),Y	\$0000	\$0000,X	\$0000,Y		Flags	Function		
ORA	09	05	15	01	11	0D	1D	19			NZ	A or \$ ⇒ A		
AND	29	25	35	21	31	2D	3D	39			NZ	A and \$ ⇒ A		
EOR	49	45	55	41	51	4D	5D	59			NZ	A xor \$ ⇒ A		
ADC	69	65	75	61	71	6D	7D	79			NZCV	A + \$ + C ⇒ A		
SBC	E9,EB†	E5	F5	E1	F1	ED	FD	F9			NZCV	A - \$ + C - 1 ⇒ A		
CMP	C9	C5	D5	C1	D1	CD	DD	D9			NZC	A - \$		
CPX	E0	E4				EC					NZC	X - \$		
CPY	CO	C4				CC					NZC	Y - \$		
DEC		C6	D6			CE	DE				NZ	\$ - 1 ⇒ \$		
DEX	CA										NZ	X - 1 ⇒ X		
DEY	8B										NZ	Y - 1 ⇒ Y		
INC		E6	F6			EE	FE				NZ	\$ + 1 ⇒ \$		
INX	EB										NZ	X + 1 ⇒ X		
INY	CB										NZ	Y + 1 ⇒ Y		
ASL	0A		06	16		0E	1E				NZC	\$ × 2 ⇒ \$		
ROL	2A		26	36		2E	3E				NZC	\$ × 2 + C ⇒ \$		
LSR	4A		46	56		4E	5E				NZC	\$/2 ⇒ \$		
ROR	6A		66	76		6E	7E				NZC	\$/2 + C × 128 ⇒ \$		

* add one cycle if page boundary crossed
† EB is an unintended opcode

Loads and Stores

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

* add one cycle if page boundary crossed

Jumps and Calls

	1 byte	3 bytes			
	(implied)	\$0000	(\$0000)	Flags	Function
BRK	7†				PC+2 ⇒ (--S), P ⇒ (--S), \$FFF ⇒ PC
RTI	40				NZCVDI (S++) ⇒ P, (S++) ⇒ PC
JSR		20			PC ⇒ (--S), \$ ⇒ PC
RTS		60			(S++) ⇒ PC
JMP		4C	6C		\$ ⇒ PC

† BRK pushes PC+2, padding with NOP recommended

Branches

	2 bytes		
	±00	Branch on	
BPL	2*	N = 0	
BMI	2*	N = 1	
BVC	2*	V = 0	
BVS	2*	V = 1	
BCC	2*	C = 0	
BCS	2*	C = 1	
BNE	2*	Z = 0	
BEQ	2*	Z = 1	

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

Transfers and Stack

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

Flag Manipulation

	1 byte	2 bytes	3 bytes	
(implied)	\$00	\$0000	\$0000	Flags set
BIT	24	2C	4	\$ bit 7 ⇒ N, \$ bit 6 ⇒ V, \$ and A ⇒ Z
CLC	18			0 ⇒ C
SEC	38			1 ⇒ C
CLD	D8			0 ⇒ D
SED	F8			1 ⇒ D
CLI	58			0 ⇒ I
SEI	78			1 ⇒ I
CLV	88			0 ⇒ V

Illegal Logical and Arithmetic Instructions

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

† highly unstable ‡ partially unstable H = high byte of the address + 1

instruction → **NOP** → size → 1 byte → opcode → EA → 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

	1 byte	2 bytes	3 bytes		
(implied)	#00	\$00	\$00,X	\$0000	\$0000,X
	1A	80	14	1C	
	3A	82	34	3C	
	5A	84	54	5C	
	7A	86	74	7C	
	DA	C2	44	D4	0C
	FA	E2	64	F4	FC
	2	2	3	4	4

* add one cycle if page boundary crossed

Illegal JAMs

Freeze the CPU

02	12	22	32	42	52
0E	1E	2E	3E	4E	5E