MICRO LOGIC CORP.
HACKENSACK, NJ

6502 (65XX)

MICROPROCESSOR INSTANT REFERENCE CARD

MICRO® CHART

INSTRUCTION SET

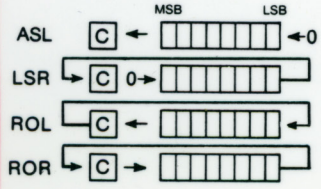
	INSTRUCTION	OP	C	B	DESCRIPTION	ADDRESSING
A	ADC #n	69	2	2	Add with carry to A	Immediate
	ADC nn	6D	4	3	Add with carry to A	Absolute
	ADC n	65	3	2	Add with carry to A	Zero Page
	ADC (n,X)	61	6	2	Add with carry to A	Ind X
	ADC (n),Y	71	5+	2	Add with carry to A	Ind Y
	ADC n,X	75	4	2	Add with carry to A	Zero Page X
	ADC nn,X	7D	4+	3	Add with carry to A	Absolute X
	ADC nn,Y	79	4+	3	Add with carry to A	Absolute Y
	AND #n	29	2	2	AND to A	Immediate
	AND nn	2D	4	3	AND to A	Absolute
	AND n	25	3	2	AND to A	Zero Page
	AND (n,X)	21	6	2	AND to A	Ind X
	AND (n),Y	31	5+	2	AND to A	Ind Y
	AND n,X	35	4	2	AND to A	Zero Page X
	AND nn,X	3D	4+	3	AND to A	Absolute X
	AND nn,Y	39	4+	3	AND to A	Absolute Y
	ASL nn	0E	6	3	Arithmetic shift left	Absolute
	ASL n	06	5	2	Arithmetic shift left	Zero Page
	ASL A	0A	2	1	Arithmetic shift left	Accumulator
	ASL n,X	16	6	2	Arithmetic shift left	Zero Page X
	ASL nn,X	1E	7	3	Arithmetic shift left	Absolute X
B	BCC n	90	2+	2	Branch if carry clear (C=0)	Relative
	BCS n	B0	2+	2	Branch if carry set (C=1)	Relative
	BEQ n	F0	2+	2	Branch if equal (Z=1)	Relative
	BNE n	D0	2+	2	Branch if not equal (Z=0)	Relative
	BMI n	30	2+	2	Branch if minus (N=1)	Relative
	BPL n	10	2+	2	Branch if plus (N=0)	Relative
	BVC n	50	2+	2	Branch if ovfl clear (V=0)	Relative
	BVS n	70	2+	2	Branch if ovfl set (V=1)	Relative
	BIT nn	2C	4	3	AND with A (A unchanged)	Absolute
	BIT n	24	3	2	AND with A (A unchanged)	Zero Page
	BRK	00	7	1	Break (force interrupt)	None
C	CLC	18	2	1	Clear carry	None
	CLD	D8	2	1	Clear decimal mode	None
	CLI	58	2	1	Clear IRQ disable	None
	CLV	B8	2	1	Clear overflow	None
	CMP #n	C9	2	2	Compare with A	Immediate
	CMP nn	CD	4	3	Compare with A	Absolute
	CMP n	C5	3	2	Compare with A	Zero Page
	CMP (n,X)	C1	6	2	Compare with A	Ind X
	CMP (n),Y	D1	5+	2	Compare with A	Ind Y
	CMP n,X	D5	4	2	Compare with A	Zero Page X
	CMP nn,X	DD	4+	3	Compare with A	Absolute X
	CMP nn,Y	D9	4+	3	Compare with A	Absolute Y
	CPX #n	E0	2	2	Compare with X	Immediate
	CPX nn	EC	4	3	Compare with X	Absolute
	CPX n	E4	3	2	Compare with X	Zero Page
	CPY #n	C0	2	2	Compare with Y	Immediate
	CPY nn	CC	4	3	Compare with Y	Absolute
	CPY n	C4	3	2	Compare with Y	Zero Page
D	DEC nn	CE	6	3	Decrement by one	Absolute
	DEC n	C6	5	2	Decrement by one	Zero Page
	DEC n,X	D6	6	2	Decrement by one	Zero Page X
	DEC nn,X	DE	7	3	Decrement by one	Absolute X
	DEX	CA	2	1	Decrement X by one	None
	DEY	88	2	1	Decrement Y by one	None
E	EOR #n	49	2	2	XOR to A	Immediate
	EOR nn	4D	4	3	XOR to A	Absolute
	EOR n	45	3	2	XOR to A	Zero Page
	EOR (n,X)	41	6	2	XOR to A	Ind X
	EOR (n),Y	51	5+	2	XOR to A	Ind Y
	EOR n,X	55	4	2	XOR to A	Zero Page X
	EOR nn,X	5D	4+	3	XOR to A	Absolute X
	EOR nn,Y	59	4+	3	XOR to A	Absolute Y
I	INC nn	EE	6	3	Increment by one	Absolute
	INC n	E6	5	2	Increment by one	Zero Page
	INC n,X	F6	6	2	Increment by one	Zero Page X
	INC nn,X	FE	7	3	Increment by one	Absolute X
	INX	E8	2	1	Increment X by one	None
	INY	C8	2	1	Increment Y by one	None
J	JMP nn	4C	3	3	Jump to new location	Absolute
	JMP (nn)	6C	5	3	Jump to new location	Indirect
	JSR nn	20	6	3	Jump to subroutine	Absolute

	INSTRUCTION	OP	C	B	DESCRIPTION	ADDRESSING
L	LDA #n	A9	2	2	Load A	Immediate
	LDA nn	AD	4	3	Load A	Absolute
	LDA n	A5	3	2	Load A	Zero Page
	LDA (n,X)	A1	6	2	Load A	Ind X
	LDA (n),Y	B1	5+	2	Load A	Ind Y
	LDA n,X	B5	4	2	Load A	Zero Page X
	LDA nn,X	BD	4+	3	Load A	Absolute X
	LDA nn,Y	B9	4+	3	Load A	Absolute Y
	LDX #n	A2	2	2	Load X	Immediate
	LDX nn	AE	4	3	Load X	Absolute
	LDX n	A6	3	2	Load X	Zero Page
	LDX nn,Y	BE	4+	3	Load X	Absolute Y
	LDX n,Y	B6	4	2	Load X	Zero Page Y
	LDY #n	A0	2	2	Load Y	Immediate
	LDY nn	AC	4	3	Load Y	Absolute
	LDY n	A4	3	2	Load Y	Zero Page
	LDY n,X	B4	4	2	Load Y	Zero Page X
	LDY nn,X	BC	4+	3	Load Y	Absolute X
	LSR nn	4E	6	3	Logical shift right	Absolute
	LSR n	46	5	2	Logical shift right	Zero Page
	LSR A	4A	2	1	Logical shift right	Accumulator
	LSR n,X	56	6	2	Logical shift right	Zero Page X
	LSR nn,X	5E	7	3	Logical shift right	Absolute X
N	NOP	EA	2	1	No operation	None
O	ORA #n	09	2	2	OR to A	Immediate
	ORA nn	0D	4	3	OR to A	Absolute
	ORA n	05	3	2	OR to A	Zero Page
	ORA (n,X)	01	6	2	OR to A	Ind X
	ORA (n),Y	11	5+	2	OR to A	Ind Y
	ORA n,X	15	4	2	OR to A	Zero Page X
	ORA nn,X	1D	4+	3	OR to A	Absolute X
	ORA nn,Y	19	4+	3	OR to A	Absolute Y
P	PHA	48	3	1	Push A onto stack	None
	PHP	08	3	1	Push P onto stack	None
	PLA	68	4	1	Pull (pop) A from stack	None
	PLP	28	4	1	Pull (pop) P from stack	None
R	ROL nn	2E	6	3	Rotate left through carry	Absolute
	ROL n	26	5	2	Rotate left through carry	Zero Page
	ROL A	2A	2	1	Rotate left through carry	Accumulator
	ROL n,X	36	6	2	Rotate left through carry	Zero Page X
	ROL nn,X	3E	7	3	Rotate left through carry	Absolute X
	ROR nn	6E	6	3	Rotate right through carry	Absolute
	ROR n	66	5	2	Rotate right through carry	Zero Page
	ROR A	6A	2	1	Rotate right through carry	Accumulator
	ROR n,X	76	6	2	Rotate right through carry	Zero Page X
	ROR nn,X	7E	7	3	Rotate right through carry	Absolute X
	RTI	40	6	1	Return from interrupt	None
	RTS	60	6	1	Return from subroutine	None
S	SBC #n	E9	2	2	Subtract with borrow from A	Immediate
	SBC nn	ED	4	3	Subtract with borrow from A	Absolute
	SBC n	E5	3	2	Subtract with borrow from A	Zero Page
	SBC (n,X)	E1	6	2	Subtract with borrow from A	Ind X
	SBC (n),Y	F1	5+	2	Subtract with borrow from A	Ind Y
	SBC n,X	F5	4	2	Subtract with borrow from A	Zero Page X
	SBC nn,X	FD	4+	3	Subtract with borrow from A	Absolute X
	SBC nn,Y	F9	4+	3	Subtract with borrow from A	Absolute Y
	SEC	38	2	1	Set carry	None
	SED	F8	2	1	Set decimal mode	None
	SEI	78	2	1	Set IRQ disable	None
	STA nn	8D	4	3	Store A	Absolute
	STA n	85	3	2	Store A	Zero Page
	STA (n,X)	81	6	2	Store A	Ind X
	STA (n),Y	91	6	2	Store A	Ind Y
	STA n,X	95	4	2	Store A	Zero Page X
	STA nn,X	9D	5	3	Store A	Absolute X
	STA nn,Y	99	5	3	Store A	Absolute Y
	STX nn	8E	4	3	Store X	Absolute
	STX n	86	3	2	Store X	Zero Page
	STX n,Y	96	4	2	Store X	Zero Page Y
	STY nn	8C	4	3	Store Y	Absolute
	STY n	84	3	2	Store Y	Zero Page
	STY n,X	94	4	2	Store Y	Zero Page X
T	TAX	AA	2	1	Transfer A to X	None
	TAY	A8	2	1	Transfer A to Y	None
	TSX	BA	2	1	Transfer S to X	None
	TXA	8A	2	1	Transfer X to A	None
	TXS	9A	2	1	Transfer X to S	None
	TYA	98	2	1	Transfer Y to A	None

Instruction Notes

ADC	A+DATA+C→A
BRK	Ignore I flag, Set B=1 Push return address+1 Push P Jump to IRQ vector
JSR	Push return address-1 Jump absolute
RTI	Pop P, Pop PC
RTS	Pop PC, Increment PC
SBC	A-DATA-$\overline{C}$ →A

Shift Instructions



Added Cycle Time

A (+) in the (C) column for branch instructions means:
Add 0 if branch not taken.
Add 1 if taken within page.
Add 2 if taken across pages.

A (+) in the (C) column for other instructions means:
Add 1 if indexing across page boundary

Assembler Symbols

.	Assembler directive
#	Immediate addressing
$	Hex number prefix
@	Octal number prefix
%	Binary number prefix
'	ASCII character prefix
()	Indirect addressing
;	In col 1 for comment

Intentionally Blank

DO NOT PLACE ON HOT SURFACES

COPYRIGHT © 1980 AND PUBLISHED BY
MICRO LOGIC CORP.
POB 174, HACKENSACK, N.J. 07602
WORLD COPYRIGHT 1980
ALL RIGHTS RESERVED-PRINTED IN U.S.A.

WHILE PREPARED WITH EXTREME CARE,
THERE ARE NO WARRANTIES EXPRESS
OR IMPLIED AS TO MERCHANTABILITY
OR FITNESS FOR PURPOSE.

MICRO CHART
AUTHOR
JAMES D. LEWIS

#101B

AVAILABLE CARDS: Z80, 6502, 8048, 8080/8085, 7400-TTL BASIC ALGORITHMS

100% PLASTIC

INSTANT ACCESS

© 1980 FOR SALE

MICRO LOGIC CORP.
HACKENSACK, NJ

6502 (65XX)

MICROPROCESSOR INSTANT REFERENCE CARD

MICRO® CHART

Hex to Instruction Conversion

LSD→	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
0-	BRK	ORA (n,X)				ORA n	ASL n		PHP	ORA #n	ASL A			ORA nn	ASL nn		0-
1-	BPL n	ORA (n),Y				ORA n,X	ASL n,X		CLC	ORA nn,Y				ORA nn,X	ASL nn,X		1-
2-	JSR nn	AND (n,X)			BIT n	AND n	ROL n		PLP	AND #n	ROL A		BIT nn	AND nn	ROL nn		2-
3-	BMI n	AND (n),Y				AND n,X	ROL n,X		SEC	AND nn,Y				AND nn,X	ROL nn,X		3-
4-	RTI	EOR (n,X)				EOR n	LSR n		PHA	EOR #n	LSR A		JMP nn	EOR nn	LSR nn		4-
5-	BVC n	EOR (n),Y				EOR n,X	LSR n,X		CLI	EOR nn,Y				EOR nn,X	LSR nn,X		5-
6-	RTS	ADC (n,X)				ADC n	ROR n		PLA	ADC #n	ROR A		JMP (nn)	ADC nn	ROR nn		6-
7-	BVS n	ADC (n),Y				ADC n,X	ROR n,X		SEI	ADC nn,Y				ADC nn,X	ROR nn,X		7-
8-		STA (n,X)			STY n	STA n	STX n		DEY		TXA		STY nn	STA nn	STX nn		8-
9-	BCC n	STA (n),Y			STY n,X	STA n,X	STX n,Y		TYA	STA nn,Y	TXS			STA nn,X			9-
A-	LDY #n	LDA (n,X)	LDX #n		LDY n	LDA n	LDX n		TAY	LDA #n	TAX		LDY nn	LDA nn	LDX nn		A-
B-	BCS n	LDA (n),Y			LDY n,X	LDA n,X	LDX n,Y		CLV	LDA nn,Y	TSX		LDY nn,X	LDA nn,X	LDX nn,Y		B-
C-	CPY #n	CMP (n,X)			CPY n	CMP n	DEC n		INY	CMP #n	DEX		CPY nn	CMP nn	DEC nn		C-
D-	BNE n	CMP (n),Y				CMP n,X	DEC n,X		CLD	CMP nn,Y				CMP nn,X	DEC nn,X		D-
E-	CPX #n	SBC (n,X)			CPX n	SBC n	INC n		INX	SBC #n	NOP		CPX nn	SBC nn	INC nn		E-
F-	BEQ n	SBC (n),Y				SBC n,X	INC n,X		SED	SBC nn,Y				SBC nn,X	INC nn,X		F-
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	

Memory Map

ZERO PAGE	0000
	00FF
DATA & STACK*	0100
	01FF
RAM I/O ROM	0200
	FFF9
NMI VECTOR	FFFA&B
RES VECTOR	FFFC&D
IRQ VECTOR	FFFE&F

*In systems with < 512 bytes of RAM the hardware can ignore signal AB8, moving stack into page zero.

Status Flags

MSB LSB
N V – B D I Z C

N=negative result
V=overflow
B=BRK instruction
D=decimal mode
I=IRQ disable
Z=zero result
C=carry=borrow

Note: above is true when flag = 1.

Overflow normally signifies signed arithmetic result is out of range.

When D=1, only ADC and SBC use decimal (BCD) arithmetic.

Interrupts

$\overline{IRQ}$ is low level sensitive.
$\overline{NMI}$ is falling edge sensitive.
Reset sets I=1.
Interrupts are processed by:

1. Push PC of unexecuted instruction.
2. Push P.
3. I=1.
4. Jump via appropriate vector.

Effect on Flags

	N V - B D I Z C
ADC	N V - - - - Z C (1)
AND	N - - - - - Z -
ASL	N - - - - - Z C
BIT	N V - - - - Z - (2)
BRK	- - - 1 - 1 - -
CLC	- - - - - - - 0
CLD	- - - - 0 - - -
CLI	- - - - - 0 - -
CLV	- 0 - - - - - -
CMP	N - - - - - Z C
CPX	N - - - - - Z C
CPY	N - - - - - Z C
DEC	N - - - - - Z -
DEX	N - - - - - Z -
DEY	N - - - - - Z -
EOR	N - - - - - Z -
INC	N - - - - - Z -
INX	N - - - - - Z -
INY	N - - - - - Z -
LDA	N - - - - - Z -
LDX	N - - - - - Z -
LDY	N - - - - - Z -
LSR	0 - - - - - Z C
ORA	N - - - - - Z -
PLA	N - - - - - Z -
PLP	N V - B D I Z C
ROL	N - - - - - Z C
ROR	N - - - - - Z C
RTI	N V - B D I Z C
SBC	N V - - - - Z C (3)
SEC	- - - - - - - 1
SED	- - - - 1 - - -
SEI	- - - - - 1 - -
TAX	N - - - - - Z -
TAY	N - - - - - Z -
TSX	N - - - - - Z -
TXA	N - - - - - Z -
TYA	N - - - - - Z -

(1) If in decimal mode Z flag is invalid.

(2) N = data bit 7
V = data bit 6
Z = AND result

(3) $\overline{C}$ = borrow

Note: unlisted instructions have no effect on flags.

Addressing Modes

Note: Full 2 byte addresses in code, stack, and data areas are stored low byte followed by high byte. Thus, in hex, JMP $1234 is: 4C 34 12

FORM	ADDRESSING	DESCRIPTION
nn	Absolute	Location nn holds data.
nn,X	Absolute X	Location nn+X holds data.
nn,Y	Absolute Y	Location nn+Y holds data.
A	Accumulator	Accumulator holds data.
#n	Immediate	n is data.
(n,X)	Ind X	Location n+X and next of page 0 hold address of data.*,**
(n), Y	Ind Y	Address of data is Y + address held by location n and next of page 0.**
(nn)	Indirect	Location nn and next hold adddress to jump to.**
n	Relative	Address to jump to is n + address of next instruction, with n treated as a signed number.
n	Zero Page	Location n of page 0 holds data.
n,X	Zero Page X	Location n+X of page 0 holds data.
n,Y	Zero Page Y	Location n+Y of page 0 holds data.

*n+X is computed discarding any carry.
**2 bytes must not cross page boundary.

ASCII Character Set

LSD \ MSD	0 000	1 001	2 010	3 011	4 100	5 101	6 110	7 111
0	0000 NUL	DLE	SP	0	@	P	'	p
1	0001 SOH	DC1	!	1	A	Q	a	q
2	0010 STX	DC2	"	2	B	R	b	r
3	0011 ETX	DC3	#	3	C	S	c	s
4	0100 EOT	DC4	$	4	D	T	d	t
5	0101 ENQ	NAK	%	5	E	U	e	u
6	0110 ACK	SYN	&	6	F	V	f	v
7	0111 BEL	ETB	'	7	G	W	g	w
8	1000 BS	CAN	(	8	H	X	h	x
9	1001 HT	EM	)	9	I	Y	i	y
A	1010 LF	SUB	*	:	J	Z	j	z
B	1011 VT	ESC	+	;	K	[	k	{
C	1100 FF	FS	,	<	L	\	l	|
D	1101 CR	GS	-	=	M	]	m	}
E	1110 SO	RS	.	>	N	↑	n	~
F	1111 SI	US	/	?	O	←	o	DEL

Miscellaneous

S points to next free byte of stack.

Stack push decrements S.

In pushing PC, high byte is pushed first.

Pre 6/76 chips have no ROR instruction.

65XX is a totally software compatible family.

This card is based on specifications from MOS Technology, Inc.

Registers

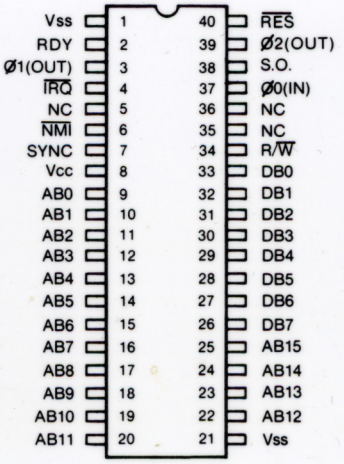
A ACCUMULATOR
Y Y INDEX REG
X X INDEX REG
PC PROGRAM COUNTER
S STACK PNTR
P FLAGS

A, Y, X, S, P = 1 byte.
Only PC is 2 bytes.

Hex and Decimal Conversion

LSD→	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	0
1	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	1
2	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	2
3	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	3
4	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	4
5	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	5
6	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	6
7	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	7
8	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	8
9	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	9
A	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	A
B	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	B
C	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	C
D	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	D
E	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	E
F	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255	F
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	

6502 Pins



Abbreviations

B = number of Bytes
C = number of Cycles. also Carry.

n = 1 byte quantity
nn = 2 byte quantity

IRQ = Interrupt ReQuest
NMI = Non Maskable Interrupt
RES = RESet
XOR = eXclusive OR
(00→0 01→1 10→1 11→0)

A,P,S,X,Y,PC=see "Registers"
N,V,B,D,I,Z,C = see "Status Flags"
.#$@%'(); = see "Assembler Symbols"

Unsigned Comparisons

example: CMP #n

A < n	BCC YES
A = n	BEQ YES
A > n	BCC NO BNE YES
A ≥ n	BCS YES
A ≠ n	BNE YES
A ≤ n	BCC YES BEQ YES

YES represents label for code to be executed if condition is true. For > & ≤, test requires both instructions.

Internally, A–n is computed to determine N,Z,C flags.