APPENDIX B

INSTRUCTION LIST

ALPHABETIC BY MNEMONIC

WITH OP CODES, EXECUTION CYCLES

AND MEMORY REQUIREMENTS

The following notation applies to this summary:

A	Accumulator
X, Y	Index Registers
M	Memory
P	Processor Status Register
S	Stack Pointer
✓	Change
—	No Change
+	Add
∧	Logical AND
-	Subtract
∀	Logical Exclusive Or
↑	Transfer from Stack
↓	Transfer to Stack
→	Transfer to
←	Transfer to
V	Logical OR
PC	Program Counter
PCH	Program Counter High
PCL	Program Counter Low
OPER	OPERAND
#	IMMEDIATE ADDRESSING MODE

Note: At the top of each table is located in parentheses a reference number (Ref: XX) which directs the user to that Section in the MCS6500 Microcomputer Family Programming Manual in which the instruction is defined and discussed.

ADC

Add memory to accumulator with carry

ADC

Operation: A + M + C → A, C

N	Z	C	I	D	V
✓	✓	✓	—	—	✓

(Ref: 2.2.1)

Addressing Mode	Assembly Language Form	OP CODE	No. Bytes	No. Cycles
Immediate	ADC # Oper	69	2	2
Zero Page	ADC Oper	65	2	3
Zero Page, X	ADC Oper, X	75	2	4
Absolute	ADC Oper	6D	3	4
Absolute, X	ADC Oper, X	7D	3	4*
Absolute, Y	ADC Oper, Y	79	3	4*
(Indirect, X)	ADC (Oper, X)	61	2	6
(Indirect), Y	ADC (Oper), Y	71	2	5*

* Add 1 if page boundary is crossed.

BIT *Test bits in memory with accumulator*

BIT

Operation: $A \wedge M$, $M_7 \rightarrow N$, $M_6 \rightarrow V$

Bit 6 and 7 are transferred to the status register. N Z C I D V

If the result of $A \wedge M$ is zero then Z = 1, otherwise M_7 ✓ — — — M_6

Z = Ø

(Ref: 4.2.1.1)

Addressing Mode	Assembly Language Form	OP CODE	No. Bytes	No. Cycles
Zero Page	BIT Oper	24	2	3
Absolute	BIT Oper	2C	3	4

SBC

SBC *Subtract memory from accumulator with borrow*

SBC

Operation: $A - M - \overline{C} \rightarrow A$

Note: $\overline{C}$ = Borrow (Ref: 2.2.2)

N	Z	C	I	D	V
✓	✓	✓	—	—	✓

Addressing Mode	Assembly Language Form	OP CODE	No. Bytes	No. Cycles
Immediate	SBC #Oper	E9	2	2
Zero Page	SBC Oper	E5	2	3
Zero Page, X	SBC Oper, X	F5	2	4
Absolute	SBC Oper	ED	3	4
Absolute, X	SBC Oper, X	FD	3	4*
Absolute, Y	SBC Oper, Y	F9	3	4*
(Indirect, X)	SBC (Oper, X)	E1	2	6
(Indirect), Y	SBC (Oper), Y	F1	2	5*

* Add 1 when page boundary is crossed.

APPENDIX C

INSTRUCTION ADDRESSING

MODES AND

RELATED EXECUTION TIMES

INSTRUCTION ADDRESSING MODES AND RELATED EXECUTION TIMES (in clock cycles)

	Accumulator	Immediate	Zero Page	Zero Page, X	Zero Page, Y	Absolute	Absolute, X	Absolute, Y	Implied	Relative	(Indirect, X)	(Indirect), Y	Absolute Indirect
ADC	.	2	3	4	.	4	4*	4*	.	.	6	5*	.
AND	.	2	3	4	.	4	4*	4*	.	.	6	5*	.
ASL	2	.	5	6	.	6	7	.	.	.	.	.	.
BCC	.	.	.	.	.	.	.	.	.	2**	.	.	.
BCS	.	.	.	.	.	.	.	.	.	2**	.	.	.
BEQ	.	.	.	.	.	.	.	.	.	2**	.	.	.
BIT	.	.	3	.	.	4	.	.	.	.	.	.	.
BMI	.	.	.	.	.	.	.	.	.	2**	.	.	.
BNE	.	.	.	.	.	.	.	.	.	2**	.	.	.
BPL	.	.	.	.	.	.	.	.	.	2**	.	.	.
BRK	.	.	.	.	.	.	.	.	.	.	.	.	.
BVC	.	.	.	.	.	.	.	.	.	2**	.	.	.
BVS	.	.	.	.	.	.	.	.	.	2**	.	.	.
CLC	.	.	.	.	.	.	.	.	2	.	.	.	.
CLD	.	.	.	.	.	.	.	.	2	.	.	.	.
CLI	.	.	.	.	.	.	.	.	2	.	.	.	.
CLV	.	.	.	.	.	.	.	.	2	.	.	.	.
CMP	.	2	3	4	.	4	4*	4*	.	.	6	5*	.
CPX	.	2	3	.	.	4	.	.	.	.	.	.	.
CPY	.	2	3	.	.	4	.	.	.	.	.	.	.
DEC	.	.	5	6	.	6	7	.	.	.	.	.	.
DEX	.	.	.	.	.	.	.	.	2	.	.	.	.
DEY	.	.	.	.	.	.	.	.	2	.	.	.	.
EOR	.	2	3	4	.	4	4*	4*	.	.	6	5	.
INC	.	.	5	6	.	6	7	.	.	.	.	.	.
INX	.	.	.	.	.	.	.	.	2	.	.	.	.
INY	.	.	.	.	.	.	.	.	2	.	.	.	.
JMP	.	.	.	.	.	3	.	.	.	.	.	.	5

	Accumulator	Immediate	Zero Page	Zero Page, X	Zero Page, Y	Absolute	Absolute, X	Absolute, Y	Implied	Relative	(Indirect, X)	(Indirect), Y	Absolute Indirect
JSR	.	.	.	.	.	6	.	.	.	.	.	.	.
LDA	.	2	3	4	.	4	4*	4*	.	.	6	5*	.
LDX	.	2	3	.	4	4	.	4*	.	.	.	.	.
LDY	.	2	3	4	.	4	4*	.	.	.	.	.	.
LSR	2	.	5	6	.	6	7	.	.	.	.	.	.
NOP	.	.	.	.	.	.	.	.	2	.	.	.	.
ORA	.	2	3	4	.	4	4*	4*	.	.	6	5*	.
PHA	.	.	.	.	.	.	.	.	3	.	.	.	.
PHP	.	.	.	.	.	.	.	.	3	.	.	.	.
PLA	.	.	.	.	.	.	.	.	4	.	.	.	.
PLP	.	.	.	.	.	.	.	.	4	.	.	.	.
ROL	2	.	5	6	.	6	7	.	.	.	.	.	.
ROR	2	.	5	6	.	6	7	.	.	.	.	.	.
RTI	.	.	.	.	.	.	.	.	6	.	.	.	.
RTS	.	.	.	.	.	.	.	.	6	.	.	.	.
SBC	.	2	3	4	.	4	4*	4*	.	.	6	5*	.
SEC	.	.	.	.	.	.	.	.	2	.	.	.	.
SED	.	.	.	.	.	.	.	.	2	.	.	.	.
SEI	.	.	.	.	.	.	.	.	2	.	.	.	.
STA	.	.	3	4	.	4	5	5	.	.	6	6	.
STX*	.	.	3	.	4	4	.	.	.	.	.	.	.
STY**	.	.	3	4	.	4	.	.	.	.	.	.	.
TAX	.	.	.	.	.	.	.	.	2	.	.	.	.
TAY	.	.	.	.	.	.	.	.	2	.	.	.	.
TSX	.	.	.	.	.	.	.	.	2	.	.	.	.
TXA	.	.	.	.	.	.	.	.	2	.	.	.	.
TXS	.	.	.	.	.	.	.	.	2	.	.	.	.
TYA	.	.	.	.	.	.	.	.	2	.	.	.	.

* Add one cycle if indexing across page boundary
** Add one cycle if branch is taken, Add one additional if branching operation crosses page boundary

APPENDIX E

SUMMARY OF ADDRESSING MODES

This appendix is to serve the user in providing a reference for the MCS650X addressing modes. Each mode of address is shown with a symbolic illustration of the bus status at each cycle during the instruction fetch and execution. The example number as found in the text is provided for reference purposes.

E.1 IMPLIED ADDRESSING

Example 5.3: Illustration of implied addressing

Clock Cycle	Address Bus	Program Counter	Data Bus	Comments
1	PC	PC + 1	OP CODE	Fetch OP CODE
2	PC + 1	PC + 1	New OP CODE	Ignore New OP CODE; Decode Old OP CODE
3	PC + 1	PC + 2	New OP CODE	Fetch New OP CODE; Execute Old OP CODE

E.2 IMMEDIATE ADDRESSING

Example 5.4: Illustration of immediate addressing

Clock Cycle	Address Bus	Program Counter	Data Bus	Comments
1	PC	PC + 1	OP CODE	Fetch OP CODE
2	PC + 1	PC + 2	Data	Fetch Data, Decode OP CODE
3	PC + 2	PC + 3	New OP CODE	Fetch New OP CODE, Execute Old OP CODE

E.3 ABSOLUTE ADDRESSING

Example 5.5: Illustration of absolute addressing

Clock Cycle	Address Bus	Program Counter	Data Bus	Comments
1	PC	PC + 1	OP CODE	Fetch OP CODE
2	PC + 1	PC + 2	ADL	Fetch ADL, Decode OP CODE
3	PC + 2	PC + 3	ADH	Fetch ADH, Retail ADL
4	ADH, ADL	PC + 3	Data	Fetch Data
5	PC + 3	PC + 4	New OP CODE	Fetch New OP CODE, Execute Old OP CODE

E.4 ZERO PAGE ADDRESSING

Example 5.6: Illustration of zero page addressing

Clock Cycle	Address Bus	Program Counter	Data Bus	Comments
1	PC	PC + 1	OP CODE	Fetch OP CODE
2	PC + 1	PC + 2	ADL	Fetch ADL, Decode OP CODE
3	00, ADL	PC + 2	Data	Fetch Data
4	PC + 2	PC + 3	New OP CODE	Fetch New OP CODE, Execute Old OP CODE

E.5 RELATIVE ADDRESSING – (Branch Positive, no crossing of page boundaries)

Example 5.8: Illustration of relative addressing--branch positive taken, no crossing of page boundaries

Cycle	Address Bus	Data Bus	External Operation	Internal Operation
1	0100	OP CODE	Fetch OP CODE	Finish Previous Operation, Increment Program Counter to 101
2	0101	+50	Fetch Offset	Interpret Instruction, Increment Program Counter to 102
3	0102	Next OP CODE	Fetch Next OP CODE	Check Flags, Add Relative to PCL, Increment Program Counter to 103
4	0152	Next OP CODE	Fetch Next OP CODE	Transfer Results to PCL, Increment Program Counter to 153

E.6 ABSOLUTE INDEXED ADDRESSING – (with page crossing)

Step 5 is deleted and the data in step 4 is valid when no page crossing occurs.

Example 6.7: Absolute Indexed; With Page Crossing

Cycle	Address Bus	Data Bus	External Operation	Internal Operation
1	0100	OP CODE	Fetch OP CODE	Finish Previous Operation Increment PC to 101
2	0101	BAL	Fetch BAL	Interpret Instruction Increment PC to 102
3	0102	BAH	Fetch BAH	Add BAL + Index Increment PC to 103
4	BAH,BAL +X	Data (Ignore)	Fetch Data (Data is ignored)	Add BAH + Carry
5	BAH+1, BAL+X	Data	Fetch Data	
6	0103	Next OP CODE	Fetch Next OP CODE	Finish Operation

Example 6.8: Illustration of Zero Page Indexing

Cycle	Address Bus	Data Bus	External Operation	Internal Operation
1	0100	OP CODE	Fetch OP CODE	Finish Previous Operation
2	0101	BAL	Fetch Base Address Low (BAL)	Interpret Instruction
3	00,BAL	Data (Discarded	Fetch Discarded Data	Add: BAL + X
4	00,BAL +X	Data	Fetch Data	
5	0102	Next OP CODE	Fetch Next OP CODE	Finish Operation

Example 6.10: Illustration of Indexed Indirect Addressing

Cycle	Address Bus	Data Bus	External Operation	Internal Operation
1	0100	OP CODE	Fetch OP CODE	Finish Previous Operation
2	0101	BAL	Fetch BAL	Interpret Instruction
3	00,BAL	DATA (Discarded)	Fetch Discarded DATA	Add BAL + X
4	00,BAL + X	ADL	Fetch ADL	Add 1 to BAL + X
5	00,BAL + X + 1		Fetch ADH	Hold ADL
6	ADH,ADL	DATA	Fetch DATA	
7	0102	Next OP	Fetch Next OP CODE	Finish Operation

E.9 INDIRECT INDEXED ADDRESSING (with page crossing)

Step 6 is deleted and the data in step 5 is valid when no page crossing occurs.

Example 6.12: Indirect Indexed Addressing (With Page Crossing)

Cycle	Address Bus	Data Bus	External Operation	Internal Operation
1	0100	OP CODE	Load OP CODE	Finish Previous Operation
2	0101	IAL	Fetch IAL	Interpret Instruction
3	00,IAL	BAL	Fetch BAL	Add 1 to IAL
4	00,IAL + 1	BAH	Fetch BAH	Add BAL to Y
5	BAH,BAL + Y	DATA (Discarded)	Fetch DATA (Discarded)	Add 1 to BAH
6	BAH + 1 BAL + Y	DATA	Fetch Data	
7	0102	Next OP CODE	Fetch Next OP CODE	Finish This Operation

APPENDIX H

REVIEW OF BINARY

AND

BINARY CODED DECIMAL

ARITHMETIC

The microprocessor automatically takes this into account and corrects for the fact that

Decimal		BCD	Hex		
79	=	01111001	79	=	01111001
+12	=	00010010	12	=	00010010
91	=	10010001	88	=	10001011

The only difference between Hex and BCD representation is that the microprocessor automatically adjusts for the fact that BCD does not allow for Hex values A - F during add and subtract operations.

The offset which follows a branch instruction is in signed two's complement form which means that

	$+50	= +80	= 01010000
and	$-50	= -80	= 10110000
		Proof	= 00000000

The sign for this operation is in bit 7 where an 0 equals positive and a 1 equals negative.

This bit is correct for the two's complement representation but also flags the microprocessor whether to carry or borrow from the address high byte.

The following 4 examples represent the combinations of offsets which might occur (all notations are in hexadecimal):

Example H.4.1: Forward reference, no page crossing

0105	BNE
0106	+55
0107	Next OP CODE

To calculate next instruction if the branch is taken

Offset	+55	01010101
Address Low for next OP CODE	07	00000111
	5C	01011100

with no carry, giving 015C as the result.

Example H.4.2: Backward reference, no page crossing

```
015A    BNE
015B    -55
015C    Next OP CODE
```

To calculate if branch is taken,

```
  Offset                -55 = AB = 10101011
+ Address Low for
    Next OP CODE        +5C = 5C = 01011100
                         07   07   00000111
```

The carry is expected because of the negative offset and is ignored, thus giving 0107 as the result.

Example H.4.3: Backward reference if page boundary crossed

```
0105    BNE
0106    -55
0107    Next OP CODE
```

To calculate if branch is taken, first calculate a low byte

```
  Offset                -55 = AB = 10101011
  Address Low for
    Next OP CODE         07 = 07 = 00000111
                         B2 = B2 = 10110010
```

There is no carry from a negative offset; therefore, a carry must be made:

```
                    -1 =  -1 = FF = 11111111
+ Address High       =  01 = 01 = 00000001
                        00   00   00000000
```

This gives 00 B2 as a result.

Example H.4.4: Forward reference across page boundary

```
00B0    BNE
00B1    +55
00B2    Next OP CODE
```

To calculate next instruction if branch is taken,

Offset	55 =	01010101
Address Low for Next OP CODE	B2 =	10110010
	07 =	00000111

with carry on positive number.

+1	1 =	00000001
Address High	00 =	00000000
	1 =	00000001

which gives 0107.