

8086 Instruction Set Summary**Data Transfer Instructions**

| | |
|-----------|---|
| MOV | Move byte or word to register or memory |
| IN, OUT | Input byte or word from port, output word to port |
| LEA | Load effective address |
| LDS, LES | Load pointer using data segment, extra segment |
| PUSH, POP | Push word onto stack, pop word off stack |
| XCHG | Exchange byte or word |
| XLAT | Translate byte using look-up table |

Logical Instructions

| | |
|------|--|
| NOT | Logical NOT of byte or word (one's complement) |
| AND | Logical AND of byte or word |
| OR | Logical OR of byte or word |
| XOR | Logical exclusive-OR of byte or word |
| TEST | Test byte or word (AND without storing) |

Shift and Rotate Instructions

| | |
|----------|---|
| SHL, SHR | Logical shift left, right byte or word? by 1 or CL |
| SAL, SAR | Arithmetic shift left, right byte or word? by 1 or CL |
| ROL, ROR | Rotate left, right byte or word? by 1 or CL |
| RCL, RCR | Rotate left, right through carry byte or word? by 1 or CL |

Arithmetic Instructions

| | |
|------------|---|
| ADD, SUB | Add, subtract byte or word |
| ADC, SBB | Add, subtract byte or word and carry (borrow) |
| INC, DEC | Increment, decrement byte or word |
| NEG | Negate byte or word (two's complement) |
| CMP | Compare byte or word (subtract without storing) |
| MUL, DIV | Multiply, divide byte or word (unsigned) |
| IMUL, IDIV | Integer multiply, divide byte or word (signed) |
| CBW, CWD | Convert byte to word, word to double word (useful before multiply/divide) |

Adjustments after arithmetic operations:

| | |
|--------------------|--|
| AAA, AAS, AAM, AAD | ASCII adjust for addition, subtraction, multiplication, division (ASCII codes 30-39) |
| DAA, DAS | Decimal adjust for addition, subtraction (binary coded decimal numbers) |

Transfer Instructions

| | |
|-----|--|
| JMP | Unconditional jump (<i>short</i> ?127/8, <i>near</i> ?32K, <i>far</i> between segments) |
|-----|--|

Conditional jumps:

| | |
|-----------|---|
| JA (JNBE) | Jump if above (not below or equal)? +127, -128 range only |
| JAE (JNB) | Jump if above or equal(not below)? +127, -128 range only |
| JB (JNAE) | Jump if below (not above or equal)? +127, -128 range only |
| JBE (JNA) | Jump if below or equal (not above)? +127, -128 range only |
| JE (JZ) | Jump if equal (zero)? +127, -128 range only |
| JG (JNLE) | Jump if greater (not less or equal)? +127, -128 range only |
| JGE (JNL) | Jump if greater or equal (not less)? +127, -128 range only |
| JL (JNGE) | Jump if less (not greater nor equal)? +127, -128 range only |
| JLE (JNG) | Jump if less or equal (not greater)? +127, -128 range only |
| JC, JNC | Jump if carry set, carry not set? +127, -128 range only |
| JO, JNO | Jump if overflow, no overflow? +127, -128 range only |
| JS, JNS | Jump if sign, no sign? +127, -128 range only |
| JNP (JPO) | Jump if no parity (parity odd)? +127, -128 range only |
| JP (JPE) | Jump if parity (parity even)? +127, -128 range only |

Loop control:

| | |
|-----------------|---|
| LOOP | Loop unconditional, count in CX, short jump to target address |
| LOOPE (LOOPZ) | Loop if equal (zero), count in CX, short jump to target address |
| LOOPNE (LOOPNZ) | Loop if not equal (not zero), count in CX, short jump to target address |
| JCXZ | Jump if CX equals zero (used to skip code in loop) |

Subroutine and Interrupt Instructions

| | |
|-----------|---|
| CALL, RET | Call, return from procedure (inside or outside current segment) |
| INT, INTO | Software interrupt, interrupt if overflow |
| IRET | Return from interrupt |

String Instructions

| | |
|--------------|---|
| MOVS | Move byte or word string |
| MOVSB, MOVSW | Move byte, word string |
| CMPS | Compare byte or word string |
| SCAS | Scan byte or word string (comparing to A or AX) |
| LDS, STOS | Load, store byte or word string to AL or AX |

Repeat instructions (placed in front of other string operations):

| | |
|--------------|-------------------------------|
| REP | Repeat |
| REPE, REPZ | Repeat while equal, zero |
| REPNE, REPNZ | Repeat while not equal (zero) |

Processor Control Instructions

Flag manipulation:

| | |
|---------------|--|
| STC, CLC, CMC | Set, clear, complement carry flag |
| STD, CLD | Set, clear direction flag |
| STI, CLI | Set, clear interrupt enable flag |
| LAHF, SAHF | Load AH from flags, store AH into flags |
| PUSHF, POPF | Push flags onto stack, pop flags off stack |

Coprocessor, multiprocessor interface:

| | |
|------|--|
| ESC | Escape to external processor interface |
| LOCK | Lock bus during next instruction |

Inactive states:

| | |
|------|----------------------------|
| NOP | No operation |
| WAIT | Wait for TEST pin activity |
| HLT | Halt processor |