8086 Instruction Set

General purpose byte or word transfer instructions	
MOV	Copy byte or word from specified source to specified destination
PUSH	Copy specified word to top of stack
POP	Copy word from top of stack to specified location
PUSHA	Copy all registers to stack (80186/80188)
POPA	Copy words from stack to all registers (80186/80188)
XCHG	Exchange bytes or exchange words
XLAT	Translate a byte in AL using a table in memory

Simple input and output port transfer instructions	
IN	Copy a byte or word from specified port to accumulator
OUT	Copy a byte or word from accumulator to specified port
Special address transfer instructions	
LEA	Load effective address of operand into specified register
LDS	Load DS register and other specified register from memory
LES	Load ES register and other specified register from memory
Flag transfer instructions	
LAHF	Load (copy to) AH with the low byte of the flag register
SAHF	Store (copy) AH register to low byte of flag register
PUSHF	Copy flag register to top of stack
POPF	Copy word at top of stack to flag register

Arithmetic instructions Addition instructions	
ADD	Add specified byte to byte or specified word to word
ADC	Add byte + byte + carry flag or word + word +carry flag
INC	Increment specified byte or specified word by 1
AAA	ASCII adjust after addition
DAA	Decimal BCD adjust after addition
Subtraction instruction	
SUB	Subtract byte from byte or word from word
SBB	Subtract byte and carry flag from byte or word and carry flag from word
DEC	Decrement specified byte a specified word by 1
NEG	Negate – invert each bit of a specified byte or word and add 1 (form 2's complement)
CMP	Compare two specified bytes or two specified words
AAS	ASCII adjust after subtraction
DAS	Decimal (BCD) adjust after subtraction

Multiplication instructions	
MUL	Multiply unsigned byte by byte or unsigned word by word
IMUL	Multiply signed byte by byte or singed word by word
AAM	ASCII adjust after multiplication

Division instructions	
DIV	Divide unsigned word by byte or unsigned double word by word
IDIV	Divide signed word by byte or signed double word by word
AAD	ASCII adjust before division
CBW	Fill upper byte of word with copies of sign bit of lower byte
CWD	Fill upper word of double word with sign bit of lower word

BIT manipulation instructions Logical instructions	
NOT	Invert each bit of a byte or word
AND	AND each bit in a byte or word with the corresponding bit in another byte or word
OR	OR each bit in a byte or word with the corresponding bit in another byte or word
XOR	Exclusive OR each bit in a byte or word with the corresponding bit in another byte or word
TEST	AND operands to update flags, but don't change operands

Shift instructions	
SHL/SAL	Shift bits of word or byte left, put zero(s) in LSB(s)
SHR	Shift bits of words or byte right, put zero(s) in MSB(s)
SAR	Shift bits of word or byte right, copy old MSB into new MSB

Rotate instructions	
ROL	Rotate bits of byte or word left, MSB to left and to CF
ROR	Rotate bits of byte or word right, LSB to MSB and to CF
RCL	Rotate bits of byte or word left, MSB to CF and CF to LSB
RCR	Rotate bits of byte or word right, LSB to CF and CF to MSB

String instructions	
REP	An instruction prefix. Repeat following instruction until CX=0
REPE/REPZ	An instruction prefix. Repeat instruction until CX=0 or zero flag ZF=1
REPNE/ REPNZ	An instruction prefix. Repeat until CX=0 or ZF=1
MOVS/ MOVSB/ MOVSW	Move byte or word from one string to another
COMPS/ COMPSB/ COMPSW	Compare two string bytes or two string words
INS/INSB/ INSW	Input string byte or word from port (80186/80188)
OUTS/ OUTSB/ OUTSW	Output string byte or word to port (80186/80188)
SCAS/ SCASB/ SCASW	Scan a string. Compare a string byte with a byte in AL or a string word with a word in AX
LODS/ LODSB/ LODSW	Load string byte into AL or string word into AX.
STOS/ STOSB/ STOSW	Store byte from AL or word from AX into string

Program execution transfer instructions	
CALL	Call a procedure (subprogram), save return address on stack
RET	Return from procedure to calling program
JMP	Go to specified address to get next instruction

Conditional transfer instructions	
JA/JNBE	Jump if above/Jump if not below or equal
JAE/JNB	Jump if above or equal/Jump if not below
JB/JNAE	Jump if below/Jump if no above or equal
JC	Jump if carry flag, CF=1
JE/JZ	Jump if equal/Jump if zero flag ZF=1
JG/JNLE	Jump if greater/Jump if not less than or equal
JL/JNGE	Jump if less than/Jump if not greater than or equal
JLE/JNG	Jump if less than or equal/Jump if not greater than
JNC	Jump if no carry (CF=0)
JNE/JNZ	Jump if not equal/Jump if not zero (ZF=0)
JNO	Jump if no overflow (overflow flag OF=0)
JNP/JPO	Jump if not parity/Jump if parity odd (PF=0)
JNS	Jump if not sign (sign flag SF=0)
JO	Jump if overflow flag OF=1
JP/JPE	Jump if parity/Jump if parity even (PF=1)
JS	Jump if sign (SF=1)

Iteration control instructions	
LOOP	Loop through a sequence of instructions until CX=0
LOOPE/ LOOPZ	Loop through a sequence of instructions while ZF=1 and CX ≠ 0
LOOPNE/ LOOPNZ	Loop through a sequence of instructions while ZF=0 and CX≠0
JCXZ	Jump to specified adder if CX=0

Interrupt instructions	
INT	Interrupt program execution, call service procedure
INTO	Interrupt program execution if OF=1
IRET	Return from interrupt service procedure to main program

High-level language interface instructions	
ENTER	Enter procedure (80186/80188)
LEAVE	Leave procedure (80186/80188)
BOUND	Check if effective address within specified array bounds (80186/80188)

Processor control instructions Flag Set/Clear instructions	
STC	Set carry flag CF to 1
CLC	Clear carry flag CF to 0
CMC	Complement the state of the carry flag CF
STD	Set direction flag DF to 1 (decrement string pointers)
CLD	Clear direction flag DF to 0
STI	Set interrupt enable flag to 1 (enable INTR input)
CLI	Clear interrupt enable flag to 0 (disable INTR input)

External hardware synchronization instructions	
HLT	Halt (to nothing) until interrupt or reset
WAIT	Wait (do nothing) until signal on the TEST pin is low
ESC	Escape to external coprocessor such as 8087 or 8089
LOCK	An instruction prefix. Prevents another processor from taking the bus while the adjacent instruction executes

No operation instruction	
NOP	No action except fetch and decode