HEXADECIMAL AND DECIMAL CONVERSION

HEXADECIMAL COLUMNS												
6		5		4		3		2		1	
HEX	DEC	HEX	DEC	HEX	DEC	HEX	DEC	HEX	DEC	HEX	DEC
0	0	0	0	0	0	0	0	0	0	0	0
1	1,048,576	1	65,536	1	4,096	1	256	1	16	1	1
2	2,097,152	2	131,072	2	8,192	2	512	2	32	2	2
3	3,145,728	3	196,608	3	12,288	3	768	3	48	3	3
4	4,194,304	4	262,144	4	16,384	4	1,024	4	64	4	4
5	5,242,880	5	327,680	5	20,480	5	1,280	5	80	5	5
6	6,291,456	6	393,216	6	24,576	6	1,536	6	96	6	6
7	7,340,032	7	458,752	7	28,672	7	1,792	7	112	7	7
8	8,388,608	8	524,288	8	32,768	8	2,048	8	128	8	8
9	9,437,184	9	589,824	9	36,864	9	2,304	9	144	9	9
A	10,485,760	A	655,360	A	40,960	A	2,560	A	160	A	10
B	11,534,336	B	720,896	B	45,056	B	2,816	B	176	B	11
C	12,582,912	C	786,432	C	49,152	C	3,072	C	192	C	12
D	13,631,488	D	851,968	D	53,248	D	3,328	D	208	D	13
E	14,680,064	E	917,504	E	57,344	E	3,584	E	224	E	14
F	15,728,640	F	983,040	F	61,440	F	3,840	F	240	F	15
7654		3210		7654		3210		7654		3210	
Byte				Byte				Byte			

POWERS OF 2

2^n	n
256	8
512	9
1 024	10
2 048	11
4 096	12
8 192	13
16 384	14
32 768	15
65 536	16
131 072	17
262 144	18
524 288	19
1 048 576	20
2 097 152	21
4 194 304	22
8 388 608	23
16 777 216	24

$2^0 = 16^0$	
$2^4 = 16^1$	
$2^8 = 16^2$	
$2^{12} = 16^3$	
$2^{16} = 16^4$	
$2^{20} = 16^5$	
$2^{24} = 16^6$	
$2^{28} = 16^7$	
$2^{32} = 16^8$	
$2^{36} = 16^9$	
$2^{40} = 16^{10}$	
$2^{44} = 16^{11}$	
$2^{48} = 16^{12}$	
$2^{52} = 16^{13}$	
$2^{56} = 16^{14}$	
$2^{60} = 16^{15}$	

POWERS OF 16

16^n	n
1	0
16	1
256	2
4 096	3
65 536	4
1 048 576	5
16 777 216	6
268 435 456	7
4 294 967 296	8
68 719 476 736	9
1 099 511 627 776	10
17 592 186 044 416	11
281 474 976 710 656	12
4 503 599 627 370 496	13
72 057 594 037 927 936	14
1 152 921 504 606 846 976	15

BACKWARD RELATIVE BRANCH TABLE

MSD \ LSD	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
8	128	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113
9	112	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97
A	96	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81
B	80	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65
C	64	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49
D	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33
E	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17
F	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1

FORWARD RELATIVE BRANCH TABLE

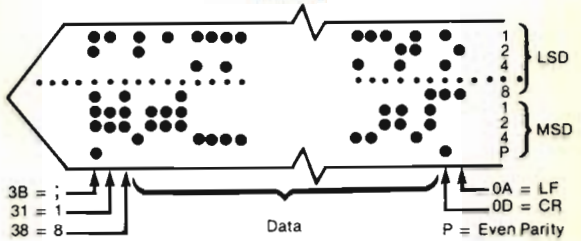
MSD \ LSD	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
2	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
3	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
4	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
5	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
6	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
7	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127

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	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	VS	/	?	O	←	o	DEL

NUL — Null
SOH — Start of Heading
STX — Start of Text
ETX — End of Text
EOT — End of Transmission
ENQ — Enquiry
ACK — Acknowledge
BEL — Bell
BS — Backspace
HT — Horizontal Tabulation
LF — Line Feed
VT — Vertical Tabulation
FF — Form Feed
CR — Carriage Return
SO — Shift Out
SI — Shift In
DLE — Data Link Escape
DC — Device Control
NAK — Negative Acknowledge
SYN — Synchronous Idle
ETB — End of Transmission Block
CAN — Cancel
EM — End of Medium
SUB — Substitute
ESC — Escape
FS — File Separator
GS — Group Separator
RS — Record Separator
US — Unit Separator
SP — Space (Blank)
DEL — Delete

PUNCHED TAPE FORMAT (ASCII)



OBJECT CODE RECORD FORMAT (ASCII)

Data Record: ;$N_1N_0A_3A_2A_1A_0D_1D_0D_1D_0 \ldots D_1D_0X_3X_2X_1X_0$CR LF (1, 2, n)

Last Record: ;$00C_3C_2C_1C_0X_3X_2X_1X_0$

where:

Two hex digits (MSD & LSD) = 1 ASCII character

; = Start of record (ASCII '3B')

N_1N_0 = No. of data bytes in record (hex.) 18_{16} max.
= 00 for last record

$A_3A_2A_1A_0$ = Starting address (hex)

D_1D_0 = Two hexadecimal digits = One 8-bit data byte

$X_3X_2X_1X_0$ = Record checksum (hex.) Hex sum of all characters in the record except ; and checksum, truncated to 16 bits (four hex digits)

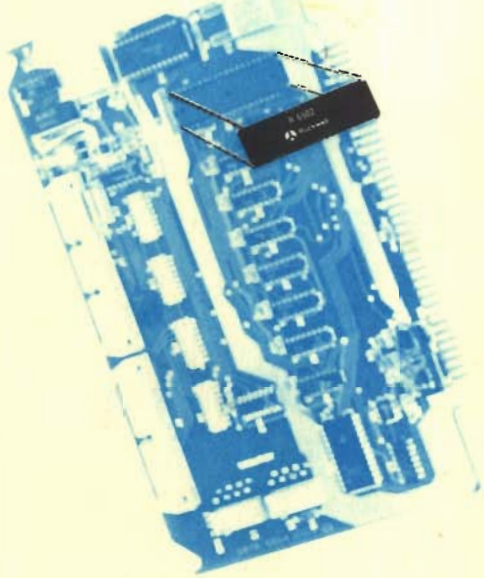
$C_3C_2C_1C_0$ = Total number of records (hex)

CR = Carriage Return (ASCII '0D')

LF = Line Feed (ASCII '0A')

Rockwell

R6500 Microprocessor Programming Reference Card



Rockwell Microelectronic Devices Sales Offices

WESTERN REGION, U.S.A.
3310 Miraloma Avenue
P.O. Box 3669
Anaheim, CA 92803
Phone: (714) 632-3698

EASTERN REGION, U.S.A.
Carolier Office Building
850-870 U.S. Route 1
North Brunswick, New Jersey 08902
Phone: (201) 246-3630

MIDWEST REGION, U.S.A.
1011 E. Touhy — Suite 245
Des Plaines, Illinois 60018
Phone: (312) 297-8862

CENTRAL REGION, U.S.A.
Contact Robert O. Whitesell & Associates
6691 East Washington Street
Indianapolis, Indiana 46219
Attn.: Milt Gamble, Mgr.
Phone: (317) 359-9283

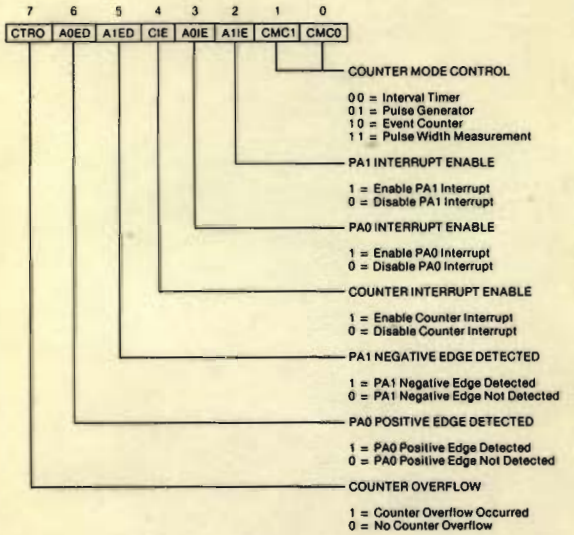
FAR EAST
Rockwell International Overseas Corp.
Ichiban-cho Central Building
22-1 Ichiban-cho, Chiyoda-ku
Tokyo 102, Japan
Phone: 265-8808

EUROPE
Rockwell International GmbH
Microelectronic Devices
Fraunhoferstrasse 11
D-8033 Munchen-Martinsried
Germany
Phone: (089) 859-9575

DOC. NO. 29650N50
REV. 2, OCTOBER 1978

R6500/1 CONTROL REGISTER

The Control Register controls four Counter operating modes and three maskable interrupts. It also reports the status of three interrupt conditions.



R6500/1 COUNTER MODES

INTERVAL TIMER (MODE 0)

In this mode the Counter is free-running, and decrements at the ø2 clock rate. The CNTR line is held in the high state.

PULSE GENERATOR (MODE 1)

In this mode the Counter is free-running, and decrements at the ø2 clock rate. The CNTR line toggles from one state to the other when Counter overflow occurs.

EVENT COUNTER (MODE 2)

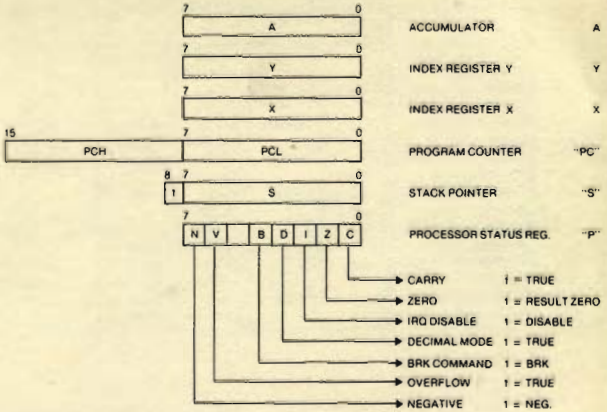
In this mode the CNTR line is used as an event input line. The Counter decrements each time a rising edge is detected on CNTR.

PULSE WIDTH MEASUREMENT (MODE 3)

This mode allows accurate measurement of the duration of a low state on the CNTR line. The Counter decrements at the ø2 clock rate as long as the CNTR is held in the low state, and stops when CNTR switches to the high state.

Note: In all modes Counter overflow sets the Control Register CTRO status bit and causes the Counter to be preset to the Latch value.

PROCESSOR PROGRAMMING MODEL



MACHINE INSTRUCTIONS

ADC	Add Memory to Accumulator with Carry
AND	AND Memory with Accumulator
ASL	Shift Left One Bit (Memory or Accumulator)
BCC	Branch on Carry Clear
BCS	Branch on Carry Set
BEQ	Branch on Result Zero
BIT	Test Bits in Memory with Accumulator
BMI	Branch on Result Minus
BNE	Branch on Result Not Zero
BPL	Branch on Result Plus
BRK	Force Break
BVC	Branch on Overflow Clear
BVS	Branch on Overflow Set
CLC	Clear Carry Flag
CLD	Clear Decimal Mode
CLI	Clear Interrupt Disable Bit
CLV	Clear Overflow Flag
CMP	Compare Memory and Accumulator
CPX	Compare Memory and Index X
CPY	Compare Memory and Index Y
DEC	Decrement Memory by One
DEX	Decrement Index X by One
DEY	Decrement Index Y by One
EOR	Exclusive-OR Memory with Accumulator
INC	Increment Memory by One
INX	Increment Index X by One
INY	Increment Index Y by One
JMP	Jump to New Location
JSR	Jump to New Location Saving Return Address
LDA	Load Accumulator with Memory
LDX	Load Index X with Memory
LDY	Load Index Y with Memory
LSR	Shift Right One Bit (Memory or Accumulator)
NOP	No Operation
ORA	OR Memory with Accumulator
PHA	Push Accumulator on Stack
PHP	Push Processor Status on Stack
PLA	Pull Accumulator from Stack
PLP	Pull Processor Status from Stack
ROL	Rotate One Bit Left (Memory or Accumulator)
ROR	Rotate One Bit Right (Memory or Accumulator)
RTI	Return from Interrupt
RTS	Return from Subroutine
SBC	Subtract Memory from Accumulator with Borrow
SEC	Set Carry Flag
SED	Set Decimal Mode
SEI	Set Interrupt Disable Status
STA	Store Accumulator in Memory
STX	Store Index X in Memory
STY	Store Index Y in Memory
TAX	Transfer Accumulator to Index X
TAY	Transfer Accumulator to Index Y
TSX	Transfer Stack Pointer to Index X
TXA	Transfer Index X to Accumulator
TXS	Transfer Index X to Stack Pointer
TYA	Transfer Index Y to Accumulator

COMPARE INSTRUCTION RESULTS

Condition	N	Z	C
A, X, or Y < Memory	1*	0	0
A, X, or Y = Memory	0	1	1
A, X, or Y > Memory	0*	0	1

*N is valid only for 2's complement compare.

R6500 INSTRUCTION SET

INSTRUCTIONS		IMMEDIATE			ABSOLUTE			ZERO PAGE			ACCUM.			IMPLIED			(IND, X)		
MNEMONIC	OPERATION	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#
ADC	A + M + C → A (4) (11)	69	2	2	6D	4	3	65	3	2							61	6	2
AND	A∧M → A (11)	29	2	2	2D	4	3	25	3	2							21	6	2
ASL	C ← 7 6 5 4 3 2 1 0 ← 0				0E	6	3	06	5	2	0A	2	1						
BCC	BRANCH ON C = 0 (2)																		
BCS	BRANCH ON C = 1 (2)																		
BEQ	BRANCH ON Z = 1 (2)																		
BIT	A∧M				2C	4	3	24	3	2									
BMI	BRANCH ON N = 1 (2)																		
BNE	BRANCH ON Z = 0 (2)																		
BPL	BRANCH ON N = 0 (2)																		
BRK	BREAK (See Fig. 1)													00	7	1			
BVC	BRANCH ON V = 0 (2)																		
BVS	BRANCH ON V = 1 (2)																		
CLC	0 → C													18	2	1			
CLD	0 → D													D8	2	1			
CLI	0 → I													58	2	1			
CLV	0 → V													B8	2	1			
CMP	A − M	C9	2	2	CD	4	3	C5	3	2							C1	6	2
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									
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
INC	M + 1 → M				EE	6	3	E6	5	2									
INX	X + 1 → X													E8	2	1			
INY	Y + 1 → Y													C8	2	1			
JMP	JUMP TO NEW LOC				4C	3	3												
JSR	JUMP SUB (See Fig. 2)				20	6	3												
LDA	M → A (1)	A9	2	2	AD	4	3	A5	3	2							A1	6	2
LDX	M → X (1)	A2	2	2	AE	4	3	A6	3	2									
LDY	M → Y (1)	A0	2	2	AC	4	3	A4	3	2									
LSR	0 → 7 6 5 4 3 2 1 0 → C				4E	6	3	46	5	2	4A	2	1						
NOP	NO OPERATION													EA	2	1			
ORA	A∨M → A	09	2	2	0D	4	3	05	3	2							01	6	2
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			
ROL	7 6 5 4 3 2 1 0 ← C				2E	6	3	26	5	2	2A	2	1						
ROR	C → 7 6 5 4 3 2 1 0				6E	6	3	66	5	2	6A	2	1						
RTI	RTRN INT (See Fig. 1)													40	6	1			
RTS	RTRN SUB (See Fig. 2)													60	6	1			
SBC	A − M − C → A (1)	E9	2	2	ED	4	3	E5	3	2							E1	6	2
SEC	1 → C													38	2	1			
SED	1 → D													F8	2	1			
SEI	1 → I													78	2	1			
STA	A → M				8D	4	3	85	3	2							81	6	2
STX	X → M				8E	4	3	86	3	2									
STY	Y → M				8C	4	3	84	3	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			

(IND), Y			Z. PAGE, X			ABS. X			ABS. Y			RELATIVE			INDIRECT			Z. PAGE, Y			PROCESSOR STATUS CODES	
OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	7 6 5 4 3 2 1 0 N V ▪ B D I Z C	MNEMONIC
71	5	2	75	4	2	7D	4	3	79	4	3										N ✓ ✓ ▪ ▪ ▪ Z C	ADC
31	5	2	35	4	2	3D	4	3	39	4	3										N ✓ ▪ ▪ ▪ ▪ Z ▪	AND
			16	6	2	1E	7	3													N ▪ ▪ ▪ ▪ ▪ Z C	ASL
												90	2	2							▪ ▪ ▪ ▪ ▪ ▪ ▪ ▪	BCC
												B0	2	2							▪ ▪ ▪ ▪ ▪ ▪ ▪ ▪	BCS
												F0	2	2							▪ ▪ ▪ ▪ ▪ ▪ ▪ ▪	BEQ
																					M7 M6 ▪ ▪ ▪ ▪ Z ▪	BIT
												30	2	2							▪ ▪ ▪ ▪ ▪ ▪ ▪ ▪	BMI
												D0	2	2							▪ ▪ ▪ ▪ ▪ ▪ ▪ ▪	BNE
												10	2	2							▪ ▪ ▪ ▪ ▪ ▪ ▪ ▪	BPL
																					▪ ▪ ▪ 1 ▪ 1 ▪ ▪	BRK
												50	2	2							▪ ▪ ▪ ▪ ▪ ▪ ▪ ▪	BVC
												70	2	2							▪ ▪ ▪ ▪ ▪ ▪ ▪ ▪	BVS
																					▪ ▪ ▪ ▪ ▪ ▪ ▪ 0	CLC
																					▪ ▪ ▪ ▪ 0 ▪ ▪ ▪	CLD
																					▪ ▪ ▪ ▪ 0 ▪ ▪ ▪	CLI
																					▪ 0 ▪ ▪ ▪ ▪ ▪ ▪	CLV
D1	5	2	D5	4	2	DD	4	3	D9	4	3										N ▪ ▪ ▪ ▪ ▪ Z C	CMP
																					N ▪ ▪ ▪ ▪ ▪ Z C	CPX
																					N ▪ ▪ ▪ ▪ ▪ Z C	CPY
			D6	6	2	DE	7	3													N ▪ ▪ ▪ ▪ ▪ Z ▪	DEC
																					N ▪ ▪ ▪ ▪ ▪ Z ▪	DEX
																					N ▪ ▪ ▪ ▪ ▪ Z ▪	DEY
51	5	2	55	4	2	5D	4	3	59	4	3										N ▪ ▪ ▪ ▪ ▪ Z ▪	EOR
			F6	6	2	FE	7	3													N ▪ ▪ ▪ ▪ ▪ Z ▪	INC
																					N ▪ ▪ ▪ ▪ ▪ Z ▪	INX
																					N ▪ ▪ ▪ ▪ ▪ Z ▪	INY
															6C	5	3				▪ ▪ ▪ ▪ ▪ ▪ ▪ ▪	JMP
																					▪ ▪ ▪ ▪ ▪ ▪ ▪ ▪	JSR
B1	5	2	B5	4	2	BD	4	3	B9	4	3										N ▪ ▪ ▪ ▪ ▪ Z ▪	LDA
									BE	4	3							B6	4	2	N ▪ ▪ ▪ ▪ ▪ Z ▪	LDX
			B4	4	2	BC	4	3													N ▪ ▪ ▪ ▪ ▪ Z ▪	LDY
			56	6	2	5E	7	3													0 ▪ ▪ ▪ ▪ ▪ Z C	LSR
																					▪ ▪ ▪ ▪ ▪ ▪ ▪ ▪	NOP
11	5	2	15	4	2	1D	4	3	19	4	3										N ▪ ▪ ▪ ▪ ▪ Z ▪	ORA
																					▪ ▪ ▪ ▪ ▪ ▪ ▪ ▪	PHA
																					▪ ▪ ▪ ▪ ▪ ▪ ▪ ▪	PHP
																					N ▪ ▪ ▪ ▪ ▪ Z ▪	PLA
																					(RESTORED)	PLP
			36	6	2	3E	7	3													N ▪ ▪ ▪ ▪ ▪ Z C	ROL
			76	6	2	7E	7	3													N ▪ ▪ ▪ ▪ ▪ Z C	ROR
																					(RESTORED)	RTI
																					▪ ▪ ▪ ▪ ▪ ▪ ▪ ▪	RTS
F1	5	2	F5	4	2	FD	4	3	F9	4	3										N ✓ ▪ ▪ ▪ ▪ Z (3)	SBC
																					▪ ▪ ▪ ▪ ▪ ▪ ▪ 1	SEC
																					▪ ▪ ▪ ▪ 1 ▪ ▪ ▪	SED
																					▪ ▪ ▪ ▪ ▪ 1 ▪ ▪	SEI
91	6	2	95	4	2	9D	5	3	99	5	3										▪ ▪ ▪ ▪ ▪ ▪ ▪ ▪	STA
																		96	4	2	▪ ▪ ▪ ▪ ▪ ▪ ▪ ▪	STX
			94	4	2																▪ ▪ ▪ ▪ ▪ ▪ ▪ ▪	STY
																					N ▪ ▪ ▪ ▪ ▪ Z ▪	TAX
																					N ▪ ▪ ▪ ▪ ▪ Z ▪	TAY
																					N ▪ ▪ ▪ ▪ ▪ Z ▪	TSX
																					N ▪ ▪ ▪ ▪ ▪ Z ▪	TXA
																					▪ ▪ ▪ ▪ ▪ ▪ ▪ ▪	TXS
																					N ▪ ▪ ▪ ▪ ▪ Z ▪	TYA

(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 = BORROW

(4) IF IN DECIMAL MODE, Z FLAG IS INVALID
ACCUMULATOR MUST BE CHECKED FOR ZERO RESULT

X	INDEX X	+	ADD	M_7	MEMORY BIT 7
Y	INDEX Y	−	SUBTRACT	M_6	MEMORY BIT 6
A	ACCUMULATOR	∧	AND	n	NO. CYCLES
M	MEMORY PER EFFECTIVE ADDRESS	∨	OR	#	NO. BYTES
Ms	MEMORY PER STACK POINTER	∨	EXCLUSIVE OR		

OPERATION CODE TABLE

LSD / MSD	0	1	2	3	4	5	6	7
0	BRK	ORA-IND, X				ORA-Z PAGE	ASL-Z PAGE	
1	BPL	ORA-IND, Y				ORA-Z PAGE, X	ASL-Z PAGE, X	
2	JSR	AND-IND, X			BIT-Z Page	AND-Z PAGE	ROL-Z PAGE	
3	BMI	AND-IND, Y				AND-Z PAGE, X	ROL-Z PAGE, X	
4	RTI	EOR-IND, X				EOR-Z PAGE	LSR-Z PAGE	
5	BVC	EOR-IND, Y				EOR-Z PAGE, X	LSR-Z PAGE, X	
6	RTS	ADC-IND, X				ADC-Z PAGE	ROR-Z PAGE	
7	BVS	ADC-IND, Y				ADC-Z PAGE, X		
8		STA-IND, X			STY-Z Page	STA-Z PAGE	STX-Z PAGE	
9	BCC	STA-IND, Y			STY-Z Page, X	STA-Z PAGE, X	STX-Z PAGE, Y	
A	LDY-IMM	LDA-IND, X	LDX-IMM		LDY-Z Page	LDA-Z PAGE	LDX-Z PAGE	
B	BCS	LDA-IND, Y			LDY-Z Page, X	LDA-Z PAGE, X	LDX-Z PAGE, Y	
C	CPY-IMM	CMP-IND, X			CPY-Z Page	CMP-Z PAGE	DEC-Z PAGE	
D	BNE	CMP-IND, Y				CMP-Z PAGE, X	DEC-Z PAGE, X	
E	CPX-IMM	SBC-IND, X			CPX-Z Page	SBC-Z PAGE	INC-Z PAGE	
F	BEQ	SBC-IND, Y				SBC-Z PAGE, X	INC-Z PAGE, X	

8	9	A	B	C	D	E	F	LSD / MSD
PHP	ORA-IMM	ASL-A			ORA-ABS	ASL-ABS		0
CLC	ORA-ABS, Y				ORA-ABS, X	ASL-ABS, X		1
PLP	AND-IMM	ROL-A		BIT-ABS	AND-ABS	ROL-ABS		2
SEC	AND-ABS, Y				AND-ABS, X	ROL-ABS, X		3
PHA	EOR-IMM	LSR-A		JUMP-ABS	EOR-ABS	LSR-ABS		4
CLI	EOR-ABS, Y				EOR-ABS, X	LSR-ABS, X		5
PLA	ADC-IMM	ROR-A		JMP-IND	ADC-ABS	ROR-ABS		6
SEI	ADC-ABS, Y				ADC-ABS, X			7
DEY		TXA		STY-ABS	STA-ABS	STX-ABS		8
TYA	STA-ABS, Y	TXS			STA-ABS, X			9
TAY	LDA-IMM	TAX		LDY-ABS	LDA-ABS	LDX-ABS		A
CLV	LDA-ABS, Y	TSX		LDY-ABS, X	LDA-ABS, X	LDX-ABS, Y		B
INY	CMP-IMM	DEX		CPY-ABS	CMP-ABS	DEC-ABS		C
CLD	CMP-ABS, Y				CMP-ABS, X	DEC-ABS, X		D
INX	SBC-IMM	NOP		CPX-ABS	SBC-ABS	INC-ABS		E
SED	SBC-ABS, Y				SBC-ABS, X	INC-ABS, X		F

ADDRESSING MODES

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 (EA). 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 the instruction 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 result 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 result being the 8 high order bits of the EA.

FIG. 1 IRQ, NMI, RTI, BRK OPERATION

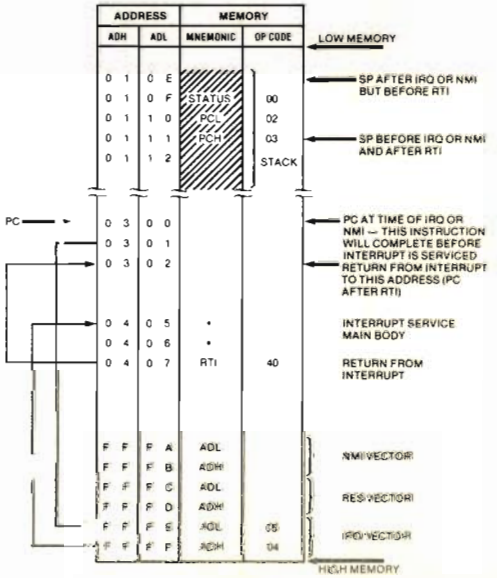
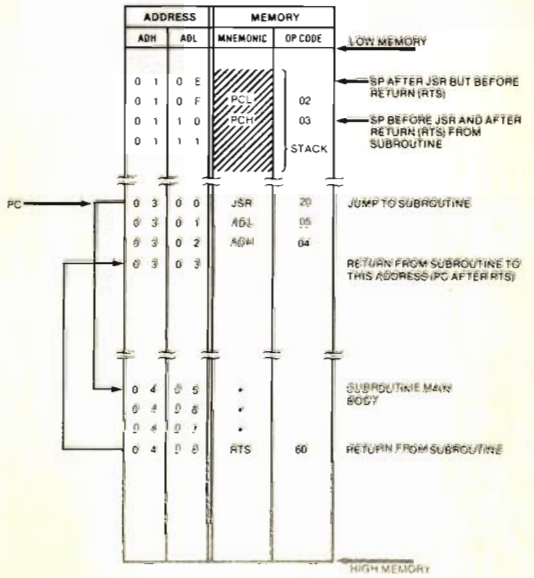
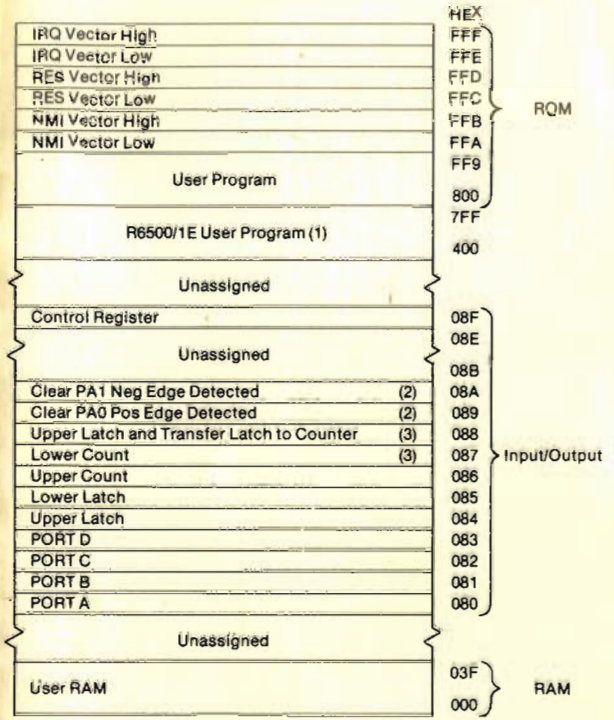


FIG. 2 JSR, RTS OPERATION



R6500/1 ONE-CHIP MICROCOMPUTER

R6500/1 MEMORY MAP



Notes:
(1) Additional 1024 bytes are decoded for external memory addressing by the R6500/1E Emulator Device.
(2) I/O command only; i.e., no stored data.
(3) Clears Counter Overflow — Bit 7 in Control Register.