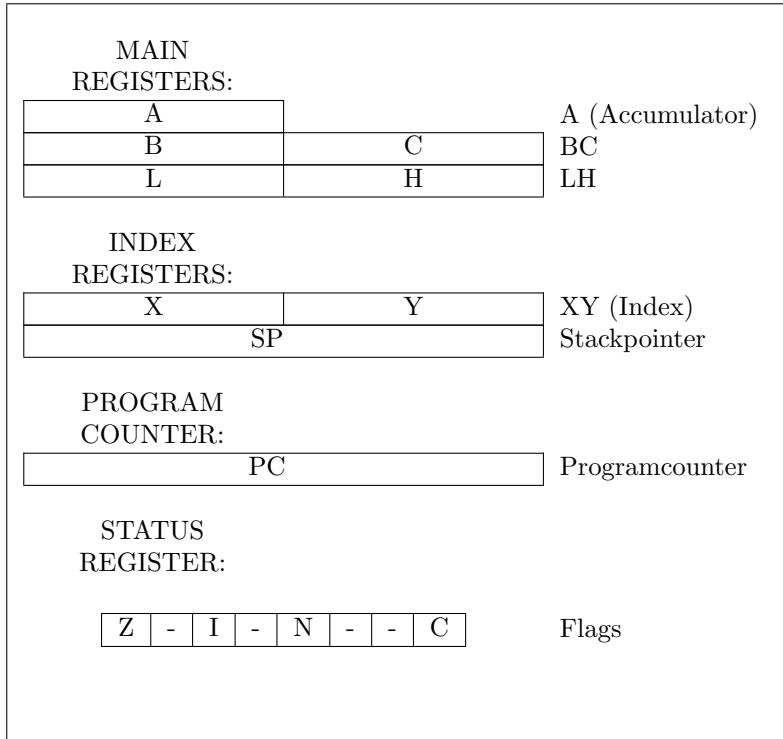


Chimera-2016-I

Assembly Language Programming

CANS TECHNOLOGIES, INC

Processor Architecture



IMMEDIATE ADDRESSING (#)

The operand is the second byte for 8 bit instructions or the second byte for the lower byte and third byte for the higher byte represent the data for given instruction, no memory addressing is required.

IMPLIED ADDRESSING(impl)

A single byte instruction in which all of the data and operands are implied through the instruction itself.

ABSOLUTE ADDRESSING(abs)

In absolute addressing the second byte of an instruction represents the low order byte of an effective address. The third byte represents the high order byte of an effective address. The two bytes are added to allow full access to 65K of memory.

INDEXED ABSOLUTE ADDRESSING(abs,X)

In indexed absolute addressing the second byte and third byte of an instruction are used in conjunction with a index register (Register X or Register Y or Register XY). the second byte of the instruction represents the low order byte of an effective address. The third byte represents the high high byte of an effective address. The result is added to the index register giving a result anywhere in memory. Any 16 bit carry is discarded.

ZERO PAGE ADDRESSING(zpg)

In zero page addressing the second byte of a instruction represents the low order byte of an effective address. The high order byte is fixed at 0 giving you access to the first 256 memory locations.

OFFSET ADDRESSING(rel)

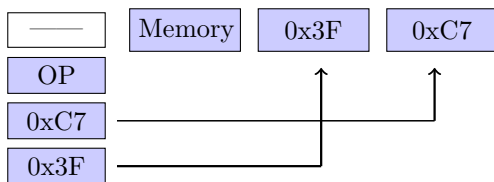
In offset addressing, the second instruction byte of the instruction is used as a offset in conjunction with the programcounter. The offset is calculated by using the given byte as signed, resulting in -128 to +127. This offset is added to the contents of the programcounter giving the effective address within -128 to +127.

REGISTER ADDRESSING

In Register addressing the name of the desintation register (and the source where applicable) is stated in the instruction needing no addition bytes.

Little Endian: Any instruction that contains 2 addition byte are arranged in the order of low first then high.

Below is a example of a opcode using absolute addressing.



Hexadecimal Matrix

		LOW NIBBLE															
		0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7	0x8	0x9	0xA	0xB	0xC	0xD	0xE	0xF
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
HIGH NIBBLE	0x0	-	-	SWI	RTI	STO	STOX	STOY	JMPR	CCC	CCS	CNE	CEQ	CMI	CPL	-	STOS
	-	-	-	impl	impl	abs	abs	abs	abs	abs	abs	abs	abs	abs	abs	-	abs
	0x1	-	-	-	-	STO	STOX	STOY	NOP	WAI	-	-	-	ADI	CPI	ANI	STOS
	-	-	-	-	-	abs,X	abs,X	abs,X	impl	impl	-	-	-	#	#	#	abs,X
	0x2	LODS	LDX	LODY	RT	STO	STOX	STOY	MVR	MVR	MVR	MVR	TAY	TYA	MSA	-	STOS
	-	#	#	#	impl	abs,Y	abs,Y	abs,Y	B , #	C , #	L , #	H , #	impl	impl	impl	-	abs,Y
	0x3	LODS	LDX	LODY	-	STO	STOX	STOY	-	JUMP	JCC	JCS	JNE	JEQ	JMI	JPL	STOS
	-	abs	abs	abs	-	abs,X	abs,X	abs,X	-	abs	abs	abs	abs	abs	abs	abs	abs,X
	0x4	LODS	LDX	LODY	LD	STO	STOX	STOY	-	-	-	DEX	INX	DEY	INY	-	STOS
	-	abs,X	abs,X	abs,X	#	zpg	zpg	zpg	-	-	-	impl	impl	impl	impl	-	zpg
	0x5	LODS	LDX	LODY	LD	TEST	INC	DEC	RR	RCL	SAL	SHR	COM	NEG	RAL	ROR	CLR
	-	abs,Y	abs,Y	abs,Y	abs	abs	abs	abs	abs	abs	abs	abs	abs	abs	abs	abs	abs
	0x6	LODS	LDX	LODY	LD	TEST	INC	DEC	RR	RCL	SAL	SHR	COM	NEG	RAL	ROR	CLR
	-	abs,X	abs,X	abs,X	abs,X	abs,X	abs,X	abs,X	abs,X	abs,X	abs,X	abs,X	abs,X	abs,X	abs,X	abs,X	abs,X
	0x7	LODS	LDX	LODY	LD	TEST	INC	DEC	RR	RCL	SAL	SHR	COM	NEG	RAL	ROR	CLR
	-	zpg	zpg	zpg	abs,Y	abs,Y	abs,Y	abs,Y	abs,Y	abs,Y	abs,Y	abs,Y	abs,Y	abs,Y	abs,Y	abs,Y	abs,Y
	0x8	-	-	-	LD	TEST	INC	DEC	RR	RCL	SAL	SHR	COM	NEG	RAL	ROR	CLR
	-	-	-	-	abs,X	abs,X	abs,X	abs,X	abs,X	abs,X	abs,X	abs,X	abs,X	abs,X	abs,X	abs,X	abs,X
	0x9	-	-	-	LD	TESTA	INCA	DECA	RRR	RCLA	SALA	SHRA	COMA	NEGA	RALA	RORA	CLRA
	-	-	-	-	zpg	A	A	A	A	A	A	A	A	A	A	A	A
	0xa	MV	MV	MV	MV	MV	MV	CLC	SEC	CLI	SEI	CMC	-	-	-	PUSH	POP
	-	A , A	B , A	C , A	L , A	H , A	M , A	impl	impl	impl	impl	impl	-	-	-	A	A
	0xb	MV	MV	MV	MV	MV	MV	ADC	SBC	ADD	SUB	CMP	OR	AND	XOR	PUSH	POP
	-	A , B	B , B	C , B	L , B	H , B	M , B	A , B	A , B	A , B	A , B	A , B	A , B	A , B	A , B	FL	FL
	0xc	MV	MV	MV	MV	MV	MV	ADC	SBC	ADD	SUB	CMP	OR	AND	XOR	PUSH	POP
	-	A , C	B , C	C , C	L , C	H , C	M , C	A , C	A , C	A , C	A , C	A , C	A , C	A , C	A , C	B	B
	0xd	MV	MV	MV	MV	MV	MV	ADC	SBC	ADD	SUB	CMP	OR	AND	XOR	PUSH	POP
	-	A , L	B , L	C , L	L , L	H , L	M , L	A , L	A , L	A , L	A , L	A , L	A , L	A , L	A , L	C	C
	0xe	MV	MV	MV	MV	MV	MV	ADC	SBC	ADD	SUB	CMP	OR	AND	XOR	PUSH	POP
	-	A , H	B , H	C , H	L , H	H , H	M , H	A , H	A , H	A , H	A , H	A , H	A , H	A , H	A , H	L	L
	0xf	MV	MV	MV	MV	MV	MV	ADC	SBC	ADD	SUB	CMP	OR	AND	XOR	PUSH	POP
	-	A , M	B , M	C , M	L , M	H , M	- , -	A , M	A , M	A , M	A , M	A , M	A , M	A , M	A , M	H	H

		OPCODE		Addressing Opcode			
		DESCRIPTION					
		Flags:	Z - I - N - - C				
		NOTES					
LD		Addressing	Opcode	STO		Addressing	Opcode
Loads Memory into Accumulator		#	0x43	Stores Accumulator into Memory		abs	0x04
Flags:	T - - - T - - -	abs,X	0x53			abs,X	0x14
notes		abs,Y	0x73	Flags:	T - - - T - - -	abs,Y	0x24
		abs,XY	0x83	notes		abs,XY	0x34
		zpg	0x93			zpg	0x44
ADC		Addressing	Opcode	SBC		Addressing	Opcode
Register added to Accumulator with Carry		A-B	0xB6	Register subtracted to Accumulator with Carry		A-B	0xB7
		A-C	0xC6			A-C	0xC7
		A-L	0xD6			A-L	0xD7
Flags:	T - - - T - - T	A-H	0xE6	Flags:	T - - - T - - T	A-H	0xE7
notes		A-M	0xF6	notes		A-M	0xF7
ADD		Addressing	Opcode	SUB		Addressing	Opcode
Register added to Accumulator		A-B	0xB8	Register subtracted to Accumulator		A-B	0xB9
		A-C	0xC8			A-C	0xC9
Flags:	T - - - T - - T	A-L	0xD8	Flags:	T - - - T - - T	A-L	0xD9
notes		A-H	0xE8	notes		A-H	0xE9
		A-M	0xF8			A-M	0xF9
CMP		Addressing	Opcode	OR		Addressing	Opcode
Register compared to Accumulator		A-B	0xBA	Register bitwise inclusive or with Accumulator		A-B	0xBB
		A-C	0xCA			A-C	0xCB
Flags:	T - - - T - - T	A-L	0xDA	Flags:		A-L	0xDB
notes		A-H	0xEA	notes		A-H	0xEB
		A-M	0xFA			A-M	0xFB
AND		Addressing	Opcode	XOR		Addressing	Opcode
Register bitwise and with Accumulator		A-B	0xBC	Register bitwise exclusive or with Accumulator		A-B	0xBD
		A-C	0xCC			A-C	0xCD
Flags:	T - - - T - - -	A-L	0xDC	Flags:	T - - - T - - -	A-L	0xDD
notes		A-H	0xEC	notes		A-H	0xED
		A-M	0xFC			A-M	0xFD
ADI		Addressing	Opcode	CPI		Addressing	Opcode
Data added to Accumulator with Carry		#	0x1C	Data compared to Accumulator		#	0x1D
				Flags:			
Flags:	T - - - T - - T			notes			
notes							
ANI		Addressing	Opcode	TEST		Addressing	Opcode
Data bitwise and with Accumulator		#	0x1E	Bit test Memory or Accumulator		abs	0x54
				Flags:		abs,X	0x64
Flags:	T - - - T - - -			notes		abs,Y	0x74
notes						abs,XY	0x84

TESTA		Addressing	Opcode
Bit test Memory or Accumulator		A	0x94
Flags:	T - - - T - - -		
notes			

INC		Addressing	Opcode
Increment Memory or Accumulator		abs	0x55
		abs,X	0x65
Flags:	T - - - T - - -	abs,Y	0x75
notes		abs,XY	0x85

INCA		Addressing	Opcode
Increment Memory or Accumulator		A	0x95
Flags:	T - - - T - - -		
notes			

DEC		Addressing	Opcode
Decrement Memory or Accumulator		abs	0x56
		abs,X	0x66
Flags:	T - - - T - - -	abs,Y	0x76
notes		abs,XY	0x86

DECA		Addressing	Opcode
Decrement Memory or Accumulator		A	0x96
Flags:	T - - - T - - -		
notes			

RR		Addressing	Opcode
Rotate right through carry Memory or Accumulator		abs	0x57
		abs,X	0x67
		abs,Y	0x77
Flags:	T - - - T - - T	abs,XY	0x87
notes			

RRA		Addressing	Opcode
Rotate right through carry Memory or Accumulator		A	0x97
Flags:	T - - - T - - T		
notes			

RCL		Addressing	Opcode
Rotate left through carry Memory or Accumulator		abs	0x58
		abs,X	0x68
		abs,Y	0x78
Flags:	T - - - T - - T	abs,XY	0x88
notes			

RCLA		Addressing	Opcode
Rotate left through carry Memory or Accumulator		A	0x98
Flags:	T - - - T - - T		
notes			

SAL		Addressing	Opcode
Arithmetic shift left Memory or Accumulator		abs	0x59
		abs,X	0x69
		abs,Y	0x79
Flags:	T - - - T - - T	abs,XY	0x89
notes			

SALA		Addressing	Opcode
Arithmetic shift left Memory or Accumulator		A	0x99
Flags:	T - - - T - - T		
notes			

SHR		Addressing	Opcode
Arithmetic shift right Memory or Accumulator		abs	0x5A
		abs,X	0x6A
		abs,Y	0x7A
Flags:	T - - - T - - T	abs,XY	0x8A
notes			

SHRA		Addressing	Opcode
Arithmetic shift right Memory or Accumulator		A	0x9A
Flags:	T - - - T - - T		
notes			

COM		Addressing	Opcode
Negate Memory or Accumulator		abs	0x5B
		abs,X	0x6B
Flags:	T - - - T - - T	abs,Y	0x7B
notes		abs,XY	0x8B

COMA		Addressing	Opcode
Negate Memory or Accumulator		A	0x9B
Flags:	T - - - T - - T		
notes			

NEG		Addressing	Opcode
2's complement Memory or Accumulator		abs	0x5C
		abs,X	0x6C
		abs,Y	0x7C
Flags:	T - - - T - - -	abs,XY	0x8C
notes			

NEGA		Addressing	Opcode
2's complement Memory or Accumulator		A	0x9C
Flags:	T - - - T - - -		
notes			

RAL		Addressing	Opcode
Rotate left without carry Memory or Accumulator		abs	0x5D
		abs,X	0x6D
		abs,Y	0x7D
Flags:	T - - - T - - -	abs,XY	0x8D
notes			

RALA		Addressing	Opcode
Rotate left without carry Memory or Accumulator		A	0x9D
Flags:	T - - - T - - -		
notes			

ROR		Addressing	Opcode
Rotate right without carry Memory or Accumulator		abs	0x5E
		abs,X	0x6E
		abs,Y	0x7E
Flags:	T - - - T - - -	abs,XY	0x8E
notes			

RORA		Addressing	Opcode
Rotate right without carry Memory or Accumulator		A	0x9E
Flags:	T - - - T - - -		
notes			

CLR		Addressing	Opcode
Clear Memory or Accumulator		abs	0x5F
		abs,X	0x6F
Flags:	1 - - - 0 - - 0	abs,Y	0x7F
notes		abs,XY	0x8F

CLRA		Addressing	Opcode
Clear Memory or Accumulator		A	0x9F
Flags:	1 - - - 0 - - 0		
notes			

MV		Addressing	Opcode
Transfer from one register to another		A-A	0xA0
		A-B	0xB0
Flags:	- - - - - - - -	A-C	0xC0
notes		A-L	0xD0
		A-H	0xE0
		A-M	0xF0
		B-A	0xA1
		B-B	0xB1
		B-C	0xC1
		B-L	0xD1
		B-H	0xE1
		B-M	0xF1
		C-A	0xA2
		C-B	0xB2
		C-C	0xC2
		C-L	0xD2
		C-H	0xE2
		C-M	0xF2
		L-A	0xA3
		L-B	0xB3
		L-C	0xC3
		L-L	0xD3
		L-H	0xE3
		L-M	0xF3
		H-A	0xA4
		H-B	0xB4
		H-C	0xC4
		H-L	0xD4
		H-H	0xE4
		H-M	0xF4
		M-A	0xA5
		M-B	0xB5
		M-C	0xC5
		M-L	0xD5
		M-H	0xE5
		—	0xF5

LDX		Addressing	Opcode
Loads Memory into register X		#	0x21
		abs	0x31
Flags:	T - - - T - - -	abs,X	0x41
notes		abs,Y	0x51
		abs,XY	0x61
		zpg	0x71

STOX		Addressing	Opcode
Stores register X into Memory		abs	0x05
		abs,X	0x15
Flags:	T - - - T - - -	abs,Y	0x25
notes		abs,XY	0x35
		zpg	0x45

DEX		Addressing	Opcode
Decrements register X		impl	0x4A
Flags:	T - - - - - - -		
notes			

INX		Addressing	Opcode
Increments register X		impl	0x4B
Flags:	T - - - - - - -		
notes			

LODY		Addressing	Opcode
Loads Memory into register Y		#	0x22
		abs	0x32
Flags:	T - - - T - - -	abs,X	0x42
notes		abs,Y	0x52
		abs,XY	0x62
		zpg	0x72

STOY		Addressing	Opcode
Stores register Y into Memory		abs	0x06
		abs,X	0x16
Flags:	T - - - T - - -	abs,Y	0x26
notes		abs,XY	0x36
		zpg	0x46

TAY		Addressing	Opcode
Transters Accumulator to register Y		impl	0x2B
Flags:	---- T ---		
notes			

TYA		Addressing	Opcode
Transters register Y to Accumulator		impl	0x2C
Flags:	T --- T ---		
notes			

DEY		Addressing	Opcode
Decrements register Y		impl	0x4C
Flags:	T - - - - -		
notes			

INY		Addressing	Opcode
Increments register Y		impl	0x4D
Flags:	T - - - - -		
notes			

LODS		Addressing	Opcode
Loads Memory into Stackpointer		#	0x20
		abs	0x30
Flags:	T - - - T - - -	abs,X	0x40
notes		abs,Y	0x50
		abs,XY	0x60
		zpg	0x70

STOS		Addressing	Opcode
Stores Stackpointer into Memory		abs	0x0F
		abs,X	0x1F
Flags:	T - - - T - - -	abs,Y	0x2F
notes		abs,XY	0x3F
		zpg	0x4F

MSA		Addressing	Opcode
Transters Status register to Accumulator		impl	0x2D
Flags:	- - - - -		
notes			

PUSH		Addressing	Opcode
Pushes Register onto the Stack		A	0xAE
		FL	0xBE
Flags:	- - - - -	B	0xCE
notes		C	0xDE
		L	0xEE
		H	0xFE

POP		Addressing	Opcode
Pop the top of the Stack into the Register		A	0xAF
		FL	0xBF
Flags:	- - - - -	B	0xCF
notes		C	0xDF
		L	0xEF
		H	0xFF

JUMP		Addressing	Opcode
Loads Memory into ProgramCounter		abs	0x38
Flags:	- - - - -		
notes			

MVR		Addressing	Opcode
Loads Memory into register		B,#	0x27
		C,#	0x28
Flags:	T - - - T - - -	L,#	0x29
notes		H,#	0x2A

JMPR		Addressing	Opcode
Jump to subroutine		abs	0x07
Flags:	- - - - -		
notes			

RT		Addressing	Opcode
Return from subroutine		impl	0x23
Flags:	- - - - -		
notes			

JCC		Addressing	Opcode
Jump on Carry clear		abs	0x39
Flags:	- - - - -		
notes			

JCS		Addressing	Opcode
Jump on Carry set		abs	0x3A
Flags:	- - - - -		
notes			

JNE		Addressing	Opcode
Jump on result not Zero		abs	0x3B
Flags:	- - - - -		
notes			

JEQ		Addressing	Opcode
Jump on result equal to Zero		abs	0x3C
Flags:	- - - - -		
notes			

JMI		Addressing	Opcode
Jump on negative result		abs	0x3D
Flags:	- - - - -		
notes			

JPL		Addressing	Opcode
Jump on positive result		abs	0x3E
Flags:	-----		
notes			

CCC		Addressing	Opcode
Call on Carry clear		abs	0x08
Flags:	-----		
notes			

CCS		Addressing	Opcode
Call on Carry set		abs	0x09
Flags:	-----		
notes			

CNE		Addressing	Opcode
Call on result not Zero		abs	0x0A
Flags:	-----		
notes			

CEQ		Addressing	Opcode
Call on result equal to Zero		abs	0x0B
Flags:	-----		
notes			

CMI		Addressing	Opcode
Call on negative result		abs	0x0C
Flags:	-----		
notes			

CPL		Addressing	Opcode
Call on positive result		abs	0x0D
Flags:	-----		
notes			

CLC		Addressing	Opcode
Clear Carry flag		impl	0xA6
Flags:	----- 0		
notes			

SEC		Addressing	Opcode
Set Carry flag		impl	0xA7
Flags:	----- 1		
notes			

CLI		Addressing	Opcode
Clear Interrupt flag		impl	0xA8
Flags:	-- 0 -----		
notes			

SEI		Addressing	Opcode
Set Interrupt flag		impl	0xA9
Flags:	-- 1 -----		
notes			

CMC		Addressing	Opcode
Compliment carry flag		impl	0xAA
Flags:	-----		
notes			

NOP		Addressing	Opcode
No operation		impl	0x17
Flags:	-----		
notes			

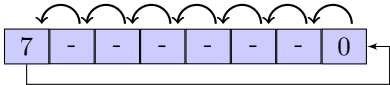
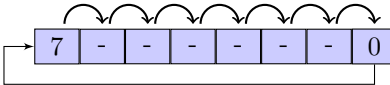
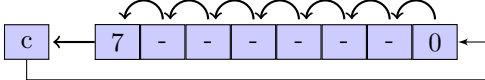
WAI		Addressing	Opcode
Wait for interrupt		impl	0x18
Flags:	-----		
notes			

SWI		Addressing	Opcode
Software interrupt		impl	0x02
Flags:	-- 1 -----		
Pushes: Accumulator Status register General purpose registers (in order)			

RTI		Addressing	Opcode
Return from software interrupt		impl	0x03
Flags:	-----		
Pops: General purpose registers (in order) Status register Accumulator			

			Registers Source								
op	details	Dest	A	FL	B	C	L	H	M	Stack	FLags
ADC	A + CF + R	A	-	-	0xB6	0xC6	0xD6	0xE6	0xF6	-	T - - - T - - T
SBC	A - CF - R	A	-	-	0xB7	0xC7	0xD7	0xE7	0xF7	-	T - - - T - - T
ADD	A + R	A	-	-	0xB8	0xC8	0xD8	0xE8	0xF8	-	T - - - T - - T
SUB	A - R	A	-	-	0xB9	0xC9	0xD9	0xE9	0xF9	-	T - - - T - - T
CMP	A - R		-	-	0xBA	0xCA	0xDA	0xEA	0xFA	-	T - - - T - - T
OR	A R	A	-	-	0xBB	0xCB	0xDB	0xEB	0xFB	-	T - - - T - - -
AND	A & R	A	-	-	0xBC	0xCC	0xDC	0xEC	0xFC	-	T - - - T - - -
XOR	A (+) R	A	-	-	0xBD	0xCD	0xDD	0xED	0xFD	-	T - - - T - - -
TESTA	A - 0	A	0x94	-	-	-	-	-	-	-	T - - - T - - -
INCA	A + 1	A	0x95	-	-	-	-	-	-	-	T - - - T - - -
DECA	A - 1	A	0x96	-	-	-	-	-	-	-	T - - - T - - -
RRA	fig 7	A	0x97	-	-	-	-	-	-	-	T - - - T - - T
RCLA	fig 6	A	0x98	-	-	-	-	-	-	-	T - - - T - - T
SALA	fig 1	A	0x99	-	-	-	-	-	-	-	T - - - T - - T
SHRA	fig 2	A	0x9A	-	-	-	-	-	-	-	T - - - T - - T
COMA	A ~	A	0x9B	-	-	-	-	-	-	-	T - - - T - - T
NEGA	0 - A	A	0x9C	-	-	-	-	-	-	-	T - - - T - - -
RALA	fig 5	A	0x9D	-	-	-	-	-	-	-	T - - - T - - -
RORA	fig 4	A	0x9E	-	-	-	-	-	-	-	T - - - T - - -
CLRA	0	A	-	-	-	-	-	-	-	-	1 - - - 0 - - 0
MV	A	A	0xA0	-	0xB0	0xC0	0xD0	0xE0	0xF0	-	
		B	0xA1	-	0xB1	0xC1	0xD1	0xE1	0xF1	-	
		C	0xA2	-	0xB2	0xC2	0xD2	0xE2	0xF2	-	
		L	0xA3	-	0xB3	0xC3	0xD3	0xE3	0xF3	-	
		H	0xA4	-	0xB4	0xC4	0xD4	0xE4	0xF4	-	
		M	0xA5	-	0xB5	0xC5	0xD5	0xE5	-	-	
			-	-	-	-	-	-	-	-	- - - - - - - -
PUSH	A -*		0xAE	0xBE	0xCE	0xDE	0xEE	0xFE	-	-	- - - - - - - -
POP	+*	A	-	-	-	-	-	-	-	0xAF	
		FL	-	-	-	-	-	-	-	0xBF	
		B	-	-	-	-	-	-	-	0xCF	
		C	-	-	-	-	-	-	-	0xDF	
		L	-	-	-	-	-	-	-	0xEF	
		H	-	-	-	-	-	-	-	0xFF	- - - - - - - -

op	details	Dest	#	impl	abs	abs,X	abs,Y	abs,XY	zpg	rel	FLags
LD	M	A	0x43	-	0x53	0x63	0x73	0x83	0x93	-	T - - - T - - -
STO	A	M	-	-	0x04	0x14	0x24	0x34	0x44	-	T - - - T - - -
ADI	A + CF + M	A	0x1C	-	-	-	-	-	-	-	T - - - T - - T
CPI	A - M		0x1D	-	-	-	-	-	-	-	T - - - T - - T
ANI	A & M	A	0x1E	-	-	-	-	-	-	-	T - - - T - - -
TEST	M - 0		-	-	0x54	0x64	0x74	0x84	-	-	T - - - T - - -
INC	M + 1	M	-	-	0x55	0x65	0x75	0x85	-	-	T - - - T - - -
DEC	M - 1	M	-	-	0x56	0x66	0x76	0x86	-	-	T - - - T - - -
RR	fig 7	M	-	-	0x57	0x67	0x77	0x87	-	-	T - - - T - - T
RCL	fig 6	M	-	-	0x58	0x68	0x78	0x88	-	-	T - - - T - - T
SAL	fig 1	M	-	-	0x59	0x69	0x79	0x89	-	-	T - - - T - - T
SHR	fig 2	M	-	-	0x5A	0x6A	0x7A	0x8A	-	-	T - - - T - - T
COM	M ~	M	-	-	0x5B	0x6B	0x7B	0x8B	-	-	T - - - T - - T
NEG	0 - M	M	-	-	0x5C	0x6C	0x7C	0x8C	-	-	T - - - T - - -
RAL	fig 5	M	-	-	0x5D	0x6D	0x7D	0x8D	-	-	T - - - T - - -
ROR	fig 4	M	-	-	0x5E	0x6E	0x7E	0x8E	-	-	T - - - T - - -
CLR	0	M	-	-	0x5F	0x6F	0x7F	0x8F	-	-	1 - - - 0 - - 0
LDX	M	X	0x21	-	0x31	0x41	0x51	0x61	0x71	-	T - - - T - - -
STOX	X	M	-	-	0x05	0x15	0x25	0x35	0x45	-	T - - - T - - -
DEX	X - 1	X	-	0x4A	-	-	-	-	-	-	T - - - - - -
INX	X + 1	X	-	0x4B	-	-	-	-	-	-	T - - - - - -
LODY	M	Y	0x22	-	0x32	0x42	0x52	0x62	0x72	-	T - - - T - - -
STOY	Y	M	-	-	0x06	0x16	0x26	0x36	0x46	-	T - - - T - - -
TAY	A	Y	-	0x2B	-	-	-	-	-	-	- - - - T - - -
TYA	Y	A	-	0x2C	-	-	-	-	-	-	T - - - T - - -
DEY	Y - 1	Y	-	0x4C	-	-	-	-	-	-	T - - - - - -
INY	Y + 1	Y	-	0x4D	-	-	-	-	-	-	T - - - - - -
LODS	M	SP	0x20	-	0x30	0x40	0x50	0x60	0x70	-	T - - - T - - -
STOS	SP	M	-	-	0x0F	0x1F	0x2F	0x3F	0x4F	-	T - - - T - - -
MSA	FL	A	-	0x2D	-	-	-	-	-	-	- - - - - -
JUMP			-	-	0x38	-	-	-	-	-	- - - - - -
MVR	M	B	0x27	-	-	-	-	-	-	-	
		C	0x28	-	-	-	-	-	-	-	
		L	0x29	-	-	-	-	-	-	-	
		H	0x2A	-	-	-	-	-	-	-	T - - - T - - -
JMPR			-	-	0x07	-	-	-	-	-	- - - - - -
RT			-	0x23	-	-	-	-	-	-	- - - - - -
JCC	CF = 0		-	-	0x39	-	-	-	-	-	- - - - - -
JCS	CF = 1		-	-	0x3A	-	-	-	-	-	- - - - - -
JNE	ZF = 0		-	-	0x3B	-	-	-	-	-	- - - - - -
JEQ	ZF = 1		-	-	0x3C	-	-	-	-	-	- - - - - -
JMI	NF = 1		-	-	0x3D	-	-	-	-	-	- - - - - -
JPL	NF = 0		-	-	0x3E	-	-	-	-	-	- - - - - -
CCC	CF = 0		-	-	0x08	-	-	-	-	-	- - - - - -
CCS	CF = 1		-	-	0x09	-	-	-	-	-	- - - - - -
CNE	ZF = 0		-	-	0x0A	-	-	-	-	-	- - - - - -
CEQ	ZF = 1		-	-	0x0B	-	-	-	-	-	- - - - - -
CMI	NF = 1		-	-	0x0C	-	-	-	-	-	- - - - - -
CPL	NF = 0		-	-	0x0D	-	-	-	-	-	- - - - - -
CLC	CF = 0		-	0xA6	-	-	-	-	-	-	- - - - - 0
SEC	CF = 1		-	0xA7	-	-	-	-	-	-	- - - - - 1
CLI	IF = 0		-	0xA8	-	-	-	-	-	-	- - 0 - - - -
SEI	IF = 1		-	0xA9	-	-	-	-	-	-	- - 1 - - - -
CMC	CF ~		-	0xAA	-	-	-	-	-	-	- - - - - -
NOP			-	0x17	-	-	-	-	-	-	- - - - - -
WAI			-	0x18	-	-	-	-	-	-	- - - - - -
SWI	-		-	0x02	-	-	-	-	-	-	- - 1 - - - -
RTI	-		-	0x03	-	-	-	-	-	-	- - - - - -

Key	
A - Accumulator FL - Status Register - Inclusive or &- logical and -* - Push to stack and decrement stack pointer R - General Register CF - Carry FLaɡ NF - Negative FLaɡ	SP - StackPointer M - Memory (+)- Exclusive or ~ - Negation +* - Increment stack pointer and pop from stack X - Index Register ZF - Zero FLaɡ IF - Zero FLaɡ
FIGURE 1: $C \leftarrow [7 \quad \quad \quad 0] \leftarrow 0$	FIGURE 2: $N \rightarrow [7 \quad \quad \quad 0] \rightarrow C$
FIGURE 4: 	FIGURE 5 
FIGURE 6: 	FIGURE 7 