Data Transfer Instructions

The data transfer instructions move data between memory and the general-purpose and segment registers.

MOV	Move data between general-purpose registers; move data between memory and general-purpose or segment registers; move immediates to general-purpose registers
XCHG	Exchange
PUSH	Push onto stack
POP	Pop off of stack
PUSHA	Push general-purpose registers onto stack
POPA	Pop general-purpose registers from stack
IN	Read from a port
OUT	Write to a port
CWD	Convert word to doubleword
CBW	Convert byte to word

Binary Arithmetic Instructions

The binary arithmetic instructions perform basic binary integer computations on byte, word, and doubleword integers located in memory and/or the general purpose registers.

ADD	Integer add
ADC	Add with carry
SUB	Subtract
SBB	Subtract with borrow
IMUL	Signed multiply
MUL	Unsigned multiply
IDIV	Signed divide
DIV	Unsigned divide
INC	Increment
DEC	Decrement
NEG	Negate
CMP	Compare

Decimal Arithmetic

The decimal arithmetic instructions perform decimal arithmetic on binary coded decimal (BCD) data.

DAA	Decimal adjust after addition
DAS	Decimal adjust after subtraction
AAA	ASCII adjust after addition
AAS	ASCII adjust after subtraction
AAM	ASCII adjust after multiplication
AAD	ASCII adjust before division

Logical Instructions

The logical instructions perform basic AND, OR, XOR, and NOT logical operations on byte, word, and doubleword values.

AND	Perform bitwise logical AND
OR	Perform bitwise logical OR
XOR	Perform bitwise logical exclusive OR
NOT	Perform bitwise logical NOT

Shift and Rotate Instructions

The shift and rotate instructions shift and rotate the bits in word and doubleword operands

SAR	Shift arithmetic right
SHR	Shift logical right
SAL/SHL	Shift arithmetic left/Shift logical left
ROR	Rotate right
ROL	Rotate left
RCR	Rotate through carry right
RCL	Rotate through carry left

Control Transfer Instructions

The control transfer instructions provide jump, conditional jump, loop, and call and return operations to control program flow.

JMP	Jump
JE/JZ	Jump if equal/Jump if zero
JNE/JNZ	Jump if not equal/Jump if not zero
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 not above or equal
JBE/JNA	Jump if below or equal/Jump if not above
JG/JNLE	Jump if greater/Jump if not less or equal
JGE/JNL	Jump if greater or equal/Jump if not less
JL/JNGE	Jump if less/Jump if not greater or equal
JLE/JNG	Jump if less or equal/Jump if not greater
JC	Jump if carry
JNC	Jump if not carry
JO	Jump if overflow
JNO	Jump if not overflow
JS	Jump if sign (negative)
JNS	Jump if not sign (non-negative)
JPO/JNP	Jump if parity odd/Jump if not parity
JPE/JP	Jump if parity even/Jump if parity
JCXZ/JECXZ	Jump register CX zero/Jump register ECX zero
LOOP	Loop with ECX counter
LOOPZ/LOOPE	Loop with ECX and zero/Loop with ECX and equal
LOOPNZ/LOOPNE	Loop with ECX and not zero/Loop with ECX and not equal
CALL	Call procedure
RET	Return
IRET	Return from interrupt
INT	Software interrupt
INTO	Interrupt on overflow

String Instructions

The string instructions operate on strings of bytes, allowing them to be moved to and from memory.

MOVS/MOVSB	Move string/Move byte string
MOVS/MOVSW	Move string/Move word string
CMPS/CMPSB	Compare string/Compare byte string
CMPS/CMPSW	Compare string/Compare word string Scan
SCAS/SCASB	string/Scan byte string
SCAS/SCASW	Scan string/Scan word string
LODS/LODSB	Load string/Load byte string
LODS/LODSW	Load string/Load word string
STOS/STOSB	Store string/Store byte string
STOS/STOSW	Store string/Store word string
REP	Repeat while CX not zero
REPE/REPZ	Repeat while equal/Repeat while zero
REPNE/REPNZ	Repeat while not equal/Repeat while not zero

Flag Control Instructions

The flag control instructions operate on the flags in the EFLAGS register.

STC	Set carry flag
CLC	Clear the carry flag
CMC	Complement the carry flag
CLD	Clear the direction flag
STD	Set direction flag
LAHF	Load flags into AH register
SAHF	Store AH register into flags
PUSHF	Push EFLAGS onto stack
POPF	Pop EFLAGS from stack
STI	Set interrupt flag
CLI	Clear the interrupt flag

Segment Register Instructions

The segment register instructions allow far pointers (segment addresses) to be loaded into the segment registers.

LDS	Load far pointer using DS
LES	Load far pointer using ES

Miscellaneous Instructions

The miscellaneous instructions provide such functions as loading an effective address, executing a "no-operation," and retrieving processor identification information.

LEA	Load effective address
NOP	No operation
XLAT/XLATB	Table lookup translation