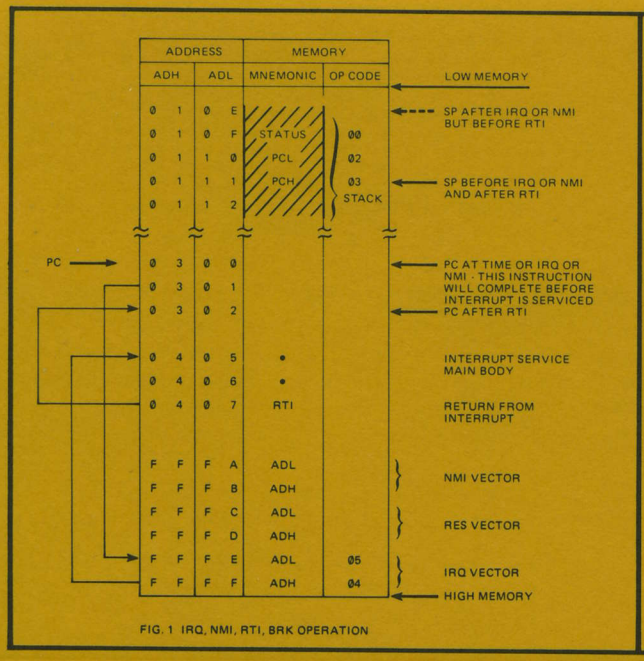


FIG. 1 IRQ, NMI, RTI, BRK OPERATION

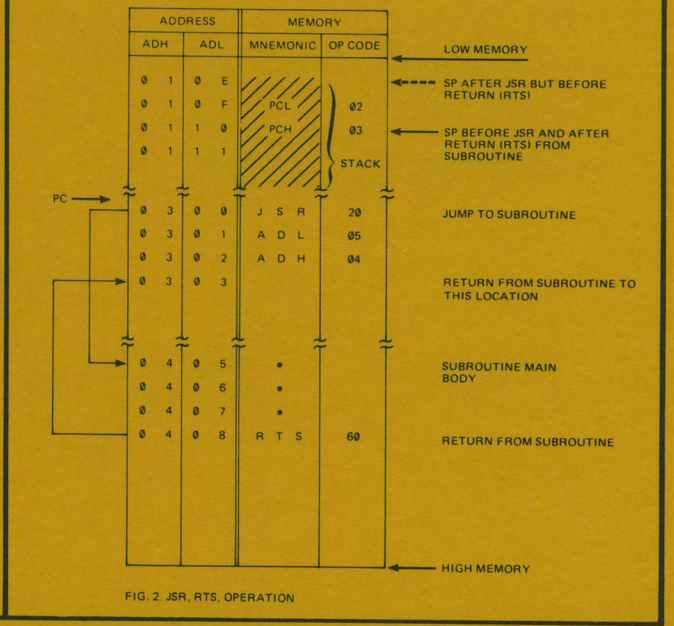


FIG. 2 JSR, RTS, OPERATION

ASSEMBLER DIRECTIVES

- OPT – SPECIFIES OPTIONS FOR ASSEMBLY
  OPTIONS ARE: (OPTIONS LISTED FIRST ARE THE DEFAULT VALUES).
  NOC (COU OR CNT) – DO NOT LIST ALL INSTRUCTIONS AND THEIR USAGE.
  NOG (GEN) – DO NOT GENERATE MORE THAN ONE LINE OF CODE FOR ASCII STRINGS.
  XRE (NOX) – PRODUCE A CROSS-REFERENCE LIST IN THE SYMBOL TABLE.
  ERR (NOE) – CREATE AN ERROR FILE.
  MEM (NOM) – CREATE AN ASSEMBLER OBJECT OUTPUT FILE.
  LIS (NOL) – PRODUCE A FULL ASSEMBLY LISTING.
- BYTE – PRODUCES A SINGLE BYTE IN MEMORY EQUAL TO EACH OPERAND SPECIFIED.
- WORD – PRODUCES AN ADDRESS (2 BYTES) IN MEMORY EQUAL TO EACH OPERAND SPECIFIED.
- DBYTE – PRODUCES TWO BYTES IN MEMORY EQUAL TO EACH OPERAND SPECIFIED.
- SKIP – GENERATE THE NUMBER OF BLANK LINES SPECIFIED BY THE OPERAND.
- PAGE – ADVANCE THE LISTING TO THE TOP OF A NEW PAGE AND CHANGE TITLE.
- END – DEFINES THE END OF A SOURCE PROGRAM.
- * = – DEFINES THE BEGINNING OF A NEW PROGRAM COUNTER SEQUENCE.

LABELS

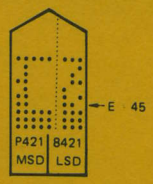
LABELS ARE THE FIRST FIELD AND MUST BE FOLLOWED BY AT LEAST ONE SPACE.
LABELS CAN BE UP TO 6 ALPHANUMERIC CHARACTERS LONG AND MUST BEGIN WITH AN ALPHA CHARACTER.
A,X,Y,S,P AND THE 56 OPCODES ARE RESERVED AND CANNOT BE USED AS LABELS.
LABEL = EXPRESSION CAN BE USED TO EQUATE LABELS TO VALUES.
LABEL *=* +N CAN BE USED TO RESERVE AREAS IN MEMORY.

CHARACTERS USED AS SPECIAL PREFIXES:

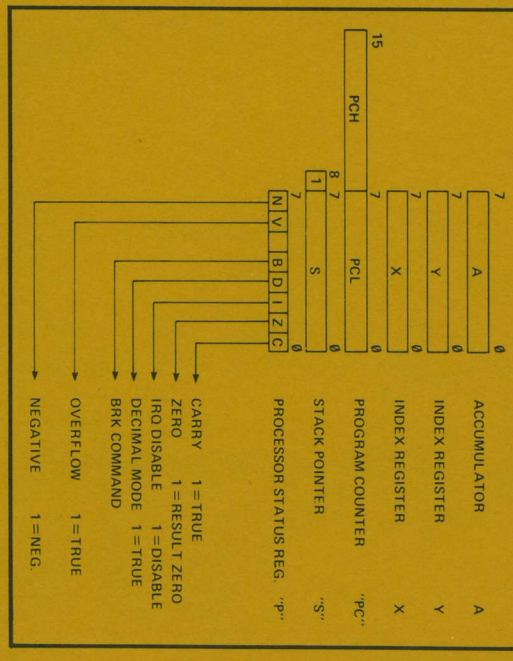
- • INDICATES AN ASSEMBLER DIRECTIVE
- = SPECIFIES THE IMMEDIATE MODE OF ADDRESSING
- $ SPECIFIES A HEXADECIMAL NUMBER
- @ SPECIFIES AN OCTAL NUMBER
- % SPECIFIES A BINARY NUMBER
- ' SPECIFIES AN ASCII LITERAL CHARACTER
- ( ) INDICATES INDIRECT ADDRESSING
- ; INDICATES FOLLOWING TEXT ARE COMMENTS
- < SPECIFIES LOWER HALF OF A 16 BIT VALUE
- > SPECIFIES UPPER HALF OF A 16 BIT VALUE

ASCII CHARACTER SET (7-BIT CODE)

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	ENG	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	VS	/	?	O	←	o	DEL



C016892 Rev. 1



ATARI®
ATARI, INC.
1265 BORREGAS AVENUE
SUNNYVALE, CA 94086
A Warner Communications Company

INSTRUCTIONS		IMMEDIATE			ABSOLUTE			ZERO PAGE			ACCUM			IMPLIED			(IND, X)			(IND), Y			Z, PAGE, X			ABS, X			ABS, Y			RELATIVE			INDIRECT			Z, PAGE, Y			CONDITION CODES					
MNEMONIC	OPERATION	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	N	Z	C	I	D	V
ADC	A + M + C → A (4) (1)	69	2	2	6D	4	3	65	3	2							61	6	2	71	5	2	75	4	2	7D	4	3	79	4	3										✓	✓	✓	–	–	✓
AND	A ∧ M → A (1)	29	2	2	2D	4	3	25	3	2							21	6	2	31	5	2	35	4	2	3D	4	3	39	4	3										✓	✓	–	–	–	–
ASL	C ← 7 0 ← 0				0E	6	3	06	5	2	0A	2	1										16	6	2	1E	7	3													✓	✓	✓	–	–	–
BCC	BRANCH ON C=0 (2)																															90	2	2							–	–	–	–	–	–
BCS	BRANCH ON C=1 (2)																															B0	2	2							–	–	–	–	–	–
BEQ	BRANCH ON Z=1 (2)																															F0	2	2							–	–	–	–	–	–
BIT	A ∧ M				2C	4	3	24	3	2																															M_7	✓	–	–	–	M_6
BMI	BRANCH ON N=1 (2)																															30	2	2							–	–	–	–	–	–
BNE	BRANCH ON Z=0 (2)																															D0	2	2							–	–	–	–	–	–
BPL	BRANCH ON N=0 (2)																															10	2	2							–	–	–	–	–	–
BRK	(See Fig. 1)													00	7	1																									–	–	–	–	–	–
BVC	BRANCH ON V=0 (2)																															50	2	2							–	–	–	–	–	–
BVS	BRANCH ON V=1 (2)																															70	2	2							–	–	–	–	–	–
CLC	0 → C													18	2	1																									–	–	0	–	–
CLD	0 → D													D8	2	1																									–	–	–	–	0	–
CLI	0 → I													58	2	1																									–	–	–	0	–	–
CLV	0 → V													B8	2	1																									–	–	–	–	–	0
CMP	A-M (1)	C9	2	2	CD	4	3	C5	3	2							C1	6	2	D1	5	2	D5	4	2	DD	4	3	D9	4	3										✓	✓	✓	–	–	–
CPX	X-M	E0	2	2	EC	4	3	E4	3	2																															✓	✓	✓	–	–	–
CPY	Y-M	C0	2	2	CC	4	3	C4	3	2																															✓	✓	✓	–	–	–
DEC	M-1 → M				CE	6	3	C6	5	2													D6	6	2	DE	7	3													✓	✓	–	–	–	–
DEX	X-1 → X													CA	2	1																									✓	✓	–	–	–	–
DEY	Y-1 → Y													88	2	1																									✓	✓	–	–	–	–
EOR	A ∀ M → A (1)	49	2	2	4D	4	3	45	3	2							41	6	2	51	5	2	55	4	2	5D	4	3	59	4	3										✓	✓	–	–	–	–
INC	M + 1 → M				EE	6	3	E6	5	2													F6	6	2	FE	7	3													✓	✓	–	–	–	–
INX	X + 1 → X													E8	2	1																									✓	✓	–	–	–	–
INY	Y + 1 → Y													C8	2	1																									✓	✓	–	–	–	–
JMP	JUMP TO NEW LOC				4C	3	3																													6C	5	3				–	–	–	–	–	–
JSR	(See Fig. 2) JUMP SUB				20	6	3																																		–	–	–	–	–	–
LDA	M → A (1)	A9	2	2	AD	4	3	A5	3	2							A1	6	2	B1	5	2	B5	4	2	BD	4	3	B9	4	3										✓	✓	–	–	–	–

		IMMEDIATE			ABSOLUTE			ZERO PAGE			ACCUM.			IMPLIED			(IND, X)			(IND), Y			Z, PAGE, X			ABS, X			ABS, Y			RELATIVE			INDIRECT			Z, PAGE, Y			CONDITION CODES					
MNEMONIC	OPERATION	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	N	Z	C	I	D	V
LDX	M → X (1)	A2	2	2	AE	4	3	A6	3	2																			BE	4	3										✓	✓	–	–	–	–
LDY	M → Y (1)	A0	2	2	AC	4	3	A4	3	2													B4	4	2	BC	4	3													✓	✓	–	–	–	–
LSR	0 → 7 0 → C				4E	6	3	46	5	2	4A	2	1										56	6	2	5E	7	3													0	✓	✓	–	–	–
NOP	NO OPERATION													EA	2	1																									–	–	–	–	–	–
ORA	A ∨ M → A	09	2	2	0D	4	3	05	3	2							01	6	2	11	5	2	15	4	2	1D	4	3	19	4	3										✓	✓	–	–	–	–
PHA	A → Ms S-1 → S													48	3	1																									–	–	–	–	–	–
PHP	P → Ms S-1 → S													08	3	1																									–	–	–	–	–	–
PLA	S + 1 → S Ms → A													68	4	1																									✓	✓	–	–	–	–
PLP	S + 1 → S Ms → P													28	4	1																									(RESTORED)					
ROL	7 0 C				2E	6	3	26	5	2	2A	2	1										36	6	2	3E	7	3													✓	✓	✓	–	–	–
ROR	C 7 0				6E	6	3	66	5	2	6A	2	1										76	6	2	7E	7	3													✓	✓	✓	–	–	–
RTI	(See Fig. 1) RTRN INT.													40	6	1																									(RESTORED)					
RTS	(See Fig. 2) RTRN SUB													60	6	1																									–	–	–	–	–	–
SBC	A-M-$\bar{C}$ → A (1)	E9	2	2	ED	4	3	E5	3	2							E1	6	2	F1	5	2	F5	4	2	FD	4	3	F9	4	3										✓	✓	(3)	–	–	✓
SEC	1 → C													38	2	1																									–	–	1	–	–	–
SED	1 → D													F8	2	1																									–	–	–	1	–	–
SEI	1 → I													78	2	1																									–	–	–	1	–	–
STA	A → M				8D	4	3	85	3	2							81	6	2	91	6	2	95	4	2	9D	5	3	99	5	3										–	–	–	–	–	–
STX	X → M				8E	4	3	86	3	2																															96	4	2	–	–	–	–	–	–
STY	Y → M				8C	4	3	84	3	2													94	4	2																	–	–	–	–	–	–
TAX	A → X													AA	2	1																									✓	✓	–	–	–	–
TAY	A → Y													A8	2	1																									✓	✓	–	–	–	–
TSX	S → X													BA	2	1																									✓	✓	–	–	–	–
TXA	X → A													8A	2	1																									✓	✓	–	–	–	–
TXS	X → S													9A	2	1																									–	–	–	–	–	–
TYA	Y → A													98	2	1																									✓	✓	–	–	–	–

(1) ADD 1 TO "N" IF PAGE BOUNDARY IS CROSSED
(2) ADD 1 TO "N" IF BRANCH OCCURS TO SAME PAGE
ADD 2 TO "N" IF BRANCH OCCURS TO DIFFERENT PAGE
(3) CARRY NOT = BELOW
(4) IF IN DECIMAL MODE Z FLAG IS INVALID
ACCUMULATOR MUST BE CHECKED FOR ZERO RESULT

X INDEX X
Y INDEX Y
A ACCUMULATOR
M MEMORY PER EFFECTIVE ADDRESS
Ms MEMORY PER STACK POINTER

+ ADD
- SUBTRACT
∧ AND
∨ OR
∀ EXCLUSIVE OR
✓ MODIFIED
– NOT MODIFIED
M_7 MEMORY BIT 7
M_6 MEMORY BIT 6
N NO. CYCLES
NO. BYTES

OP-CODE TABLE

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

IMM - IMMEDIATE ADDRESSING – THE OPERAND IS CONTAINED IN THE SECOND BYTE OF THE INSTRUCTION.

ABS - ABSOLUTE ADDRESSING – THE SECOND BYTE OF THE INSTRUCTION CONTAINS THE 8 LOW ORDER BITS OF THE EFFECTIVE ADDRESS. THE THIRD BYTE CONTAINS THE 8 HIGH ORDER BITS OF THE EFFECTIVE ADDRESS.

Z, PAGE - ZERO PAGE ADDRESSING – SECOND BYTE CONTAINS THE 8 LOW ORDER BITS OF THE EFFECTIVE ADDRESS. THE 8 HIGH ORDER BITS ARE ZERO.

A, - ACCUMULATOR – ONE BYTE INSTRUCTION OPERATING ON THE ACCUMULATOR.

Z, PAGE, X - Z PAGE, Y - ZERO PAGE INDEXED – THE SECOND BYTE OF THEINSTRUCTION IS ADDED TO THE INDEX (CARRY IS DROPPED) TO FORM THE LOW ORDER BYTE OF THE EA. THE HIGH ORDER BYTE OF THE EA IS ZEROS.

ABS, X ABS, Y ABSOLUTE INDEXED – THE EFFECTIVE ADDRESS IS FORMED BY ADDING THE INDEX TO THE SECOND AND THIRD BYTE OF THE INSTRUCTION.

(IND, X) - INDEXED INDIRECT – THE SECOND BYTE OF THE INSTRUCTION IS ADDED TO THE X INDEX, DISCARDING THE CARRY, THE RESULTS POINTS TO A LOCATION ON PAGE ZERO WHICH CONTAINS THE 8 LOW ORDER BITS OF THE EA. THE NEXT BYTE CONTAINS THE 8 HIGH ORDER BITS.

(IND), Y - INDIRECT INDEXED – THE SECOND BYTE OF THE INSTRUCTION POINTS TO A LOCATION IN PAGE ZERO. THE CONTENTS OF THIS MEMORY LOCATION IS ADDED TO THE Y INDEX, THE RESULT BEING THE LOW ORDER EIGHT BITS OF THE EA. THE CARRY FROM THIS OPERATION IS ADDED TO THE CONTENTS OF THE NEXT PAGE ZERO LOCATION, THE RESULTS BEING THE 8 HIGH ORDER BITS OF THE EA.