INTEL ARCHITECTURE MMX™ INSTRUCTION SET

The IA MMX technology defines fifty-seven new instructions. The instructions are grouped into the following functional categories:

- Arithmetic Instructions
- Comparison Instructions
- Conversion Instructions
- Logical Instructions
- Shift Instructions
- Data Transfer Instructions
- Empty MMX State (EMMS) Instruction
Many of the instructions have multiple variations depending on the data types they support. Each variation has a different suffix.

For example the PADD instruction has three variations: PADDB, PADDW, and PADDQ, where the letters B, W, and D represent byte, word, and doubleword.

Instructions vary by:

- Data type: packed bytes, packed words, packed doublewords or quadwords
- Signed - Unsigned numbers
- Wraparound - Saturate arithmetic
A typical MMX instruction has this syntax:

Prefix: P for Packed

Instruction operation: for example - ADD, CMP, or XOR

Suffix:

--US for Unsigned Saturation

--S for Signed saturation

--B, W, D, Q for the data type: packed byte, packed word, packed doubleword, or quadword.

Instructions that have different input and output data elements have two data-type suffixes. For example, the conversion instruction converts from one data type to another. It has two suffixes: one for the original data type and the second for the converted data type.
This is an example of an instruction mnemonic syntax:

PADDUSW (Packed Add Unsigned with Saturation for Word)

P = Packed

ADD = the instruction operation

US = Unsigned Saturation

W = Word
The following conventions apply to all MMX instructions (except the EMMS instruction):

The instructions reference and operate on two operands: the source and destination operands. The right operand is the source and the left operand is the destination. The destination operand may also supply one of the inputs for the operation. The instruction overwrites the destination operand with the result.

When one of the operands is a memory location, the linear address corresponds to the address of the least significant byte of the referenced memory data.

The MMX instructions do not affect the condition flags.
Packed Addition and Subtraction

The PADD (Packed Add) and PSUB (Packed Subtract) instructions add or subtract the signed or unsigned data elements of the source operand to or from the destination operand in wrap-around mode. These instructions support packed byte, packed word, and packed doubleword data types.

The PADDS (Packed Add with Saturation) and PSUBS (Packed Subtract with Saturation) instructions add or subtract the signed data elements of the source operand to or from the signed data elements of the destination operand and saturate the result to the limits of the signed data-type range. These instructions support packed byte and packed word data types.
The PADDUS (Packed Add Unsigned with Saturation) and PSUBUS (Packed Subtract Unsigned with Saturation) instructions add or subtract the unsigned data elements of the source operand to or from the unsigned data elements of the destination operand and saturate the result to the limits of the unsigned data-type range. These instructions support packed byte and packed word data types.

**Packed Multiply Add**

The PMADDWD (Packed Multiply and Add) instruction calculates the products of the signed words of the source and destination operands. The four intermediate 32-bit doubleword products are summed in pairs to produce two 32-bit doubleword results.
Packed Multiplication

Packed multiplication instructions perform four multiplications on pairs of signed 16-bit operands, producing 32-bit intermediate results. Users may choose the low-order or high-order parts of each 32-bit result.
Comparison Instructions

The PCMPEQ (Packed Compare for Equal) and PCMPGT (Packed Compare for Greater Than) instructions compare the corresponding data elements in the source and destination operands for equality or value greater than, respectively.

These instructions generate a mask of ones or zeros which are written to the destination operand. Logical operations can use the mask to select elements.

This can be used to implement a packed conditional move operation without a branch or a set of branch instructions. No flags are set.
Pack and Unpack

The Pack and Unpack instructions perform conversions between the packed data types.

The PACKSS (Packed with Signed Saturation) instruction converts signed words into signed bytes or signed doublewords into signed words, in signed saturation mode.

The PACKUS (Packed with Unsigned Saturation) instruction converts signed words into unsigned bytes, in unsigned saturation mode.

The PUNPCKH (Unpack High Packed Data) and PUNPCKL (Unpack Low Packed Data) instructions convert bytes to words, words to doublewords, or doublewords to quadwords.
Logical Instructions

The PAND (Bitwise Logical And), PANDN (Bitwise Logical And Not), POR (Bitwise Logical OR), and PXOR (Bitwise Logical Exclusive OR) instructions perform bitwise logical operations on 64-bit quantities.

Data Transfer Instructions

The MOVD (Move 32 Bits) instruction transfers 32 bits of packed data from memory to MMX registers and visa versa, or from integer registers to MMX registers and visa versa.

The MOVQ (Move 64 Bits) instruction transfers 64-bits of packed data from memory to MMX registers and vise versa, or transfers data between MMX registers.
Shift Instructions

The logical shift left, logical shift right and arithmetic shift right instructions shift each element by a specified number of bits. The logical left and right shifts also enable a 64-bit quantity (quadword) to be shifted as one block, assisting in data type conversions and alignment operations.

The PSLL (Packed Shift Left Logical) and PSRL (Packed Shift Right Logical) instructions perform a logical left or right shift, and fill the empty high or low order bit positions with zeros. These instructions support packed word, packed doubleword, and quadword data types.

The PSRA (Packed Shift Right Arithmetic) instruction performs an arithmetic right shift, copying the sign bit into empty bit positions on the upper end of the operand.