[**X86 Opcode and Instruction Reference Home**](http://ref.x86asm.net/index.html)

Other editions: [coder64-abc](http://ref.x86asm.net/coder64-abc.html), [coder-abc](http://ref.x86asm.net/coder-abc.html), [geek32-abc](http://ref.x86asm.net/geek32-abc.html), [geek64-abc](http://ref.x86asm.net/geek64-abc.html), [geek-abc](http://ref.x86asm.net/geek-abc.html)

[32-bit ModR/M Byte](http://ref.x86asm.net/coder32-abc.html#modrm_byte_32) | [32-bit SIB Byte](http://ref.x86asm.net/coder32-abc.html#sib_byte_32)

[16-bit ModR/M Byte](http://ref.x86asm.net/coder32-abc.html#modrm_byte_16)

**alphabetic index:**

[A](http://ref.x86asm.net/coder32-abc.html#A)  [B](http://ref.x86asm.net/coder32-abc.html#B)  [C](http://ref.x86asm.net/coder32-abc.html#C)  [D](http://ref.x86asm.net/coder32-abc.html#D)  [E](http://ref.x86asm.net/coder32-abc.html#E)  [F](http://ref.x86asm.net/coder32-abc.html#F)  [G](http://ref.x86asm.net/coder32-abc.html#G)  [H](http://ref.x86asm.net/coder32-abc.html#H)  [I](http://ref.x86asm.net/coder32-abc.html#I)  [J](http://ref.x86asm.net/coder32-abc.html#J)  [L](http://ref.x86asm.net/coder32-abc.html#L)  [M](http://ref.x86asm.net/coder32-abc.html#M)  [N](http://ref.x86asm.net/coder32-abc.html#N)  [O](http://ref.x86asm.net/coder32-abc.html#O)  [P](http://ref.x86asm.net/coder32-abc.html#P)  [R](http://ref.x86asm.net/coder32-abc.html#R)  [S](http://ref.x86asm.net/coder32-abc.html#S)  [T](http://ref.x86asm.net/coder32-abc.html#T)  [U](http://ref.x86asm.net/coder32-abc.html#U)  [V](http://ref.x86asm.net/coder32-abc.html#V)  [W](http://ref.x86asm.net/coder32-abc.html#W)  [X](http://ref.x86asm.net/coder32-abc.html#X)

| [mnemonic](http://ref.x86asm.net/#column_mnemonic) | [op1](http://ref.x86asm.net/#column_op) | [op2](http://ref.x86asm.net/#column_op) | [op3](http://ref.x86asm.net/#column_op) | [op4](http://ref.x86asm.net/#column_op) | [iext](http://ref.x86asm.net/#column_iext) | [pf](http://ref.x86asm.net/#column_pf) | [0F](http://ref.x86asm.net/#column_0F) | [po](http://ref.x86asm.net/#column_po) | [so](http://ref.x86asm.net/#column_so) | [o](http://ref.x86asm.net/#column_o) | [proc](http://ref.x86asm.net/#column_proc) | [st](http://ref.x86asm.net/#column_st) | [m](http://ref.x86asm.net/#column_m) | [rl](http://ref.x86asm.net/#column_rl) | [x](http://ref.x86asm.net/#column_x) | [tested f](http://ref.x86asm.net/#column_tested_f_modif_f_def_f_undef_f) | [modif f](http://ref.x86asm.net/#column_tested_f_modif_f_def_f_undef_f) | [def f](http://ref.x86asm.net/#column_tested_f_modif_f_def_f_undef_f) | [undef f](http://ref.x86asm.net/#column_tested_f_modif_f_def_f_undef_f) | [f values](http://ref.x86asm.net/#column_f_values) | [description, notes](http://ref.x86asm.net/#column_description_notes) |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| AAA | ***AL*** | ***AH*** |  |  |  |  |  | 37 |  |  |  |  |  |  |  | .....a.. | o..szapc | .....a.c | o..sz.p. |  | ASCII Adjust After Addition |
| AAD | ***AL*** | ***AH*** |  |  |  |  |  | D5 | 0A |  |  |  |  |  |  |  | o..szapc | ...sz.p. | o....a.c |  | ASCII Adjust AX Before Division |
| AAM | ***AL*** | ***AH*** |  |  |  |  |  | D4 | 0A |  |  |  |  |  |  |  | o..szapc | ...sz.p. | o....a.c |  | ASCII Adjust AX After Multiply |
| AAS | ***AL*** | ***AH*** |  |  |  |  |  | 3F |  |  |  |  |  |  |  | .....a.. | o..szapc | .....a.c | o..sz.p. |  | ASCII Adjust AL After Subtraction |
| ADC | **r/m8** | r8 |  |  |  |  |  | 10 |  | r |  |  |  |  | L | .......c | o..szapc | o..szapc |  |  | Add with Carry |
| ADC | **r/m16/32** | r16/32 |  |  |  |  |  | 11 |  | r |  |  |  |  | L | .......c | o..szapc | o..szapc |  |  | Add with Carry |
| ADC | **r8** | r/m8 |  |  |  |  |  | 12 |  | r |  |  |  |  |  | .......c | o..szapc | o..szapc |  |  | Add with Carry |
| ADC | **r16/32** | r/m16/32 |  |  |  |  |  | 13 |  | r |  |  |  |  |  | .......c | o..szapc | o..szapc |  |  | Add with Carry |
| ADC | **AL** | imm8 |  |  |  |  |  | 14 |  |  |  |  |  |  |  | .......c | o..szapc | o..szapc |  |  | Add with Carry |
| ADC | **eAX** | imm16/32 |  |  |  |  |  | 15 |  |  |  |  |  |  |  | .......c | o..szapc | o..szapc |  |  | Add with Carry |
| ADC | **r/m8** | imm8 |  |  |  |  |  | 80 |  | 2 |  |  |  |  | L | .......c | o..szapc | o..szapc |  |  | Add with Carry |
| ADC | **r/m16/32** | imm16/32 |  |  |  |  |  | 81 |  | 2 |  |  |  |  | L | .......c | o..szapc | o..szapc |  |  | Add with Carry |
| ADC | **r/m8** | imm8 |  |  |  |  |  | 82 |  | 2 |  |  |  |  | L | .......c | o..szapc | o..szapc |  |  | Add with Carry |
| ADC | **r/m16/32** | imm8 |  |  |  |  |  | 83 |  | 2 |  |  |  |  | L | .......c | o..szapc | o..szapc |  |  | Add with Carry |
| ADD | **r/m8** | r8 |  |  |  |  |  | 00 |  | r |  |  |  |  | L |  | o..szapc | o..szapc |  |  | Add |
| ADD | **r/m16/32** | r16/32 |  |  |  |  |  | 01 |  | r |  |  |  |  | L |  | o..szapc | o..szapc |  |  | Add |
| ADD | **r8** | r/m8 |  |  |  |  |  | 02 |  | r |  |  |  |  |  |  | o..szapc | o..szapc |  |  | Add |
| ADD | **r16/32** | r/m16/32 |  |  |  |  |  | 03 |  | r |  |  |  |  |  |  | o..szapc | o..szapc |  |  | Add |
| ADD | **AL** | imm8 |  |  |  |  |  | 04 |  |  |  |  |  |  |  |  | o..szapc | o..szapc |  |  | Add |
| ADD | **eAX** | imm16/32 |  |  |  |  |  | 05 |  |  |  |  |  |  |  |  | o..szapc | o..szapc |  |  | Add |
| ADD | **r/m8** | imm8 |  |  |  |  |  | 80 |  | 0 |  |  |  |  | L |  | o..szapc | o..szapc |  |  | Add |
| ADD | **r/m16/32** | imm16/32 |  |  |  |  |  | 81 |  | 0 |  |  |  |  | L |  | o..szapc | o..szapc |  |  | Add |
| ADD | **r/m8** | imm8 |  |  |  |  |  | 82 |  | 0 |  |  |  |  | L |  | o..szapc | o..szapc |  |  | Add |
| ADD | **r/m16/32** | imm8 |  |  |  |  |  | 83 |  | 0 |  |  |  |  | L |  | o..szapc | o..szapc |  |  | Add |
| ADDPD | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 58 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Add Packed Double-FP Values |
| ADDPS | **xmm** | xmm/m128 |  |  | sse1 |  | 0F | 58 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Add Packed Single-FP Values |
| ADDSD | **xmm** | xmm/m64 |  |  | sse2 | F2 | 0F | 58 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Add Scalar Double-FP Values |
| ADDSS | **xmm** | xmm/m32 |  |  | sse1 | F3 | 0F | 58 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Add Scalar Single-FP Values |
| ADDSUBPD | **xmm** | xmm/m128 |  |  | sse3 | 66 | 0F | D0 |  | r | P4++ |  |  |  |  |  |  |  |  |  | Packed Double-FP Add/Subtract |
| ADDSUBPS | **xmm** | xmm/m128 |  |  | sse3 | F2 | 0F | D0 |  | r | P4++ |  |  |  |  |  |  |  |  |  | Packed Single-FP Add/Subtract |
| *ADX* | ***AL*** | ***AH*** | imm8 |  |  |  |  | D5 |  |  |  |  |  |  |  |  | o..szapc | ...sz.p. | o....a.c |  | Adjust AX Before Division |
| *AMX* | ***AL*** | ***AH*** | imm8 |  |  |  |  | D4 |  |  |  |  |  |  |  |  | o..szapc | ...sz.p. | o....a.c |  | Adjust AX After Multiply |
| AND | **r/m8** | r8 |  |  |  |  |  | 20 |  | r |  |  |  |  | L |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical AND |
| AND | **r/m16/32** | r16/32 |  |  |  |  |  | 21 |  | r |  |  |  |  | L |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical AND |
| AND | **r8** | r/m8 |  |  |  |  |  | 22 |  | r |  |  |  |  |  |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical AND |
| AND | **r16/32** | r/m16/32 |  |  |  |  |  | 23 |  | r |  |  |  |  |  |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical AND |
| AND | **AL** | imm8 |  |  |  |  |  | 24 |  |  |  |  |  |  |  |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical AND |
| AND | **eAX** | imm16/32 |  |  |  |  |  | 25 |  |  |  |  |  |  |  |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical AND |
| AND | **r/m8** | imm8 |  |  |  |  |  | 80 |  | 4 |  |  |  |  | L |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical AND |
| AND | **r/m16/32** | imm16/32 |  |  |  |  |  | 81 |  | 4 |  |  |  |  | L |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical AND |
| AND | **r/m8** | imm8 |  |  |  |  |  | 82 |  | 4 |  |  |  |  | L |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical AND |
| AND | **r/m16/32** | imm8 |  |  |  |  |  | 83 |  | 4 | 03+ |  |  |  | L |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical AND |
| ANDNPD | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 55 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Bitwise Logical AND NOT of Packed Double-FP Values |
| ANDNPS | **xmm** | xmm/m128 |  |  | sse1 |  | 0F | 55 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Bitwise Logical AND NOT of Packed Single-FP Values |
| ANDPD | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 54 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Bitwise Logical AND of Packed Double-FP Values |
| ANDPS | **xmm** | xmm/m128 |  |  | sse1 |  | 0F | 54 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Bitwise Logical AND of Packed Single-FP Values |
| ARPL | r/m16 | r16 |  |  |  |  |  | 63 |  | r | 02+ |  |  |  |  |  | ....z... | ....z... |  |  | Adjust RPL Field of Segment Selector |
| BLENDPD | **xmm** | xmm/m128 | imm8 |  | sse41 | 66 | 0F | 3A | 0D | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Blend Packed Double-FP Values |
| BLENDPS | **xmm** | xmm/m128 | imm8 |  | sse41 | 66 | 0F | 3A | 0C | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Blend Packed Single-FP Values |
| BLENDVPD | **xmm** | xmm/m128 | *XMM0* |  | sse41 | 66 | 0F | 38 | 15 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Variable Blend Packed Double-FP Values |
| BLENDVPS | **xmm** | xmm/m128 | *XMM0* |  | sse41 | 66 | 0F | 38 | 14 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Variable Blend Packed Single-FP Values |
| BOUND | r16/32 | m16/32&16/32 | *eFlags* |  |  |  |  | 62 |  | r | 01+ |  |  | f |  |  | ..i..... | ..i..... |  | ..i..... | Check Array Index Against Bounds |
| BSF | **r16/32** | r/m16/32 |  |  |  |  | 0F | BC |  | r | 03+ |  |  |  |  |  | o..szapc | ....z... | o..s.apc |  | Bit Scan Forward |
| BSR | **r16/32** | r/m16/32 |  |  |  |  | 0F | BD |  | r | 03+ |  |  |  |  |  | o..szapc | ....z... | o..s.apc |  | Bit Scan Reverse |
| BSWAP | **r16/32** |  |  |  |  |  | 0F | C8+r | |  | 04+ | D[21](http://ref.x86asm.net/coder32-abc.html#gen_note_BSWAP_0FC8) |  |  |  |  |  |  |  |  | Byte Swap |
| BT | r/m16/32 | r16/32 |  |  |  |  | 0F | A3 |  | r | 03+ |  |  |  |  |  | o..szapc | .......c | o..szap. |  | Bit Test |
| BT | r/m16/32 | imm8 |  |  |  |  | 0F | BA |  | 4 | 03+ |  |  |  |  |  | o..szapc | .......c | o..szap. |  | Bit Test |
| BTC | **r/m16/32** | imm8 |  |  |  |  | 0F | BA |  | 7 | 03+ |  |  |  | L |  | o..szapc | .......c | o..szap. |  | Bit Test and Complement |
| BTC | **r/m16/32** | r16/32 |  |  |  |  | 0F | BB |  | r | 03+ |  |  |  | L |  | o..szapc | .......c | o..szap. |  | Bit Test and Complement |
| BTR | **r/m16/32** | r16/32 |  |  |  |  | 0F | B3 |  | r | 03+ |  |  |  | L |  | o..szapc | .......c | o..szap. |  | Bit Test and Reset |
| BTR | **r/m16/32** | imm8 |  |  |  |  | 0F | BA |  | 6 | 03+ |  |  |  | L |  | o..szapc | .......c | o..szap. |  | Bit Test and Reset |
| BTS | **r/m16/32** | r16/32 |  |  |  |  | 0F | AB |  | r | 03+ |  |  |  | L |  | o..szapc | .......c | o..szap. |  | Bit Test and Set |
| BTS | **r/m16/32** | imm8 |  |  |  |  | 0F | BA |  | 5 | 03+ |  |  |  | L |  | o..szapc | .......c | o..szap. |  | Bit Test and Set |
| CALL | rel16/32 |  |  |  |  |  |  | E8 |  |  |  |  |  |  |  |  |  |  |  |  | Call Procedure |
| CALL | r/m16/32 |  |  |  |  |  |  | FF |  | 2 |  |  |  |  |  |  |  |  |  |  | Call Procedure |
| CALLF | ptr16:16/32 |  |  |  |  |  |  | 9A |  |  |  |  |  |  |  |  |  |  |  |  | Call Procedure |
| CALLF | m16:16/32 |  |  |  |  |  |  | FF |  | 3 |  |  |  |  |  |  |  |  |  |  | Call Procedure |
| CBW | ***AX*** | *AL* |  |  |  |  |  | 98 |  |  |  |  |  |  |  |  |  |  |  |  | Convert Byte to Word |
| CDQ | ***EDX*** | *EAX* |  |  |  |  |  | 99 |  |  | 03+ |  |  |  |  |  |  |  |  |  | Convert Doubleword to Quadword |
| CLC |  |  |  |  |  |  |  | F8 |  |  |  |  |  |  |  |  | .......c | .......c |  | .......c | Clear Carry Flag |
| CLD |  |  |  |  |  |  |  | FC |  |  |  |  |  |  |  |  | .d...... | .d...... |  | .d...... | Clear Direction Flag |
| CLFLUSH | m8 |  |  |  | sse2 |  | 0F | AE |  | 7 | P4+ |  |  |  |  |  |  |  |  |  | Flush Cache Line |
| CLI |  |  |  |  |  |  |  | FA |  |  |  |  |  | f[1](http://ref.x86asm.net/coder32-abc.html#rflags_iopl) |  |  | ..i..... | ..i..... |  | ..i..... | Clear Interrupt Flag |
| CLTS | ***CR0*** |  |  |  |  |  | 0F | 06 |  |  | 02+ |  |  | 0 |  |  |  |  |  |  | Clear Task-Switched Flag in CR0 |
| CMC |  |  |  |  |  |  |  | F5 |  |  |  |  |  |  |  | .......c | .......c | .......c |  |  | Complement Carry Flag |
| CMOVB | **r16/32** | r/m16/32 |  |  |  |  | 0F | 42 |  | r | PP+ |  |  |  |  | .......c |  |  |  |  | Conditional Move - below/not above or equal/carry (CF=1) |
| CMOVNAE | **r16/32** | r/m16/32 |  |  |
| CMOVC | **r16/32** | r/m16/32 |  |  |
| CMOVBE | **r16/32** | r/m16/32 |  |  |  |  | 0F | 46 |  | r | PP+ |  |  |  |  | ....z..c |  |  |  |  | Conditional Move - below or equal/not above (CF=1 OR ZF=1) |
| CMOVNA | **r16/32** | r/m16/32 |  |  |
| CMOVL | **r16/32** | r/m16/32 |  |  |  |  | 0F | 4C |  | r | PP+ |  |  |  |  | o..s.... |  |  |  |  | Conditional Move - less/not greater (SF!=OF) |
| CMOVNGE | **r16/32** | r/m16/32 |  |  |
| CMOVLE | **r16/32** | r/m16/32 |  |  |  |  | 0F | 4E |  | r | PP+ |  |  |  |  | o..sz... |  |  |  |  | Conditional Move - less or equal/not greater ((ZF=1) OR (SF!=OF)) |
| CMOVNG | **r16/32** | r/m16/32 |  |  |
| CMOVNB | **r16/32** | r/m16/32 |  |  |  |  | 0F | 43 |  | r | PP+ |  |  |  |  | .......c |  |  |  |  | Conditional Move - not below/above or equal/not carry (CF=0) |
| CMOVAE | **r16/32** | r/m16/32 |  |  |
| CMOVNC | **r16/32** | r/m16/32 |  |  |
| CMOVNBE | **r16/32** | r/m16/32 |  |  |  |  | 0F | 47 |  | r | PP+ |  |  |  |  | ....z..c |  |  |  |  | Conditional Move - not below or equal/above (CF=0 AND ZF=0) |
| CMOVA | **r16/32** | r/m16/32 |  |  |
| CMOVNL | **r16/32** | r/m16/32 |  |  |  |  | 0F | 4D |  | r | PP+ |  |  |  |  | o..s.... |  |  |  |  | Conditional Move - not less/greater or equal (SF=OF) |
| CMOVGE | **r16/32** | r/m16/32 |  |  |
| CMOVNLE | **r16/32** | r/m16/32 |  |  |  |  | 0F | 4F |  | r | PP+ |  |  |  |  | o..sz... |  |  |  |  | Conditional Move - not less nor equal/greater ((ZF=0) AND (SF=OF)) |
| CMOVG | **r16/32** | r/m16/32 |  |  |
| CMOVNO | **r16/32** | r/m16/32 |  |  |  |  | 0F | 41 |  | r | PP+ |  |  |  |  | o....... |  |  |  |  | Conditional Move - not overflow (OF=0) |
| CMOVNP | **r16/32** | r/m16/32 |  |  |  |  | 0F | 4B |  | r | PP+ |  |  |  |  | ......p. |  |  |  |  | Conditional Move - not parity/parity odd (PF=0) |
| CMOVPO | **r16/32** | r/m16/32 |  |  |
| CMOVNS | **r16/32** | r/m16/32 |  |  |  |  | 0F | 49 |  | r | PP+ |  |  |  |  | ...s.... |  |  |  |  | Conditional Move - not sign (SF=0) |
| CMOVNZ | **r16/32** | r/m16/32 |  |  |  |  | 0F | 45 |  | r | PP+ |  |  |  |  | ....z... |  |  |  |  | Conditional Move - not zero/not equal (ZF=0) |
| CMOVNE | **r16/32** | r/m16/32 |  |  |
| CMOVO | **r16/32** | r/m16/32 |  |  |  |  | 0F | 40 |  | r | PP+ |  |  |  |  | o....... |  |  |  |  | Conditional Move - overflow (OF=1) |
| CMOVP | **r16/32** | r/m16/32 |  |  |  |  | 0F | 4A |  | r | PP+ |  |  |  |  | ......p. |  |  |  |  | Conditional Move - parity/parity even (PF=1) |
| CMOVPE | **r16/32** | r/m16/32 |  |  |
| CMOVS | **r16/32** | r/m16/32 |  |  |  |  | 0F | 48 |  | r | PP+ |  |  |  |  | ...s.... |  |  |  |  | Conditional Move - sign (SF=1) |
| CMOVZ | **r16/32** | r/m16/32 |  |  |  |  | 0F | 44 |  | r | PP+ |  |  |  |  | ....z... |  |  |  |  | Conditional Move - zero/equal (ZF=1) |
| CMOVE | **r16/32** | r/m16/32 |  |  |
| CMP | r/m8 | r8 |  |  |  |  |  | 38 |  | r |  |  |  |  |  |  | o..szapc | o..szapc |  |  | Compare Two Operands |
| CMP | r/m16/32 | r16/32 |  |  |  |  |  | 39 |  | r |  |  |  |  |  |  | o..szapc | o..szapc |  |  | Compare Two Operands |
| CMP | r8 | r/m8 |  |  |  |  |  | 3A |  | r |  |  |  |  |  |  | o..szapc | o..szapc |  |  | Compare Two Operands |
| CMP | r16/32 | r/m16/32 |  |  |  |  |  | 3B |  | r |  |  |  |  |  |  | o..szapc | o..szapc |  |  | Compare Two Operands |
| CMP | AL | imm8 |  |  |  |  |  | 3C |  |  |  |  |  |  |  |  | o..szapc | o..szapc |  |  | Compare Two Operands |
| CMP | eAX | imm16/32 |  |  |  |  |  | 3D |  |  |  |  |  |  |  |  | o..szapc | o..szapc |  |  | Compare Two Operands |
| CMP | r/m8 | imm8 |  |  |  |  |  | 80 |  | 7 |  |  |  |  |  |  | o..szapc | o..szapc |  |  | Compare Two Operands |
| CMP | r/m16/32 | imm16/32 |  |  |  |  |  | 81 |  | 7 |  |  |  |  |  |  | o..szapc | o..szapc |  |  | Compare Two Operands |
| CMP | r/m8 | imm8 |  |  |  |  |  | 82 |  | 7 |  |  |  |  |  |  | o..szapc | o..szapc |  |  | Compare Two Operands |
| CMP | r/m16/32 | imm8 |  |  |  |  |  | 83 |  | 7 |  |  |  |  |  |  | o..szapc | o..szapc |  |  | Compare Two Operands |
| CMPPD | **xmm** | xmm/m128 | imm8 |  | sse2 | 66 | 0F | C2 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Compare Packed Double-FP Values |
| CMPPS | **xmm** | xmm/m128 | imm8 |  | sse1 |  | 0F | C2 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Compare Packed Single-FP Values |
| CMPS | m8 | m8 |  |  |  |  |  | A6 |  |  |  |  |  |  |  | .d...... | o..szapc | o..szapc |  |  | Compare String Operands |
| CMPSB | *m8* | *m8* |  |  |
| CMPS | m16 | m16 |  |  |  |  |  | A7 |  |  |  |  |  |  |  | .d...... | o..szapc | o..szapc |  |  | Compare String Operands |
| CMPSW | *m16* | *m16* |  |  |
| CMPS | m16/32 | m16/32 |  |  |  |  |  | A7 |  |  | 03+ |  |  |  |  | .d...... | o..szapc | o..szapc |  |  | Compare String Operands |
| CMPSD | *m32* | *m32* |  |  |
| CMPSD | **xmm** | xmm/m64 | imm8 |  | sse2 | F2 | 0F | C2 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Compare Scalar Double-FP Values |
| CMPSS | **xmm** | xmm/m32 | imm8 |  | sse1 | F3 | 0F | C2 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Compare Scalar Single-FP Values |
| CMPXCHG | **r/m8** | ***AL*** | r8 |  |  |  | 0F | B0 |  | r | 04+ | D[18](http://ref.x86asm.net/coder32-abc.html#gen_note_CMPXCHG_0FB0_0FB1) |  |  | L |  | o..szapc | o..szapc |  |  | Compare and Exchange |
| CMPXCHG | **r/m16/32** | ***eAX*** | r16/32 |  |  |  | 0F | B1 |  | r | 04+ | D[18](http://ref.x86asm.net/coder32-abc.html#gen_note_CMPXCHG_0FB0_0FB1) |  |  | L |  | o..szapc | o..szapc |  |  | Compare and Exchange |
| CMPXCHG8B | **m64** | ***EAX*** | ***EDX*** | ... |  |  | 0F | C7 |  | 1 | P1+ | D[20](http://ref.x86asm.net/coder32-abc.html#gen_note_CMPXCHG8B_CMPXCHG16B_0FC7_1) |  |  | L |  | ....z... | ....z... |  |  | Compare and Exchange Bytes |
| COMISD | xmm | xmm/m64 |  |  | sse2 | 66 | 0F | 2F |  | r | P4+ |  |  |  |  |  | ....z.pc | ....z.pc |  |  | Compare Scalar Ordered Double-FP Values and Set EFLAGS |
| COMISS | xmm | xmm/m32 |  |  | sse1 |  | 0F | 2F |  | r | P3+ |  |  |  |  |  | ....z.pc | ....z.pc |  |  | Compare Scalar Ordered Single-FP Values and Set EFLAGS |
| CPUID | ***IA32\_BIOS\_…*** | ***EAX*** | ***ECX*** | ... |  |  | 0F | A2 |  |  | 04++ |  |  |  |  |  |  |  |  |  | CPU Identification |
| CRC32 | **r32** | r/m8 |  |  | sse42 | F2 | 0F | 38 | F0 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Accumulate CRC32 Value |
| CRC32 | **r32** | r/m16/32 |  |  | sse42 | F2 | 0F | 38 | F1 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Accumulate CRC32 Value |
| CS | *CS* |  |  |  |  | 2E |  |  |  |  |  |  |  |  |  |  |  |  |  |  | CS segment override prefix |
| CVTDQ2PD | **xmm** | xmm/m128 |  |  | sse2 | F3 | 0F | E6 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Convert Packed DW Integers to Double-FP Values |
| CVTDQ2PS | **xmm** | xmm/m128 |  |  | sse2 |  | 0F | 5B |  | r | P4+ |  |  |  |  |  |  |  |  |  | Convert Packed DW Integers to Single-FP Values |
| CVTPD2DQ | **xmm** | xmm/m128 |  |  | sse2 | F2 | 0F | E6 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Convert Packed Double-FP Values to DW Integers |
| CVTPD2PI | **mm** | xmm/m128 |  |  | sse2 | 66 | 0F | 2D |  | r | P4+ |  |  |  |  |  |  |  |  |  | Convert Packed Double-FP Values to DW Integers |
| CVTPD2PS | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 5A |  | r | P4+ |  |  |  |  |  |  |  |  |  | Convert Packed Double-FP Values to Single-FP Values |
| CVTPI2PD | **xmm** | mm/m64 |  |  | sse2 | 66 | 0F | 2A |  | r | P4+ |  |  |  |  |  |  |  |  |  | Convert Packed DW Integers to Double-FP Values |
| CVTPI2PS | **xmm** | mm/m64 |  |  | sse1 |  | 0F | 2A |  | r | P3+ |  |  |  |  |  |  |  |  |  | Convert Packed DW Integers to Single-FP Values |
| CVTPS2DQ | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 5B |  | r | P4+ |  |  |  |  |  |  |  |  |  | Convert Packed Single-FP Values to DW Integers |
| CVTPS2PD | **xmm** | xmm/m128 |  |  | sse2 |  | 0F | 5A |  | r | P4+ |  |  |  |  |  |  |  |  |  | Convert Packed Single-FP Values to Double-FP Values |
| CVTPS2PI | **mm** | xmm/m64 |  |  | sse1 |  | 0F | 2D |  | r | P3+ |  |  |  |  |  |  |  |  |  | Convert Packed Single-FP Values to DW Integers |
| CVTSD2SI | **r32** | xmm/m64 |  |  | sse2 | F2 | 0F | 2D |  | r | P4+ |  |  |  |  |  |  |  |  |  | Convert Scalar Double-FP Value to DW Integer |
| CVTSD2SS | **xmm** | xmm/m64 |  |  | sse2 | F2 | 0F | 5A |  | r | P4+ |  |  |  |  |  |  |  |  |  | Convert Scalar Double-FP Value to Scalar Single-FP Value |
| CVTSI2SD | **xmm** | r/m32 |  |  | sse2 | F2 | 0F | 2A |  | r | P4+ |  |  |  |  |  |  |  |  |  | Convert DW Integer to Scalar Double-FP Value |
| CVTSI2SS | **xmm** | r/m32 |  |  | sse1 | F3 | 0F | 2A |  | r | P3+ |  |  |  |  |  |  |  |  |  | Convert DW Integer to Scalar Single-FP Value |
| CVTSS2SD | **xmm** | xmm/m32 |  |  | sse2 | F3 | 0F | 5A |  | r | P4+ |  |  |  |  |  |  |  |  |  | Convert Scalar Single-FP Value to Scalar Double-FP Value |
| CVTSS2SI | **r32** | xmm/m32 |  |  | sse1 | F3 | 0F | 2D |  | r | P3+ |  |  |  |  |  |  |  |  |  | Convert Scalar Single-FP Value to DW Integer |
| CVTTPD2DQ | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | E6 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Convert with Trunc. Packed Double-FP Values to DW Integers |
| CVTTPD2PI | **mm** | xmm/m128 |  |  | sse2 | 66 | 0F | 2C |  | r | P4+ |  |  |  |  |  |  |  |  |  | Convert with Trunc. Packed Double-FP Values to DW Integers |
| CVTTPS2DQ | **xmm** | xmm/m128 |  |  | sse2 | F3 | 0F | 5B |  | r | P4+ |  |  |  |  |  |  |  |  |  | Convert with Trunc. Packed Single-FP Values to DW Integers |
| CVTTPS2PI | **mm** | xmm/m64 |  |  | sse1 |  | 0F | 2C |  | r | P3+ |  |  |  |  |  |  |  |  |  | Convert with Trunc. Packed Single-FP Values to DW Integers |
| CVTTSD2SI | **r32** | xmm/m64 |  |  | sse2 | F2 | 0F | 2C |  | r | P4+ |  |  |  |  |  |  |  |  |  | Conv. with Trunc. Scalar Double-FP Value to Signed DW Int |
| CVTTSS2SI | **r32** | xmm/m32 |  |  | sse1 | F3 | 0F | 2C |  | r | P3+ |  |  |  |  |  |  |  |  |  | Convert with Trunc. Scalar Single-FP Value to DW Integer |
| CWD | ***DX*** | *AX* |  |  |  |  |  | 99 |  |  |  |  |  |  |  |  |  |  |  |  | Convert Word to Doubleword |
| CWDE | ***EAX*** | *AX* |  |  |  |  |  | 98 |  |  | 03+ |  |  |  |  |  |  |  |  |  | Convert Word to Doubleword |
| DAA | ***AL*** |  |  |  |  |  |  | 27 |  |  |  |  |  |  |  | .....a.c | o..szapc | ...szapc | o....... |  | Decimal Adjust AL after Addition |
| DAS | ***AL*** |  |  |  |  |  |  | 2F |  |  |  |  |  |  |  | .....a.c | o..szapc | ...szapc | o....... |  | Decimal Adjust AL after Subtraction |
| DEC | **r16/32** |  |  |  |  |  |  | 48+r | |  |  |  |  |  |  |  | o..szap. | o..szap. |  |  | Decrement by 1 |
| DEC | **r/m8** |  |  |  |  |  |  | FE |  | 1 |  |  |  |  | L |  | o..szap. | o..szap. |  |  | Decrement by 1 |
| DEC | **r/m16/32** |  |  |  |  |  |  | FF |  | 1 |  |  |  |  | L |  | o..szap. | o..szap. |  |  | Decrement by 1 |
| DIV | ***AL*** | ***AH*** | *AX* | r/m8 |  |  |  | F6 |  | 6 |  |  |  |  |  |  | o..szapc |  | o..szapc |  | Unsigned Divide |
| DIV | ***eDX*** | ***eAX*** | r/m16/32 |  |  |  |  | F7 |  | 6 |  |  |  |  |  |  | o..szapc |  | o..szapc |  | Unsigned Divide |
| DIVPD | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 5E |  | r | P4+ |  |  |  |  |  |  |  |  |  | Divide Packed Double-FP Values |
| DIVPS | **xmm** | xmm/m128 |  |  | sse1 |  | 0F | 5E |  | r | P3+ |  |  |  |  |  |  |  |  |  | Divide Packed Single-FP Values |
| DIVSD | **xmm** | xmm/m64 |  |  | sse2 | F2 | 0F | 5E |  | r | P4+ |  |  |  |  |  |  |  |  |  | Divide Scalar Double-FP Values |
| DIVSS | **xmm** | xmm/m32 |  |  | sse1 | F3 | 0F | 5E |  | r | P3+ |  |  |  |  |  |  |  |  |  | Divide Scalar Single-FP Values |
| DPPD | **xmm** | xmm/m128 |  |  | sse41 | 66 | 0F | 3A | 41 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Dot Product of Packed Double-FP Values |
| DPPS | **xmm** | xmm/m128 |  |  | sse41 | 66 | 0F | 3A | 40 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Dot Product of Packed Single-FP Values |
| DS | *DS* |  |  |  |  | 3E |  |  |  |  |  |  |  |  |  |  |  |  |  |  | DS segment override prefix |
| EMMS |  |  |  |  | mmx |  | 0F | 77 |  |  | PX+ |  |  |  |  |  |  |  |  |  | Empty MMX Technology State |
| ENTER | ***eBP*** | imm16 | imm8 |  |  |  |  | C8 |  |  | 01+ |  |  |  |  |  |  |  |  |  | Make Stack Frame for Procedure Parameters |
| ES | *ES* |  |  |  |  | 26 |  |  |  |  |  |  |  |  |  |  |  |  |  |  | ES segment override prefix |
| EXTRACTPS | **r/m32** | xmm | imm8 |  | sse41 | 66 | 0F | 3A | 17 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Extract Packed Single-FP Value |
| F2XM1 | ***ST*** |  |  |  |  |  |  | D9 | F0 | 6 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Compute 2x-1 |
| FABS | ***ST*** |  |  |  |  |  |  | D9 | E1 | 4 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Absolute Value |
| FADD | ***ST*** | m32real |  |  |  |  |  | D8 |  | 0 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Add |
| FADD | **ST** | STi |  |  |
| FADD | ***ST*** | m64real |  |  |  |  |  | DC |  | 0 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Add |
| FADD | **STi** | ST |  |  |  |  |  | DC |  | 0 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Add |
| FADDP | **STi** | ST |  |  |  |  |  | DE |  | 0 |  |  |  |  | p |  | 0123 | .1.. | 0.23 |  | Add and Pop |
| FADDP | ***ST1*** | *ST* |  |  |  |  |  | DE | C1 | 0 |  |  |  |  | p |  | 0123 | .1.. | 0.23 |  | Add and Pop |
| FBLD | ***ST*** | m80dec |  |  |  |  |  | DF |  | 4 |  |  |  |  | s |  | 0123 | .1.. | 0.23 |  | Load Binary Coded Decimal |
| FBSTP | **m80dec** | *ST* |  |  |  |  |  | DF |  | 6 |  |  |  |  | p |  | 0123 | .1.. | 0.23 |  | Store BCD Integer and Pop |
| FCHS | ***ST*** |  |  |  |  |  |  | D9 | E0 | 4 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Change Sign |
| FCLEX |  |  |  |  |  | 9B |  | DB | E2 | 4 |  |  |  |  |  |  | 0123 |  | 0123 |  | Clear Exceptions |
| FCMOVB | **ST** | STi |  |  |  |  |  | DA |  | 0 | PP+ |  |  |  |  | .......c | 0123 | .1.. | 0.23 |  | FP Conditional Move - below (CF=1) |
| FCMOVBE | **ST** | STi |  |  |  |  |  | DA |  | 2 | PP+ |  |  |  |  | ....z... | 0123 | .1.. | 0.23 |  | FP Conditional Move - below or equal (CF=1 or ZF=1) |
| FCMOVE | **ST** | STi |  |  |  |  |  | DA |  | 1 | PP+ |  |  |  |  | ....z... | 0123 | .1.. | 0.23 |  | FP Conditional Move - equal (ZF=1) |
| FCMOVNB | **ST** | STi |  |  |  |  |  | DB |  | 0 | PP+ |  |  |  |  | .......c | 0123 | .1.. | 0.23 |  | FP Conditional Move - not below (CF=0) |
| FCMOVNBE | **ST** | STi |  |  |  |  |  | DB |  | 2 | PP+ |  |  |  |  | ....z... | 0123 | .1.. | 0.23 |  | FP Conditional Move - below or equal (CF=0 and ZF=0) |
| FCMOVNE | **ST** | STi |  |  |  |  |  | DB |  | 1 | PP+ |  |  |  |  | ....z... | 0123 | .1.. | 0.23 |  | FP Conditional Move - not equal (ZF=0) |
| FCMOVNU | **ST** | STi |  |  |  |  |  | DB |  | 3 | PP+ |  |  |  |  | ......p. | 0123 | .1.. | 0.23 |  | FP Conditional Move - not unordered (PF=0) |
| FCMOVU | **ST** | STi |  |  |  |  |  | DA |  | 3 | PP+ |  |  |  |  | ......p. | 0123 | .1.. | 0.23 |  | FP Conditional Move - unordered (PF=1) |
| FCOM | *ST* | STi/m32real |  |  |  |  |  | D8 |  | 2 |  |  |  |  |  |  | 0123 | 0123 |  |  | Compare Real |
| FCOM | *ST* | *ST1* |  |  |  |  |  | D8 | D1 | 2 |  |  |  |  |  |  | 0123 | 0123 |  |  | Compare Real |
| FCOM | *ST* | m64real |  |  |  |  |  | DC |  | 2 |  |  |  |  |  |  | 0123 | 0123 |  |  | Compare Real |
| *FCOM2* | *ST* | STi |  |  |  |  |  | DC |  | 2 | 03+ | U[8](http://ref.x86asm.net/coder32-abc.html#gen_note_x87_fpu_undoc_aliases) |  |  |  |  | 0123 | 0123 |  |  | Compare Real |
| FCOMI | ST | STi |  |  |  |  |  | DB |  | 6 | PP+ |  |  |  |  |  | o...z.pc .1.. | o...z.pc .1.. |  | o....... | Compare Floating Point Values and Set EFLAGS |
| FCOMIP | ST | STi |  |  |  |  |  | DF |  | 6 | PP+ |  |  |  | p |  | o...z.pc .1.. | o...z.pc .1.. |  | o....... | Compare Floating Point Values and Set EFLAGS and Pop |
| FCOMP | *ST* | STi/m32real |  |  |  |  |  | D8 |  | 3 |  |  |  |  | p |  | 0123 | 0123 |  |  | Compare Real and Pop |
| FCOMP | *ST* | *ST1* |  |  |  |  |  | D8 | D9 | 3 |  |  |  |  | p |  | 0123 | 0123 |  |  | Compare Real and Pop |
| FCOMP | *ST* | m64real |  |  |  |  |  | DC |  | 3 |  |  |  |  | p |  | 0123 | 0123 |  |  | Compare Real and Pop |
| *FCOMP3* | *ST* | STi |  |  |  |  |  | DC |  | 3 | 03+ | U[8](http://ref.x86asm.net/coder32-abc.html#gen_note_x87_fpu_undoc_aliases) |  |  | p |  | 0123 | 0123 |  |  | Compare Real and Pop |
| *FCOMP5* | *ST* | STi |  |  |  |  |  | DE |  | 2 | 03+ | U[8](http://ref.x86asm.net/coder32-abc.html#gen_note_x87_fpu_undoc_aliases) |  |  | p |  | 0123 | 0123 |  |  | Compare Real and Pop |
| FCOMPP | *ST* | *ST1* |  |  |  |  |  | DE | D9 | 3 |  |  |  |  | P |  | 0123 | 0123 |  |  | Compare Real and Pop Twice |
| FCOS | ***ST*** |  |  |  |  |  |  | D9 | FF | 7 |  |  |  |  |  |  | 0123 | .12. | 0..3 |  | Cosine |
| FDECSTP |  |  |  |  |  |  |  | D9 | F6 | 6 |  |  |  |  |  |  | 0123 | .1.. | 0.23 | .0.. | Decrement Stack-Top Pointer |
| FDIV | ***ST*** | m32real |  |  |  |  |  | D8 |  | 6 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Divide |
| FDIV | **ST** | STi |  |  |
| FDIV | ***ST*** | m64real |  |  |  |  |  | DC |  | 6 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Divide |
| FDIV | **STi** | ST |  |  |  |  |  | DC |  | 7 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Divide and Pop |
| FDIVP | **STi** | ST |  |  |  |  |  | DE |  | 7 |  |  |  |  | p |  | 0123 | .1.. | 0.23 |  | Divide and Pop |
| FDIVP | ***ST1*** | *ST* |  |  |  |  |  | DE | F9 | 7 |  |  |  |  | p |  | 0123 | .1.. | 0.23 |  | Divide and Pop |
| FDIVR | ***ST*** | m32real |  |  |  |  |  | D8 |  | 7 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Reverse Divide |
| FDIVR | **ST** | STi |  |  |
| FDIVR | **STi** | ST |  |  |  |  |  | DC |  | 6 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Reverse Divide |
| FDIVR | ***ST*** | m64real |  |  |  |  |  | DC |  | 7 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Reverse Divide |
| FDIVRP | **STi** | ST |  |  |  |  |  | DE |  | 6 |  |  |  |  | p |  | 0123 | .1.. | 0.23 |  | Reverse Divide and Pop |
| FDIVRP | ***ST1*** | *ST* |  |  |  |  |  | DE | F1 | 6 |  |  |  |  | p |  | 0123 | .1.. | 0.23 |  | Reverse Divide and Pop |
| FFREE | STi |  |  |  |  |  |  | DD |  | 0 |  |  |  |  |  |  | 0123 |  | 0123 |  | Free Floating-Point Register |
| *FFREEP* | STi |  |  |  |  |  |  | DF |  | 0 |  | D[7](http://ref.x86asm.net/coder32-abc.html#gen_note_FFREEP_DF_1) |  |  | p |  | 0123 |  | 0123 |  | Free Floating-Point Register and Pop |
| FIADD | ***ST*** | m32int |  |  |  |  |  | DA |  | 0 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Add |
| FIADD | ***ST*** | m16int |  |  |  |  |  | DE |  | 0 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Add |
| FICOM | *ST* | m32int |  |  |  |  |  | DA |  | 2 |  |  |  |  |  |  | 0123 | 0123 |  |  | Compare Integer |
| FICOM | *ST* | m16int |  |  |  |  |  | DE |  | 2 |  |  |  |  |  |  | 0123 | 0123 |  |  | Compare Integer |
| FICOMP | *ST* | m32int |  |  |  |  |  | DA |  | 3 |  |  |  |  | p |  | 0123 | 0123 |  |  | Compare Integer and Pop |
| FICOMP | *ST* | m16int |  |  |  |  |  | DE |  | 3 |  |  |  |  | p |  | 0123 | 0123 |  |  | Compare Integer and Pop |
| FIDIV | ***ST*** | m32int |  |  |  |  |  | DA |  | 6 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Divide |
| FIDIV | ***ST*** | m16int |  |  |  |  |  | DE |  | 6 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Divide |
| FIDIVR | ***ST*** | m32int |  |  |  |  |  | DA |  | 7 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Reverse Divide |
| FIDIVR | ***ST*** | m16int |  |  |  |  |  | DE |  | 7 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Reverse Divide |
| FILD | ***ST*** | m32int |  |  |  |  |  | DB |  | 0 |  |  |  |  | s |  | 0123 | .1.. | 0.23 |  | Load Integer |
| FILD | ***ST*** | m16int |  |  |  |  |  | DF |  | 0 |  |  |  |  | s |  | 0123 | .1.. | 0.23 |  | Load Integer |
| FILD | ***ST*** | m64int |  |  |  |  |  | DF |  | 5 |  |  |  |  | s |  | 0123 | .1.. | 0.23 |  | Load Integer |
| FIMUL | ***ST*** | m32int |  |  |  |  |  | DA |  | 1 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Multiply |
| FIMUL | ***ST*** | m16int |  |  |  |  |  | DE |  | 1 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Multiply |
| FINCSTP |  |  |  |  |  |  |  | D9 | F7 | 6 |  |  |  |  |  |  | 0123 | .1.. | 0.23 | .0.. | Increment Stack-Top Pointer |
| FINIT |  |  |  |  |  | 9B |  | DB | E3 | 4 |  |  |  |  |  |  | 0123 |  |  | 0000 | Initialize Floating-Point Unit |
| FIST | **m32int** | *ST* |  |  |  |  |  | DB |  | 2 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Store Integer |
| FIST | **m16int** | *ST* |  |  |  |  |  | DF |  | 2 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Store Integer |
| FISTP | **m32int** | *ST* |  |  |  |  |  | DB |  | 3 |  |  |  |  | p |  | 0123 | .1.. | 0.23 |  | Store Integer and Pop |
| FISTP | **m16int** | *ST* |  |  |  |  |  | DF |  | 3 |  |  |  |  | p |  | 0123 | .1.. | 0.23 |  | Store Integer and Pop |
| FISTP | **m64int** | *ST* |  |  |  |  |  | DF |  | 7 |  |  |  |  | p |  | 0123 | .1.. | 0.23 |  | Store Integer and Pop |
| FISTTP | **m32int** | *ST* |  |  | sse3 |  |  | DB |  | 1 | P4++ |  |  |  | p |  | 0123 | .1.. | 0.23 | .0.. | Store Integer with Truncation and Pop |
| FISTTP | **m64int** | *ST* |  |  | sse3 |  |  | DD |  | 1 | P4++ |  |  |  | p |  | 0123 | .1.. | 0.23 | .0.. | Store Integer with Truncation and Pop |
| FISTTP | **m16int** | *ST* |  |  | sse3 |  |  | DF |  | 1 | P4++ |  |  |  | p |  | 0123 | .1.. | 0.23 | .0.. | Store Integer with Truncation and Pop |
| FISUB | ***ST*** | m32int |  |  |  |  |  | DA |  | 4 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Subtract |
| FISUB | ***ST*** | m16int |  |  |  |  |  | DE |  | 4 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Subtract |
| FISUBR | ***ST*** | m32int |  |  |  |  |  | DA |  | 5 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Reverse Subtract |
| FISUBR | ***ST*** | m16int |  |  |  |  |  | DE |  | 5 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Reverse Subtract |
| FLD | ***ST*** | STi/m32real |  |  |  |  |  | D9 |  | 0 |  |  |  |  | s |  | 0123 | .1.. | 0.23 |  | Load Floating Point Value |
| FLD | ***ST*** | m80real |  |  |  |  |  | DB |  | 5 |  |  |  |  | s |  | 0123 | .1.. | 0.23 |  | Load Floating Point Value |
| FLD | ***ST*** | m64real |  |  |  |  |  | DD |  | 0 |  |  |  |  | s |  | 0123 | .1.. | 0.23 |  | Load Floating Point Value |
| FLD1 | ***ST*** |  |  |  |  |  |  | D9 | E8 | 5 |  |  |  |  | s |  | 0123 | .1.. | 0.23 |  | Load Constant +1.0 |
| FLDCW | m16 |  |  |  |  |  |  | D9 |  | 5 |  |  |  |  |  |  | 0123 |  | 0123 |  | Load x87 FPU Control Word |
| FLDENV | m14/28 |  |  |  |  |  |  | D9 |  | 4 |  |  |  |  |  |  | 0123 | 0123 |  |  | Load x87 FPU Environment |
| FLDL2E | ***ST*** |  |  |  |  |  |  | D9 | EA | 5 |  |  |  |  | s |  | 0123 | .1.. | 0.23 |  | Load Constant log2e |
| FLDL2T | ***ST*** |  |  |  |  |  |  | D9 | E9 | 5 |  |  |  |  | s |  | 0123 | .1.. | 0.23 |  | Load Constant log210 |
| FLDLG2 | ***ST*** |  |  |  |  |  |  | D9 | EC | 5 |  |  |  |  | s |  | 0123 | .1.. | 0.23 |  | Load Constant log102 |
| FLDLN2 | ***ST*** |  |  |  |  |  |  | D9 | ED | 5 |  |  |  |  | s |  | 0123 | .1.. | 0.23 |  | Load Constant loge2 |
| FLDPI | ***ST*** |  |  |  |  |  |  | D9 | EB | 5 |  |  |  |  | s |  | 0123 | .1.. | 0.23 |  | Load Constant π |
| FLDZ | ***ST*** |  |  |  |  |  |  | D9 | EE | 5 |  |  |  |  | s |  | 0123 | .1.. | 0.23 |  | Load Constant +0.0 |
| FMUL | ***ST*** | m32real |  |  |  |  |  | D8 |  | 1 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Multiply |
| FMUL | **ST** | STi |  |  |
| FMUL | ***ST*** | m64real |  |  |  |  |  | DC |  | 1 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Multiply |
| FMUL | **STi** | ST |  |  |  |  |  | DC |  | 1 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Multiply |
| FMULP | **STi** | ST |  |  |  |  |  | DE |  | 1 |  |  |  |  | p |  | 0123 | .1.. | 0.23 |  | Multiply and Pop |
| FMULP | ***ST1*** | *ST* |  |  |  |  |  | DE | C9 | 1 |  |  |  |  | p |  | 0123 | .1.. | 0.23 |  | Multiply and Pop |
| FNCLEX |  |  |  |  |  |  |  | DB | E2 | 4 |  |  |  |  |  |  | 0123 |  | 0123 |  | Clear Exceptions |
| FNDISI*nop* |  |  |  |  |  |  |  | DB | E1 | 4 | 01+ | D[5](http://ref.x86asm.net/coder32-abc.html#gen_note_FNENI_DBE0_FNDISI_DBE1) |  |  |  |  |  |  |  |  | Treated as Integer NOP |
| FNENI*nop* |  |  |  |  |  |  |  | DB | E0 | 4 | 01+ | D[5](http://ref.x86asm.net/coder32-abc.html#gen_note_FNENI_DBE0_FNDISI_DBE1) |  |  |  |  |  |  |  |  | Treated as Integer NOP |
| FNINIT |  |  |  |  |  |  |  | DB | E3 | 4 |  |  |  |  |  |  | 0123 |  |  | 0000 | Initialize Floating-Point Unit |
| FNOP |  |  |  |  |  |  |  | D9 | D0 | 2 |  |  |  |  |  |  | 0123 |  | 0123 |  | No Operation |
| FNSAVE | **m94/108** | *ST* | *ST1* | ... |  |  |  | DD |  | 6 |  |  |  |  |  |  | 0123 | 0123 |  | 0000 | Store x87 FPU State |
| FNSETPM*nop* |  |  |  |  |  |  |  | DB | E4 | 4 | 03+ | D[6](http://ref.x86asm.net/coder32-abc.html#gen_note_FNSETPM_DBE4) |  |  |  |  |  |  |  |  | Treated as Integer NOP |
| FNSTCW | **m16** |  |  |  |  |  |  | D9 |  | 7 |  |  |  |  |  |  | 0123 |  | 0123 |  | Store x87 FPU Control Word |
| FNSTENV | **m14/28** |  |  |  |  |  |  | D9 |  | 6 |  |  |  |  |  |  | 0123 |  | 0123 |  | Store x87 FPU Environment |
| FNSTSW | **m16** |  |  |  |  |  |  | DD |  | 7 |  |  |  |  |  |  | 0123 |  | 0123 |  | Store x87 FPU Status Word |
| FNSTSW | **AX** |  |  |  |  |  |  | DF | E0 | 4 | 02+ |  |  |  |  |  | 0123 |  | 0123 |  | Store x87 FPU Status Word |
| FPATAN | ***ST1*** | *ST* |  |  |  |  |  | D9 | F3 | 6 |  |  |  |  | p |  | 0123 | .1.. | 0.23 |  | Partial Arctangent and Pop |
| FPREM | ***ST*** | *ST1* |  |  |  |  |  | D9 | F8 | 7 |  |  |  |  |  |  | 0123 | 0123 |  |  | Partial Remainder (for compatibility with i8087 and i287) |
| FPREM1 | ***ST*** | *ST1* |  |  |  |  |  | D9 | F5 | 6 |  |  |  |  |  |  | 0123 | 0123 |  |  | IEEE Partial Remainder |
| FPTAN | ***ST*** |  |  |  |  |  |  | D9 | F2 | 6 |  |  |  |  | s |  | 0123 | .12. | 0..3 |  | Partial Tangent |
| FRNDINT | ***ST*** |  |  |  |  |  |  | D9 | FC | 7 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Round to Integer |
| FRSTOR | ***ST*** | ***ST1*** | ***ST2*** | ... |  |  |  | DD |  | 4 |  |  |  |  |  |  | 0123 | 0123 |  |  | Restore x87 FPU State |
| FS | *FS* |  |  |  |  | 64 |  |  |  |  | 03+ |  |  |  |  |  |  |  |  |  | FS segment override prefix |
| FSAVE | **m94/108** | *ST* | *ST1* | ... |  | 9B |  | DD |  | 6 |  |  |  |  |  |  | 0123 | 0123 |  | 0000 | Store x87 FPU State |
| FSCALE | ***ST*** | *ST1* |  |  |  |  |  | D9 | FD | 7 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Scale |
| FSIN | ***ST*** |  |  |  |  |  |  | D9 | FE | 7 |  |  |  |  |  |  | 0123 | .12. | 0..3 |  | Sine |
| FSINCOS | ***ST*** |  |  |  |  |  |  | D9 | FB | 7 |  |  |  |  | s |  | 0123 | .12. | 0..3 |  | Sine and Cosine |
| FSQRT | ***ST*** |  |  |  |  |  |  | D9 | FA | 7 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Square Root |
| FST | **m32real** | *ST* |  |  |  |  |  | D9 |  | 2 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Store Floating Point Value |
| FST | **m64real** | *ST* |  |  |  |  |  | DD |  | 2 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Store Floating Point Value |
| FST | ***ST*** | STi |  |  |  |  |  | DD |  | 2 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Store Floating Point Value |
| FSTCW | **m16** |  |  |  |  | 9B |  | D9 |  | 7 |  |  |  |  |  |  | 0123 |  | 0123 |  | Store x87 FPU Control Word |
| FSTENV | **m14/28** |  |  |  |  | 9B |  | D9 |  | 6 |  |  |  |  |  |  | 0123 |  | 0123 |  | Store x87 FPU Environment |
| FSTP | **m32real** | *ST* |  |  |  |  |  | D9 |  | 3 |  |  |  |  | p |  | 0123 | .1.. | 0.23 |  | Store Floating Point Value and Pop |
| FSTP | **m80real** | *ST* |  |  |  |  |  | DB |  | 7 |  |  |  |  | p |  | 0123 | .1.. | 0.23 |  | Store Floating Point Value and Pop |
| FSTP | **m64real** | *ST* |  |  |  |  |  | DD |  | 3 |  |  |  |  | p |  | 0123 | .1.. | 0.23 |  | Store Floating Point Value and Pop |
| FSTP | ***ST*** | STi |  |  |  |  |  | DD |  | 3 |  |  |  |  | p |  | 0123 | .1.. | 0.23 |  | Store Floating Point Value and Pop |
| *FSTP1* | **STi** | *ST* |  |  |  |  |  | D9 |  | 3 | 03+ | U[8](http://ref.x86asm.net/coder32-abc.html#gen_note_x87_fpu_undoc_aliases) |  |  | p |  | 0123 | .1.. | 0.23 |  | Store Floating Point Value and Pop |
| *FSTP8* | **STi** | *ST* |  |  |  |  |  | DF |  | 2 | 03+ | U[8](http://ref.x86asm.net/coder32-abc.html#gen_note_x87_fpu_undoc_aliases) |  |  | p |  | 0123 | .1.. | 0.23 |  | Store Floating Point Value and Pop |
| *FSTP9* | **STi** | *ST* |  |  |  |  |  | DF |  | 3 | 03+ | U[8](http://ref.x86asm.net/coder32-abc.html#gen_note_x87_fpu_undoc_aliases) |  |  | p |  | 0123 | .1.. | 0.23 |  | Store Floating Point Value and Pop |
| FSTSW | **m16** |  |  |  |  | 9B |  | DD |  | 7 |  |  |  |  |  |  | 0123 |  | 0123 |  | Store x87 FPU Status Word |
| FSTSW | **AX** |  |  |  |  | 9B |  | DF | E0 | 4 | 02+ |  |  |  |  |  | 0123 |  | 0123 |  | Store x87 FPU Status Word |
| FSUB | ***ST*** | m32real |  |  |  |  |  | D8 |  | 4 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Subtract |
| FSUB | **ST** | STi |  |  |
| FSUB | ***ST*** | m64real |  |  |  |  |  | DC |  | 4 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Subtract |
| FSUB | **STi** | ST |  |  |  |  |  | DC |  | 5 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Subtract |
| FSUBP | **STi** | ST |  |  |  |  |  | DE |  | 5 |  |  |  |  | p |  | 0123 | .1.. | 0.23 |  | Subtract and Pop |
| FSUBP | ***ST1*** | *ST* |  |  |  |  |  | DE | E9 | 5 |  |  |  |  | p |  | 0123 | .1.. | 0.23 |  | Subtract and Pop |
| FSUBR | ***ST*** | m32real |  |  |  |  |  | D8 |  | 5 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Reverse Subtract |
| FSUBR | **ST** | STi |  |  |
| FSUBR | **STi** | ST |  |  |  |  |  | DC |  | 4 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Reverse Subtract |
| FSUBR | ***ST*** | m64real |  |  |  |  |  | DC |  | 5 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Reverse Subtract |
| FSUBRP | **STi** | ST |  |  |  |  |  | DE |  | 4 |  |  |  |  | p |  | 0123 | .1.. | 0.23 |  | Reverse Subtract and Pop |
| FSUBRP | ***ST1*** | *ST* |  |  |  |  |  | DE | E1 | 4 |  |  |  |  | p |  | 0123 | .1.. | 0.23 |  | Reverse Subtract and Pop |
| FTST | *ST* |  |  |  |  |  |  | D9 | E4 | 4 |  |  |  |  |  |  | 0123 | 0123 |  |  | Test |
| FUCOM | *ST* | STi |  |  |  |  |  | DD |  | 4 | 03+ |  |  |  |  |  | 0123 | 0123 |  |  | Unordered Compare Floating Point Values |
| FUCOM | *ST* | *ST1* |  |  |  |  |  | DD | E1 | 4 | 03+ |  |  |  |  |  | 0123 | 0123 |  |  | Unordered Compare Floating Point Values |
| FUCOMI | ST | STi |  |  |  |  |  | DB |  | 5 | PP+ |  |  |  |  |  | o...z.pc .1.. | o...z.pc .1.. |  | o....... | Unordered Compare Floating Point Values and Set EFLAGS |
| FUCOMIP | ST | STi |  |  |  |  |  | DF |  | 5 | PP+ |  |  |  | p |  | o...z.pc .1.. | o...z.pc .1.. |  | o....... | Unordered Compare Floating Point Values and Set EFLAGS and Pop |
| FUCOMP | *ST* | STi |  |  |  |  |  | DD |  | 5 | 03+ |  |  |  | p |  | 0123 | 0123 |  |  | Unordered Compare Floating Point Values and Pop |
| FUCOMP | *ST* | *ST1* |  |  |  |  |  | DD | E9 | 5 | 03+ |  |  |  | p |  | 0123 | 0123 |  |  | Unordered Compare Floating Point Values and Pop |
| FUCOMPP | *ST* | *ST1* |  |  |  |  |  | DA | E9 | 5 | 03+ |  |  |  | P |  | 0123 | 0123 |  |  | Unordered Compare Floating Point Values and Pop Twice |
| FWAIT |  |  |  |  |  |  |  | 9B |  |  |  |  |  |  |  |  | 0123 |  | 0123 |  | Check pending unmasked floating-point exceptions |
| WAIT |  |  |  |  |
| FXAM | *ST* |  |  |  |  |  |  | D9 | E5 | 4 |  |  |  |  |  |  | 0123 | 0123 |  |  | Examine |
| FXCH | ***ST*** | **STi** |  |  |  |  |  | D9 |  | 1 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Exchange Register Contents |
| FXCH | ***ST*** | ***ST1*** |  |  |  |  |  | D9 | C9 | 1 |  |  |  |  |  |  | 0123 | .1.. | 0.23 |  | Exchange Register Contents |
| *FXCH4* | ***ST*** | **STi** |  |  |  |  |  | DD |  | 1 | 03+ | U[8](http://ref.x86asm.net/coder32-abc.html#gen_note_x87_fpu_undoc_aliases) |  |  |  |  | 0123 | .1.. | 0.23 |  | Exchange Register Contents |
| *FXCH7* | ***ST*** | **STi** |  |  |  |  |  | DF |  | 1 | 03+ | U[8](http://ref.x86asm.net/coder32-abc.html#gen_note_x87_fpu_undoc_aliases) |  |  |  |  | 0123 | .1.. | 0.23 |  | Exchange Register Contents |
| FXRSTOR | ***ST*** | ***ST1*** | ***ST2*** | ... |  |  | 0F | AE |  | 1 | P2++ |  |  |  |  |  |  |  |  |  | Restore x87 FPU, MMX, XMM, and MXCSR State |
| FXSAVE | **m512** | *ST* | *ST1* | ... |  |  | 0F | AE |  | 0 | P2++ |  |  |  |  |  |  |  |  |  | Save x87 FPU, MMX, XMM, and MXCSR State |
| FXTRACT | ***ST*** |  |  |  |  |  |  | D9 | F4 | 6 |  |  |  |  | s |  | 0123 | .1.. | 0.23 |  | Extract Exponent and Significand |
| FYL2X | ***ST1*** | *ST* |  |  |  |  |  | D9 | F1 | 6 |  |  |  |  | p |  | 0123 | .1.. | 0.23 |  | Compute y × log2x and Pop |
| FYL2XP1 | ***ST1*** | *ST* |  |  |  |  |  | D9 | F9 | 7 |  |  |  |  | p |  | 0123 | .1.. | 0.23 |  | Compute y × log2(x+1) and Pop |
| GETSEC | *EAX* |  |  |  | smx |  | 0F | 37 |  |  | C2++ | D[16](http://ref.x86asm.net/coder32-abc.html#gen_note_GETSEC_0F37) |  |  |  |  |  |  |  |  | GETSEC Leaf Functions |
| GS | *GS* |  |  |  |  | 65 |  |  |  |  | 03+ |  |  |  |  |  |  |  |  |  | GS segment override prefix |
| HADDPD | **xmm** | xmm/m128 |  |  | sse3 | 66 | 0F | 7C |  | r | P4++ |  |  |  |  |  |  |  |  |  | Packed Double-FP Horizontal Add |
| HADDPS | **xmm** | xmm/m128 |  |  | sse3 | F2 | 0F | 7C |  | r | P4++ |  |  |  |  |  |  |  |  |  | Packed Single-FP Horizontal Add |
| HINT\_NOP | r/m16/32 |  |  |  |  |  | 0F | 18 |  | 4 | PP+ | M[14](http://ref.x86asm.net/coder32-abc.html#gen_note_hintable_nop_0F18_0F1F) |  |  |  |  |  |  |  |  | Hintable NOP |
| HINT\_NOP | r/m16/32 |  |  |  |  |  | 0F | 18 |  | 5 | PP+ | M[14](http://ref.x86asm.net/coder32-abc.html#gen_note_hintable_nop_0F18_0F1F) |  |  |  |  |  |  |  |  | Hintable NOP |
| HINT\_NOP | r/m16/32 |  |  |  |  |  | 0F | 18 |  | 6 | PP+ | M[14](http://ref.x86asm.net/coder32-abc.html#gen_note_hintable_nop_0F18_0F1F) |  |  |  |  |  |  |  |  | Hintable NOP |
| HINT\_NOP | r/m16/32 |  |  |  |  |  | 0F | 18 |  | 7 | PP+ | M[14](http://ref.x86asm.net/coder32-abc.html#gen_note_hintable_nop_0F18_0F1F) |  |  |  |  |  |  |  |  | Hintable NOP |
| HINT\_NOP | r/m16/32 |  |  |  |  |  | 0F | 19 |  |  | PP+ | M[14](http://ref.x86asm.net/coder32-abc.html#gen_note_hintable_nop_0F18_0F1F) |  |  |  |  |  |  |  |  | Hintable NOP |
| HINT\_NOP | r/m16/32 |  |  |  |  |  | 0F | 1A |  |  | PP+ | M[14](http://ref.x86asm.net/coder32-abc.html#gen_note_hintable_nop_0F18_0F1F) |  |  |  |  |  |  |  |  | Hintable NOP |
| HINT\_NOP | r/m16/32 |  |  |  |  |  | 0F | 1B |  |  | PP+ | M[14](http://ref.x86asm.net/coder32-abc.html#gen_note_hintable_nop_0F18_0F1F) |  |  |  |  |  |  |  |  | Hintable NOP |
| HINT\_NOP | r/m16/32 |  |  |  |  |  | 0F | 1C |  |  | PP+ | M[14](http://ref.x86asm.net/coder32-abc.html#gen_note_hintable_nop_0F18_0F1F) |  |  |  |  |  |  |  |  | Hintable NOP |
| HINT\_NOP | r/m16/32 |  |  |  |  |  | 0F | 1D |  |  | PP+ | M[14](http://ref.x86asm.net/coder32-abc.html#gen_note_hintable_nop_0F18_0F1F) |  |  |  |  |  |  |  |  | Hintable NOP |
| HINT\_NOP | r/m16/32 |  |  |  |  |  | 0F | 1E |  |  | PP+ | M[14](http://ref.x86asm.net/coder32-abc.html#gen_note_hintable_nop_0F18_0F1F) |  |  |  |  |  |  |  |  | Hintable NOP |
| HINT\_NOP | r/m16/32 |  |  |  |  |  | 0F | 1F |  | 1 | PP+ | M[14](http://ref.x86asm.net/coder32-abc.html#gen_note_hintable_nop_0F18_0F1F) |  |  |  |  |  |  |  |  | Hintable NOP |
| HINT\_NOP | r/m16/32 |  |  |  |  |  | 0F | 1F |  | 2 | PP+ | M[14](http://ref.x86asm.net/coder32-abc.html#gen_note_hintable_nop_0F18_0F1F) |  |  |  |  |  |  |  |  | Hintable NOP |
| HINT\_NOP | r/m16/32 |  |  |  |  |  | 0F | 1F |  | 3 | PP+ | M[14](http://ref.x86asm.net/coder32-abc.html#gen_note_hintable_nop_0F18_0F1F) |  |  |  |  |  |  |  |  | Hintable NOP |
| HINT\_NOP | r/m16/32 |  |  |  |  |  | 0F | 1F |  | 4 | PP+ | M[14](http://ref.x86asm.net/coder32-abc.html#gen_note_hintable_nop_0F18_0F1F) |  |  |  |  |  |  |  |  | Hintable NOP |
| HINT\_NOP | r/m16/32 |  |  |  |  |  | 0F | 1F |  | 5 | PP+ | M[14](http://ref.x86asm.net/coder32-abc.html#gen_note_hintable_nop_0F18_0F1F) |  |  |  |  |  |  |  |  | Hintable NOP |
| HINT\_NOP | r/m16/32 |  |  |  |  |  | 0F | 1F |  | 6 | PP+ | M[14](http://ref.x86asm.net/coder32-abc.html#gen_note_hintable_nop_0F18_0F1F) |  |  |  |  |  |  |  |  | Hintable NOP |
| HINT\_NOP | r/m16/32 |  |  |  |  |  | 0F | 1F |  | 7 | PP+ | M[14](http://ref.x86asm.net/coder32-abc.html#gen_note_hintable_nop_0F18_0F1F) |  |  |  |  |  |  |  |  | Hintable NOP |
| HLT |  |  |  |  |  |  |  | F4 |  |  |  |  |  | 0 |  |  |  |  |  |  | Halt |
| HSUBPD | **xmm** | xmm/m128 |  |  | sse3 | 66 | 0F | 7D |  | r | P4++ |  |  |  |  |  |  |  |  |  | Packed Double-FP Horizontal Subtract |
| HSUBPS | **xmm** | xmm/m128 |  |  | sse3 | F2 | 0F | 7D |  | r | P4++ |  |  |  |  |  |  |  |  |  | Packed Single-FP Horizontal Subtract |
| IDIV | ***AL*** | ***AH*** | *AX* | r/m8 |  |  |  | F6 |  | 7 |  |  |  |  |  |  | o..szapc |  | o..szapc |  | Signed Divide |
| IDIV | ***eDX*** | ***eAX*** | r/m16/32 |  |  |  |  | F7 |  | 7 |  |  |  |  |  |  | o..szapc |  | o..szapc |  | Signed Divide |
| IMUL | **r16/32** | r/m16/32 | imm16/32 |  |  |  |  | 69 |  | r | 01+ |  |  |  |  |  | o..szapc | o......c | ...szap. |  | Signed Multiply |
| IMUL | **r16/32** | r/m16/32 | imm8 |  |  |  |  | 6B |  | r | 01+ |  |  |  |  |  | o..szapc | o......c | ...szap. |  | Signed Multiply |
| IMUL | ***AX*** | *AL* | r/m8 |  |  |  |  | F6 |  | 5 |  |  |  |  |  |  | o..szapc | o......c | ...szap. |  | Signed Multiply |
| IMUL | ***eDX*** | ***eAX*** | r/m16/32 |  |  |  |  | F7 |  | 5 |  |  |  |  |  |  | o..szapc | o......c | ...szap. |  | Signed Multiply |
| IMUL | **r16/32** | r/m16/32 |  |  |  |  | 0F | AF |  | r | 03+ |  |  |  |  |  | o..szapc | o......c | ...szap. |  | Signed Multiply |
| IN | **AL** | imm8 |  |  |  |  |  | E4 |  |  |  |  |  | f[1](http://ref.x86asm.net/coder32-abc.html#rflags_iopl) |  |  |  |  |  |  | Input from Port |
| IN | **eAX** | imm8 |  |  |  |  |  | E5 |  |  |  |  |  | f[1](http://ref.x86asm.net/coder32-abc.html#rflags_iopl) |  |  |  |  |  |  | Input from Port |
| IN | **AL** | DX |  |  |  |  |  | EC |  |  |  |  |  | f[1](http://ref.x86asm.net/coder32-abc.html#rflags_iopl) |  |  |  |  |  |  | Input from Port |
| IN | **eAX** | DX |  |  |  |  |  | ED |  |  |  |  |  | f[1](http://ref.x86asm.net/coder32-abc.html#rflags_iopl) |  |  |  |  |  |  | Input from Port |
| INC | **r16/32** |  |  |  |  |  |  | 40+r | |  |  |  |  |  |  |  | o..szap. | o..szap. |  |  | Increment by 1 |
| INC | **r/m8** |  |  |  |  |  |  | FE |  | 0 |  |  |  |  | L |  | o..szap. | o..szap. |  |  | Increment by 1 |
| INC | **r/m16/32** |  |  |  |  |  |  | FF |  | 0 |  |  |  |  | L |  | o..szap. | o..szap. |  |  | Increment by 1 |
| INS | **m8** | DX |  |  |  |  |  | 6C |  |  | 01+ |  |  | f[1](http://ref.x86asm.net/coder32-abc.html#rflags_iopl) |  | .d...... |  |  |  |  | Input from Port to String |
| INSB | ***m8*** | *DX* |  |  |
| INS | **m16** | DX |  |  |  |  |  | 6D |  |  | 01+ |  |  | f[1](http://ref.x86asm.net/coder32-abc.html#rflags_iopl) |  | .d...... |  |  |  |  | Input from Port to String |
| INSW | ***m16*** | *DX* |  |  |
| INS | **m16/32** | DX |  |  |  |  |  | 6D |  |  | 03+ |  |  | f[1](http://ref.x86asm.net/coder32-abc.html#rflags_iopl) |  | .d...... |  |  |  |  | Input from Port to String |
| INSD | ***m32*** | *DX* |  |  |
| INSERTPS | **xmm** | m32 | imm8 |  | sse41 | 66 | 0F | 3A | 21 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Insert Packed Single-FP Value |
| INSERTPS | **xmm** | xmm | imm8 |  |
| INT | 3 | *eFlags* |  |  |  |  |  | CC |  |  |  |  |  | f |  |  | ..i..... | ..i..... |  | ..i..... | Call to Interrupt Procedure |
| INT | imm8 | *eFlags* |  |  |  |  |  | CD |  |  |  |  |  | f |  |  | ..i..... | ..i..... |  | ..i..... | Call to Interrupt Procedure |
| INT1 | *eFlags* |  |  |  |  |  |  | F1 |  |  | 03+ | U[9](http://ref.x86asm.net/coder32-abc.html#gen_note_u_INT1_ICEBP_F1) |  |  |  |  | ..i..... | ..i..... |  | ..i..... | Call to Interrupt Procedure |
| ICEBP | *eFlags* |  |  |  |
| INTO | *eFlags* |  |  |  |  |  |  | CE |  |  |  |  |  | f |  | o....... | ..i..... | ..i..... |  | ..i..... | Call to Interrupt Procedure |
| INVD |  |  |  |  |  |  | 0F | 08 |  |  | 04+ |  |  | 0 |  |  |  |  |  |  | Invalidate Internal Caches |
| INVEPT | r32 | m128 |  |  | vmx | 66 | 0F | 38 | 80 | r | C2++ | D[23](http://ref.x86asm.net/coder32-abc.html#gen_note_VMX_vs_SVM) | P | 0 |  |  | o..szapc | o..szapc |  |  | Invalidate Translations Derived from EPT |
| INVLPG | m |  |  |  |  |  | 0F | 01 |  | 7 | 04+ |  |  | 0 |  |  |  |  |  |  | Invalidate TLB Entry |
| INVVPID | r32 | m128 |  |  | vmx | 66 | 0F | 38 | 81 | r | C2++ | D[23](http://ref.x86asm.net/coder32-abc.html#gen_note_VMX_vs_SVM) | P | 0 |  |  | o..szapc | o..szapc |  |  | Invalidate Translations Based on VPID |
| IRET | ***Flags*** |  |  |  |  |  |  | CF |  |  |  |  |  | f |  |  |  |  |  |  | Interrupt Return |
| IRETD | ***EFlags*** |  |  |  |  |  |  | CF |  |  | 03+ |  |  | f |  |  |  |  |  |  | Interrupt Return |
| JB | rel8 |  |  |  |  |  |  | 72 |  |  |  |  |  |  |  | .......c |  |  |  |  | Jump short if below/not above or equal/carry (CF=1) |
| JNAE | rel8 |  |  |  |
| JC | rel8 |  |  |  |
| JB | rel16/32 |  |  |  |  |  | 0F | 83 |  |  | 03+ |  |  |  |  | .......c |  |  |  |  | Jump near if below/not above or equal/carry (CF=1) |
| JNAE | rel16/32 |  |  |  |
| JC | rel16/32 |  |  |  |
| JBE | rel8 |  |  |  |  |  |  | 76 |  |  |  |  |  |  |  | ....z..c |  |  |  |  | Jump short if below or equal/not above (CF=1 OR ZF=1) |
| JNA | rel8 |  |  |  |
| JBE | rel16/32 |  |  |  |  |  | 0F | 86 |  |  | 03+ |  |  |  |  | ....z..c |  |  |  |  | Jump near if below or equal/not above (CF=1 OR ZF=1) |
| JNA | rel16/32 |  |  |  |
| JCXZ | rel8 | *CX* |  |  |  |  |  | E3 |  |  |  |  |  |  |  |  |  |  |  |  | Jump short if eCX register is 0 |
| JECXZ | rel8 | *ECX* |  |  |
| JL | rel8 |  |  |  |  |  |  | 7C |  |  |  |  |  |  |  | o..s.... |  |  |  |  | Jump short if less/not greater (SF!=OF) |
| JNGE | rel8 |  |  |  |
| JL | rel16/32 |  |  |  |  |  | 0F | 8C |  |  | 03+ |  |  |  |  | o..s.... |  |  |  |  | Jump near if less/not greater (SF!=OF) |
| JNGE | rel16/32 |  |  |  |
| JLE | rel8 |  |  |  |  |  |  | 7E |  |  |  |  |  |  |  | o..sz... |  |  |  |  | Jump short if less or equal/not greater ((ZF=1) OR (SF!=OF)) |
| JNG | rel8 |  |  |  |
| JLE | rel16/32 |  |  |  |  |  | 0F | 8E |  |  | 03+ |  |  |  |  | o..sz... |  |  |  |  | Jump near if less or equal/not greater ((ZF=1) OR (SF!=OF)) |
| JNG | rel16/32 |  |  |  |
| JMP | rel16/32 |  |  |  |  |  |  | E9 |  |  |  |  |  |  |  |  |  |  |  |  | Jump |
| JMP | rel8 |  |  |  |  |  |  | EB |  |  |  |  |  |  |  |  |  |  |  |  | Jump |
| JMP | r/m16/32 |  |  |  |  |  |  | FF |  | 4 |  |  |  |  |  |  |  |  |  |  | Jump |
| JMPF | ptr16:16/32 |  |  |  |  |  |  | EA |  |  |  |  |  |  |  |  |  |  |  |  | Jump |
| JMPF | m16:16/32 |  |  |  |  |  |  | FF |  | 5 |  |  |  |  |  |  |  |  |  |  | Jump |
| JNB | rel8 |  |  |  |  |  |  | 73 |  |  |  |  |  |  |  | .......c |  |  |  |  | Jump short if not below/above or equal/not carry (CF=0) |
| JAE | rel8 |  |  |  |
| JNC | rel8 |  |  |  |
| JNB | rel16/32 |  |  |  |  |  | 0F | 83 |  |  | 03+ |  |  |  |  | .......c |  |  |  |  | Jump near if not below/above or equal/not carry (CF=0) |
| JAE | rel16/32 |  |  |  |
| JNC | rel16/32 |  |  |  |
| JNBE | rel8 |  |  |  |  |  |  | 77 |  |  |  |  |  |  |  | ....z..c |  |  |  |  | Jump short if not below or equal/above (CF=0 AND ZF=0) |
| JA | rel8 |  |  |  |
| JNBE | rel16/32 |  |  |  |  |  | 0F | 87 |  |  | 03+ |  |  |  |  | ....z..c |  |  |  |  | Jump near if not below or equal/above (CF=0 AND ZF=0) |
| JA | rel16/32 |  |  |  |
| JNL | rel8 |  |  |  |  |  |  | 7D |  |  |  |  |  |  |  | o..s.... |  |  |  |  | Jump short if not less/greater or equal (SF=OF) |
| JGE | rel8 |  |  |  |
| JNL | rel16/32 |  |  |  |  |  | 0F | 8D |  |  | 03+ |  |  |  |  | o..s.... |  |  |  |  | Jump near if not less/greater or equal (SF=OF) |
| JGE | rel16/32 |  |  |  |
| JNLE | rel8 |  |  |  |  |  |  | 7F |  |  |  |  |  |  |  | o..sz... |  |  |  |  | Jump short if not less nor equal/greater ((ZF=0) AND (SF=OF)) |
| JG | rel8 |  |  |  |
| JNLE | rel16/32 |  |  |  |  |  | 0F | 8F |  |  | 03+ |  |  |  |  | o..sz... |  |  |  |  | Jump near if not less nor equal/greater ((ZF=0) AND (SF=OF)) |
| JG | rel16/32 |  |  |  |
| JNO | rel8 |  |  |  |  |  |  | 71 |  |  |  |  |  |  |  | o....... |  |  |  |  | Jump short if not overflow (OF=0) |
| JNO | rel16/32 |  |  |  |  |  | 0F | 81 |  |  | 03+ |  |  |  |  | o....... |  |  |  |  | Jump near if not overflow (OF=0) |
| JNP | rel8 |  |  |  |  |  |  | 7B |  |  |  |  |  |  |  | ......p. |  |  |  |  | Jump short if not parity/parity odd (PF=0) |
| JPO | rel8 |  |  |  |
| JNP | rel16/32 |  |  |  |  |  | 0F | 8B |  |  | 03+ |  |  |  |  | ......p. |  |  |  |  | Jump near if not parity/parity odd (PF=0) |
| JPO | rel16/32 |  |  |  |
| JNS | rel8 |  |  |  |  |  |  | 79 |  |  |  |  |  |  |  | ...s.... |  |  |  |  | Jump short if not sign (SF=0) |
| JNS | rel16/32 |  |  |  |  |  | 0F | 89 |  |  | 03+ |  |  |  |  | ...s.... |  |  |  |  | Jump near if not sign (SF=0) |
| JNZ | rel8 |  |  |  |  |  |  | 75 |  |  |  |  |  |  |  | ....z... |  |  |  |  | Jump short if not zero/not equal (ZF=0) |
| JNE | rel8 |  |  |  |
| JNZ | rel16/32 |  |  |  |  |  | 0F | 85 |  |  | 03+ |  |  |  |  | ....z... |  |  |  |  | Jump near if not zero/not equal (ZF=0) |
| JNE | rel16/32 |  |  |  |
| JO | rel8 |  |  |  |  |  |  | 70 |  |  |  |  |  |  |  | o....... |  |  |  |  | Jump short if overflow (OF=1) |
| JO | rel16/32 |  |  |  |  |  | 0F | 80 |  |  | 03+ |  |  |  |  | o....... |  |  |  |  | Jump near if overflow (OF=1) |
| JP | rel8 |  |  |  |  |  |  | 7A |  |  |  |  |  |  |  | ......p. |  |  |  |  | Jump short if parity/parity even (PF=1) |
| JPE | rel8 |  |  |  |
| JP | rel16/32 |  |  |  |  |  | 0F | 8A |  |  | 03+ |  |  |  |  | ......p. |  |  |  |  | Jump near if parity/parity even (PF=1) |
| JPE | rel16/32 |  |  |  |
| JS | rel8 |  |  |  |  |  |  | 78 |  |  |  |  |  |  |  | ...s.... |  |  |  |  | Jump short if sign (SF=1) |
| JS | rel16/32 |  |  |  |  |  | 0F | 88 |  |  | 03+ |  |  |  |  | ...s.... |  |  |  |  | Jump near if sign (SF=1) |
| JZ | rel8 |  |  |  |  |  |  | 74 |  |  |  |  |  |  |  | ....z... |  |  |  |  | Jump short if zero/equal (ZF=1) |
| JE | rel8 |  |  |  |
| JZ | rel16/32 |  |  |  |  |  | 0F | 84 |  |  | 03+ |  |  |  |  | ....z... |  |  |  |  | Jump near if zero/equal (ZF=1) |
| JE | rel16/32 |  |  |  |
| LAHF | ***AH*** |  |  |  |  |  |  | 9F |  |  |  |  |  |  |  | ...szapc |  |  |  |  | Load Status Flags into AH Register |
| LAR | **r16/32** | m16 |  |  |  |  | 0F | 02 |  | r | 02+ |  | P |  |  |  | ....z... | ....z... |  |  | Load Access Rights Byte |
| LAR | **r16/32** | r16/32 |  |  |
| LDDQU | **xmm** | m128 |  |  | sse3 | F2 | 0F | F0 |  | r | P4++ |  |  |  |  |  |  |  |  |  | Load Unaligned Integer 128 Bits |
| LDMXCSR | m32 |  |  |  | sse1 |  | 0F | AE |  | 2 | P3+ |  |  |  |  |  |  |  |  |  | Load MXCSR Register |
| LDS | ***DS*** | **r16/32** | m16:16/32 |  |  |  |  | C5 |  | r |  |  |  |  |  |  |  |  |  |  | Load Far Pointer |
| LEA | **r16/32** | m |  |  |  |  |  | 8D |  | r |  |  |  |  |  |  |  |  |  |  | Load Effective Address |
| LEAVE | ***eBP*** |  |  |  |  |  |  | C9 |  |  | 01+ |  |  |  |  |  |  |  |  |  | High Level Procedure Exit |
| LES | ***ES*** | **r16/32** | m16:16/32 |  |  |  |  | C4 |  | r |  |  |  |  |  |  |  |  |  |  | Load Far Pointer |
| LFENCE |  |  |  |  | sse2 |  | 0F | AE |  | 5 | P4+ |  |  |  |  |  |  |  |  |  | Load Fence |
| LFS | ***FS*** | **r16/32** | m16:16/32 |  |  |  | 0F | B4 |  | r | 03+ |  |  |  |  |  |  |  |  |  | Load Far Pointer |
| LGDT | ***GDTR*** | m |  |  |  |  | 0F | 01 |  | 2 | 02+ |  |  | 0 |  |  |  |  |  |  | Load Global Descriptor Table Register |
| LGS | ***GS*** | **r16/32** | m16:16/32 |  |  |  | 0F | B5 |  | r | 03+ |  |  |  |  |  |  |  |  |  | Load Far Pointer |
| LIDT | ***IDTR*** | m |  |  |  |  | 0F | 01 |  | 3 | 02+ |  |  | 0 |  |  |  |  |  |  | Load Interrupt Descriptor Table Register |
| LLDT | ***LDTR*** | r/m16 |  |  |  |  | 0F | 00 |  | 2 | 02+ |  | P | 0 |  |  |  |  |  |  | Load Local Descriptor Table Register |
| LMSW | ***MSW*** | r/m16 |  |  |  |  | 0F | 01 |  | 6 | 02+ |  |  | 0 |  |  |  |  |  |  | Load Machine Status Word |
| LOCK |  |  |  |  |  | F0 |  |  |  |  |  |  |  |  |  |  |  |  |  |  | Assert LOCK# Signal Prefix |
| LODS | ***AL*** | m8 |  |  |  |  |  | AC |  |  |  |  |  |  |  | .d...... |  |  |  |  | Load String |
| LODSB | ***AL*** | *m8* |  |  |
| LODS | ***AX*** | m16 |  |  |  |  |  | AD |  |  |  |  |  |  |  | .d...... |  |  |  |  | Load String |
| LODSW | ***AX*** | *m16* |  |  |
| LODS | ***eAX*** | m16/32 |  |  |  |  |  | AD |  |  | 03+ |  |  |  |  | .d...... |  |  |  |  | Load String |
| LODSD | ***EAX*** | *m32* |  |  |
| LOOP | ***eCX*** | rel8 |  |  |  |  |  | E2 |  |  |  |  |  |  |  |  |  |  |  |  | Decrement count; Jump short if count!=0 |
| LOOPNZ | ***eCX*** | rel8 |  |  |  |  |  | E0 |  |  |  |  |  |  |  | ....z... |  |  |  |  | Decrement count; Jump short if count!=0 and ZF=0 |
| LOOPNE | ***eCX*** | rel8 |  |  |
| LOOPZ | ***eCX*** | rel8 |  |  |  |  |  | E1 |  |  |  |  |  |  |  | ....z... |  |  |  |  | Decrement count; Jump short if count!=0 and ZF=1 |
| LOOPE | ***eCX*** | rel8 |  |  |
| LSL | **r16/32** | m16 |  |  |  |  | 0F | 03 |  | r | 02+ |  | P |  |  |  | ....z... | ....z... |  |  | Load Segment Limit |
| LSL | **r16/32** | r16/32 |  |  |
| LSS | ***SS*** | **r16/32** | m16:16/32 |  |  |  | 0F | B2 |  | r | 03+ |  |  |  |  |  |  |  |  |  | Load Far Pointer |
| LTR | ***TR*** | r/m16 |  |  |  |  | 0F | 00 |  | 3 | 02+ |  | P | 0 |  |  |  |  |  |  | Load Task Register |
| MASKMOVDQU | ***m128*** | xmm | xmm |  | sse2 | 66 | 0F | F7 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Store Selected Bytes of Double Quadword |
| MASKMOVQ | ***m64*** | **mm** | mm |  | sse1 |  | 0F | F7 |  | r | P3+ | D[22](http://ref.x86asm.net/coder32-abc.html#gen_note_MASKMOVQ_0FF7) |  |  |  |  |  |  |  |  | Store Selected Bytes of Quadword |
| MAXPD | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 5F |  | r | P4+ |  |  |  |  |  |  |  |  |  | Return Maximum Packed Double-FP Values |
| MAXPS | **xmm** | xmm/m128 |  |  | sse1 |  | 0F | 5F |  | r | P3+ |  |  |  |  |  |  |  |  |  | Return Maximum Packed Single-FP Values |
| MAXSD | **xmm** | xmm/m64 |  |  | sse2 | F2 | 0F | 5F |  | r | P4+ |  |  |  |  |  |  |  |  |  | Return Maximum Scalar Double-FP Value |
| MAXSS | **xmm** | xmm/m32 |  |  | sse1 | F3 | 0F | 5F |  | r | P3+ |  |  |  |  |  |  |  |  |  | Return Maximum Scalar Single-FP Value |
| MFENCE |  |  |  |  | sse2 |  | 0F | AE |  | 6 | P4+ |  |  |  |  |  |  |  |  |  | Memory Fence |
| MINPD | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 5D |  | r | P4+ |  |  |  |  |  |  |  |  |  | Return Minimum Packed Double-FP Values |
| MINPS | **xmm** | xmm/m128 |  |  | sse1 |  | 0F | 5D |  | r | P3+ |  |  |  |  |  |  |  |  |  | Return Minimum Packed Single-FP Values |
| MINSD | **xmm** | xmm/m64 |  |  | sse2 | F2 | 0F | 5D |  | r | P4+ |  |  |  |  |  |  |  |  |  | Return Minimum Scalar Double-FP Value |
| MINSS | **xmm** | xmm/m32 |  |  | sse1 | F3 | 0F | 5D |  | r | P3+ |  |  |  |  |  |  |  |  |  | Return Minimum Scalar Single-FP Value |
| MONITOR | *m8* | *ECX* | *EDX* |  | sse3 |  | 0F | 01 | C8 | 1 | P4++ |  |  | 0 |  |  |  |  |  |  | Set Up Monitor Address |
| MOV | **r/m8** | r8 |  |  |  |  |  | 88 |  | r |  |  |  |  |  |  |  |  |  |  | Move |
| MOV | **r/m16/32** | r16/32 |  |  |  |  |  | 89 |  | r |  |  |  |  |  |  |  |  |  |  | Move |
| MOV | **r8** | r/m8 |  |  |  |  |  | 8A |  | r |  |  |  |  |  |  |  |  |  |  | Move |
| MOV | **r16/32** | r/m16/32 |  |  |  |  |  | 8B |  | r |  |  |  |  |  |  |  |  |  |  | Move |
| MOV | **m16** | Sreg |  |  |  |  |  | 8C |  | r |  |  |  |  |  |  |  |  |  |  | Move |
| MOV | **r16/32** | Sreg |  |  |
| MOV | **Sreg** | r/m16 |  |  |  |  |  | 8E |  | r |  |  |  |  |  |  |  |  |  |  | Move |
| MOV | **AL** | moffs8 |  |  |  |  |  | A0 |  |  |  |  |  |  |  |  |  |  |  |  | Move |
| MOV | **eAX** | moffs16/32 |  |  |  |  |  | A1 |  |  |  |  |  |  |  |  |  |  |  |  | Move |
| MOV | **moffs8** | AL |  |  |  |  |  | A2 |  |  |  |  |  |  |  |  |  |  |  |  | Move |
| MOV | **moffs16/32** | eAX |  |  |  |  |  | A3 |  |  |  |  |  |  |  |  |  |  |  |  | Move |
| MOV | **r8** | imm8 |  |  |  |  |  | B0+r | |  |  |  |  |  |  |  |  |  |  |  | Move |
| MOV | **r16/32** | imm16/32 |  |  |  |  |  | B8+r | |  |  |  |  |  |  |  |  |  |  |  | Move |
| MOV | **r/m8** | imm8 |  |  |  |  |  | C6 |  | 0 |  |  |  |  |  |  |  |  |  |  | Move |
| MOV | **r/m16/32** | imm16/32 |  |  |  |  |  | C7 |  | 0 |  |  |  |  |  |  |  |  |  |  | Move |
| MOV | **r32** | CRn |  |  |  |  | 0F | 20 |  | r | 03+ |  |  | 0 |  |  | o..szapc |  | o..szapc |  | Move to/from Control Registers |
| MOV | **r32** | CRn |  |  |  |  | 0F | 20 |  | r | 03+ | U[15](http://ref.x86asm.net/coder32-abc.html#gen_note_u_MOV_CR_DR_TR_0F20_0F21_0F22_0F23_0F24_0F26) |  | 0 |  |  | o..szapc |  | o..szapc |  | Move to/from Control Registers |
| MOV | **r32** | DRn |  |  |  |  | 0F | 21 |  | r | 03+ |  |  | 0 |  |  | o..szapc |  | o..szapc |  | Move to/from Debug Registers |
| MOV | **r32** | DRn |  |  |  |  | 0F | 21 |  | r | 03+ | U[15](http://ref.x86asm.net/coder32-abc.html#gen_note_u_MOV_CR_DR_TR_0F20_0F21_0F22_0F23_0F24_0F26) |  | 0 |  |  | o..szapc |  | o..szapc |  | Move to/from Debug Registers |
| MOV | **CRn** | r32 |  |  |  |  | 0F | 22 |  | r | 03+ |  |  | 0 |  |  | o..szapc |  | o..szapc |  | Move to/from Control Registers |
| MOV | **CRn** | r32 |  |  |  |  | 0F | 22 |  | r | 03+ | U[15](http://ref.x86asm.net/coder32-abc.html#gen_note_u_MOV_CR_DR_TR_0F20_0F21_0F22_0F23_0F24_0F26) |  | 0 |  |  | o..szapc |  | o..szapc |  | Move to/from Control Registers |
| MOV | **DRn** | r32 |  |  |  |  | 0F | 23 |  | r | 03+ |  |  | 0 |  |  | o..szapc |  | o..szapc |  | Move to/from Debug Registers |
| MOV | **DRn** | r64 |  |  |  |  | 0F | 23 |  | r | 03+ | U[15](http://ref.x86asm.net/coder32-abc.html#gen_note_u_MOV_CR_DR_TR_0F20_0F21_0F22_0F23_0F24_0F26) |  | 0 |  |  | o..szapc |  | o..szapc |  | Move to/from Debug Registers |
| MOVAPD | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 28 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Move Aligned Packed Double-FP Values |
| MOVAPD | **xmm/m128** | xmm |  |  | sse2 | 66 | 0F | 29 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Move Aligned Packed Double-FP Values |
| MOVAPS | **xmm** | xmm/m128 |  |  | sse1 |  | 0F | 28 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Move Aligned Packed Single-FP Values |
| MOVAPS | **xmm/m128** | xmm |  |  | sse1 |  | 0F | 29 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Move Aligned Packed Single-FP Values |
| MOVBE | **r16/32** | m16/32 |  |  |  |  | 0F | 38 | F0 | r | C2++ |  |  |  |  |  |  |  |  |  | Move Data After Swapping Bytes |
| MOVBE | **m16/32** | r16/32 |  |  |  |  | 0F | 38 | F1 | r | C2++ |  |  |  |  |  |  |  |  |  | Move Data After Swapping Bytes |
| MOVD | **mm** | r/m32 |  |  | mmx |  | 0F | 6E |  | r | PX+ |  |  |  |  |  |  |  |  |  | Move Doubleword |
| MOVD | **xmm** | r/m32 |  |  | sse2 | 66 | 0F | 6E |  | r | P4+ |  |  |  |  |  |  |  |  |  | Move Doubleword |
| MOVD | **r/m32** | mm |  |  | mmx |  | 0F | 7E |  | r | PX+ |  |  |  |  |  |  |  |  |  | Move Doubleword |
| MOVD | **r/m32** | xmm |  |  | sse2 | 66 | 0F | 7E |  | r | P4+ |  |  |  |  |  |  |  |  |  | Move Doubleword |
| MOVDDUP | **xmm** | xmm/m64 |  |  | sse3 | F2 | 0F | 12 |  | r | P4++ |  |  |  |  |  |  |  |  |  | Move One Double-FP and Duplicate |
| MOVDQ2Q | **mm** | xmm |  |  | sse2 | F2 | 0F | D6 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Move Quadword from XMM to MMX Technology Register |
| MOVDQA | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 6F |  | r | P4+ |  |  |  |  |  |  |  |  |  | Move Aligned Double Quadword |
| MOVDQA | **xmm/m128** | xmm |  |  | sse2 | 66 | 0F | 7F |  | r | P4+ |  |  |  |  |  |  |  |  |  | Move Aligned Double Quadword |
| MOVDQU | **xmm** | xmm/m128 |  |  | sse2 | F3 | 0F | 6F |  | r | P4+ |  |  |  |  |  |  |  |  |  | Move Unaligned Double Quadword |
| MOVDQU | **xmm/m128** | xmm |  |  | sse2 | F3 | 0F | 7F |  | r | P4+ |  |  |  |  |  |  |  |  |  | Move Unaligned Double Quadword |
| MOVHLPS | **xmm** | xmm |  |  | sse1 |  | 0F | 12 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Move Packed Single-FP Values High to Low |
| MOVHPD | **xmm** | m64 |  |  | sse2 | 66 | 0F | 16 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Move High Packed Double-FP Value |
| MOVHPD | **m64** | xmm |  |  | sse2 | 66 | 0F | 17 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Move High Packed Double-FP Value |
| MOVHPS | **xmm** | m64 |  |  | sse1 |  | 0F | 16 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Move High Packed Single-FP Values |
| MOVHPS | **m64** | xmm |  |  | sse1 |  | 0F | 17 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Move High Packed Single-FP Values |
| MOVLHPS | **xmm** | xmm |  |  | sse1 |  | 0F | 16 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Move Packed Single-FP Values Low to High |
| MOVLPD | **xmm** | m64 |  |  | sse2 | 66 | 0F | 12 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Move Low Packed Double-FP Value |
| MOVLPD | **m64** | xmm |  |  | sse2 | 66 | 0F | 13 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Move Low Packed Double-FP Value |
| MOVLPS | **xmm** | m64 |  |  | sse1 |  | 0F | 12 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Move Low Packed Single-FP Values |
| MOVLPS | **m64** | xmm |  |  | sse1 |  | 0F | 13 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Move Low Packed Single-FP Values |
| MOVMSKPD | **r32** | xmm |  |  | sse2 | 66 | 0F | 50 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Extract Packed Double-FP Sign Mask |
| MOVMSKPS | **r32** | xmm |  |  | sse1 |  | 0F | 50 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Extract Packed Single-FP Sign Mask |
| MOVNTDQ | **m128** | xmm |  |  | sse2 | 66 | 0F | E7 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Store Double Quadword Using Non-Temporal Hint |
| MOVNTDQA | **xmm** | m128 |  |  | sse41 | 66 | 0F | 38 | 2A | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Load Double Quadword Non-Temporal Aligned Hint |
| MOVNTI | **m32** | r32 |  |  | sse2 |  | 0F | C3 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Store Doubleword Using Non-Temporal Hint |
| MOVNTPD | **m128** | xmm |  |  | sse2 | 66 | 0F | 2B |  | r | P4+ |  |  |  |  |  |  |  |  |  | Store Packed Double-FP Values Using Non-Temporal Hint |
| MOVNTPS | **m128** | xmm |  |  | sse1 |  | 0F | 2B |  | r | P3+ |  |  |  |  |  |  |  |  |  | Store Packed Single-FP Values Using Non-Temporal Hint |
| MOVNTQ | **m64** | mm |  |  | sse1 |  | 0F | E7 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Store of Quadword Using Non-Temporal Hint |
| MOVQ | **mm** | mm/m64 |  |  | mmx |  | 0F | 6F |  | r | PX+ |  |  |  |  |  |  |  |  |  | Move Quadword |
| MOVQ | **xmm** | xmm/m64 |  |  | sse2 | F3 | 0F | 7E |  | r | P4+ |  |  |  |  |  |  |  |  |  | Move Quadword |
| MOVQ | **mm/m64** | mm |  |  | mmx |  | 0F | 7F |  | r | PX+ |  |  |  |  |  |  |  |  |  | Move Quadword |
| MOVQ | **xmm/m64** | xmm |  |  | sse2 | 66 | 0F | D6 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Move Quadword |
| MOVQ2DQ | **xmm** | mm |  |  | sse2 | F3 | 0F | D6 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Move Quadword from MMX Technology to XMM Register |
| MOVS | **m8** | m8 |  |  |  |  |  | A4 |  |  |  |  |  |  |  | .d...... |  |  |  |  | Move Data from String to String |
| MOVSB | ***m8*** | *m8* |  |  |
| MOVS | **m16** | m16 |  |  |  |  |  | A5 |  |  |  |  |  |  |  | .d...... |  |  |  |  | Move Data from String to String |
| MOVSW | ***m16*** | *m16* |  |  |
| MOVS | **m16/32** | m16/32 |  |  |  |  |  | A5 |  |  | 03+ |  |  |  |  | .d...... |  |  |  |  | Move Data from String to String |
| MOVSD | ***m32*** | *m32* |  |  |
| MOVSD | **xmm** | xmm/m64 |  |  | sse2 | F2 | 0F | 10 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Move Scalar Double-FP Value |
| MOVSD | **xmm/m64** | xmm |  |  | sse2 | F2 | 0F | 11 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Move Scalar Double-FP Value |
| MOVSHDUP | **xmm** | xmm/m64 |  |  | sse3 | F3 | 0F | 16 |  | r | P4++ |  |  |  |  |  |  |  |  |  | Move Packed Single-FP High and Duplicate |
| MOVSLDUP | **xmm** | xmm/m64 |  |  | sse3 | F3 | 0F | 12 |  | r | P4++ |  |  |  |  |  |  |  |  |  | Move Packed Single-FP Low and Duplicate |
| MOVSS | **xmm** | xmm/m32 |  |  | sse1 | F3 | 0F | 10 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Move Scalar Single-FP Values |
| MOVSS | **xmm/m32** | xmm |  |  | sse1 | F3 | 0F | 11 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Move Scalar Single-FP Values |
| MOVSX | **r16/32** | r/m8 |  |  |  |  | 0F | BE |  | r | 03+ |  |  |  |  |  |  |  |  |  | Move with Sign-Extension |
| MOVSX | **r16/32** | r/m16 |  |  |  |  | 0F | BF |  | r | 03+ |  |  |  |  |  |  |  |  |  | Move with Sign-Extension |
| MOVUPD | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 10 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Move Unaligned Packed Double-FP Value |
| MOVUPD | **xmm/m128** | xmm |  |  | sse2 | 66 | 0F | 11 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Move Unaligned Packed Double-FP Values |
| MOVUPS | **xmm** | xmm/m128 |  |  | sse1 |  | 0F | 10 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Move Unaligned Packed Single-FP Values |
| MOVUPS | **xmm/m128** | xmm |  |  | sse1 |  | 0F | 11 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Move Unaligned Packed Single-FP Values |
| MOVZX | **r16/32** | r/m8 |  |  |  |  | 0F | B6 |  | r | 03+ |  |  |  |  |  |  |  |  |  | Move with Zero-Extend |
| MOVZX | **r16/32** | r/m16 |  |  |  |  | 0F | B7 |  | r | 03+ |  |  |  |  |  |  |  |  |  | Move with Zero-Extend |
| MPSADBW | **xmm** | xmm/m128 | imm8 |  | sse41 | 66 | 0F | 3A | 42 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Compute Multiple Packed Sums of Absolute Difference |
| MUL | ***AX*** | *AL* | r/m8 |  |  |  |  | F6 |  | 4 |  |  |  |  |  |  | o..szapc | o......c | ...szap. |  | Unsigned Multiply |
| MUL | ***eDX*** | ***eAX*** | r/m16/32 |  |  |  |  | F7 |  | 4 |  |  |  |  |  |  | o..szapc | o......c | ...szap. |  | Unsigned Multiply |
| MULPD | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 59 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Multiply Packed Double-FP Values |
| MULPS | **xmm** | xmm/m128 |  |  | sse1 |  | 0F | 59 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Multiply Packed Single-FP Values |
| MULSD | **xmm** | xmm/m64 |  |  | sse2 | F2 | 0F | 59 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Multiply Scalar Double-FP Values |
| MULSS | **xmm** | xmm/m32 |  |  | sse1 | F3 | 0F | 59 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Multiply Scalar Single-FP Value |
| MWAIT | *EAX* | *ECX* |  |  | sse3 |  | 0F | 01 | C9 | 1 | P4++ |  |  | 0 |  |  |  |  |  |  | Monitor Wait |
| NEG | **r/m8** |  |  |  |  |  |  | F6 |  | 3 |  |  |  |  | L |  | o..szapc | o..szapc |  |  | Two's Complement Negation |
| NEG | **r/m16/32** |  |  |  |  |  |  | F7 |  | 3 |  |  |  |  | L |  | o..szapc | o..szapc |  |  | Two's Complement Negation |
| NOP |  |  |  |  |  |  |  | 90 |  |  |  | D[1](http://ref.x86asm.net/coder32-abc.html#gen_note_90_NOP) |  |  |  |  |  |  |  |  | No Operation |
| NOP | r/m16/32 |  |  |  |  |  | 0F | 0D |  |  | PP+ | M[13](http://ref.x86asm.net/coder32-abc.html#gen_note_NOP_0F0D) |  |  |  |  |  |  |  |  | No Operation |
| NOP | r/m16/32 |  |  |  |  |  | 0F | 1F |  | 0 | P4++ |  |  |  |  |  |  |  |  |  | No Operation |
| NOT | **r/m8** |  |  |  |  |  |  | F6 |  | 2 |  |  |  |  | L |  |  |  |  |  | One's Complement Negation |
| NOT | **r/m16/32** |  |  |  |  |  |  | F7 |  | 2 |  |  |  |  | L |  |  |  |  |  | One's Complement Negation |
| OR | **r/m8** | r8 |  |  |  |  |  | 08 |  | r |  |  |  |  | L |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical Inclusive OR |
| OR | **r/m16/32** | r16/32 |  |  |  |  |  | 09 |  | r |  |  |  |  | L |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical Inclusive OR |
| OR | **r8** | r/m8 |  |  |  |  |  | 0A |  | r |  |  |  |  |  |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical Inclusive OR |
| OR | **r16/32** | r/m16/32 |  |  |  |  |  | 0B |  | r |  |  |  |  |  |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical Inclusive OR |
| OR | **AL** | imm8 |  |  |  |  |  | 0C |  |  |  |  |  |  |  |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical Inclusive OR |
| OR | **eAX** | imm16/32 |  |  |  |  |  | 0D |  |  |  |  |  |  |  |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical Inclusive OR |
| OR | **r/m8** | imm8 |  |  |  |  |  | 80 |  | 1 |  |  |  |  | L |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical Inclusive OR |
| OR | **r/m16/32** | imm16/32 |  |  |  |  |  | 81 |  | 1 |  |  |  |  | L |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical Inclusive OR |
| OR | **r/m8** | imm8 |  |  |  |  |  | 82 |  | 1 |  |  |  |  | L |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical Inclusive OR |
| OR | **r/m16/32** | imm8 |  |  |  |  |  | 83 |  | 1 | 03+ |  |  |  | L |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical Inclusive OR |
| ORPD | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 56 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Bitwise Logical OR of Double-FP Values |
| ORPS | **xmm** | xmm/m128 |  |  | sse1 |  | 0F | 56 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Bitwise Logical OR of Single-FP Values |
| OUT | **imm8** | AL |  |  |  |  |  | E6 |  |  |  |  |  | f[1](http://ref.x86asm.net/coder32-abc.html#rflags_iopl) |  |  |  |  |  |  | Output to Port |
| OUT | **imm8** | eAX |  |  |  |  |  | E7 |  |  |  |  |  | f[1](http://ref.x86asm.net/coder32-abc.html#rflags_iopl) |  |  |  |  |  |  | Output to Port |
| OUT | **DX** | AL |  |  |  |  |  | EE |  |  |  |  |  | f[1](http://ref.x86asm.net/coder32-abc.html#rflags_iopl) |  |  |  |  |  |  | Output to Port |
| OUT | **DX** | eAX |  |  |  |  |  | EF |  |  |  |  |  | f[1](http://ref.x86asm.net/coder32-abc.html#rflags_iopl) |  |  |  |  |  |  | Output to Port |
| OUTS | **DX** | m8 |  |  |  |  |  | 6E |  |  | 01+ |  |  | f[1](http://ref.x86asm.net/coder32-abc.html#rflags_iopl) |  | .d...... |  |  |  |  | Output String to Port |
| OUTSB | ***DX*** | *m8* |  |  |
| OUTS | **DX** | m16 |  |  |  |  |  | 6F |  |  | 01+ |  |  | f[1](http://ref.x86asm.net/coder32-abc.html#rflags_iopl) |  | .d...... |  |  |  |  | Output String to Port |
| OUTSW | ***DX*** | *m16* |  |  |
| OUTS | **DX** | m16/32 |  |  |  |  |  | 6F |  |  | 03+ |  |  | f[1](http://ref.x86asm.net/coder32-abc.html#rflags_iopl) |  | .d...... |  |  |  |  | Output String to Port |
| OUTSD | ***DX*** | *m32* |  |  |
| PABSB | **mm** | mm/m64 |  |  | ssse3 |  | 0F | 38 | 1C | r | C2+ |  |  |  |  |  |  |  |  |  | Packed Absolute Value |
| PABSB | **xmm** | xmm/m128 |  |  | ssse3 | 66 | 0F | 38 | 1C | r | C2+ |  |  |  |  |  |  |  |  |  | Packed Absolute Value |
| PABSD | **mm** | mm/m64 |  |  | ssse3 |  | 0F | 38 | 1E | r | C2+ |  |  |  |  |  |  |  |  |  | Packed Absolute Value |
| PABSD | **xmm** | xmm/m128 |  |  | ssse3 | 66 | 0F | 38 | 1E | r | C2+ |  |  |  |  |  |  |  |  |  | Packed Absolute Value |
| PABSW | **mm** | mm/m64 |  |  | ssse3 |  | 0F | 38 | 1D | r | C2+ |  |  |  |  |  |  |  |  |  | Packed Absolute Value |
| PABSW | **xmm** | xmm/m128 |  |  | ssse3 | 66 | 0F | 38 | 1D | r | C2+ |  |  |  |  |  |  |  |  |  | Packed Absolute Value |
| PACKSSDW | **mm** | mm/m64 |  |  | mmx |  | 0F | 6B |  | r | PX+ |  |  |  |  |  |  |  |  |  | Pack with Signed Saturation |
| PACKSSDW | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 6B |  | r | P4+ |  |  |  |  |  |  |  |  |  | Pack with Signed Saturation |
| PACKSSWB | **mm** | mm/m64 |  |  | mmx |  | 0F | 63 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Pack with Signed Saturation |
| PACKSSWB | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 63 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Pack with Signed Saturation |
| PACKUSDW | **xmm** | xmm/m128 |  |  | sse41 | 66 | 0F | 38 | 2B | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Pack with Unsigned Saturation |
| PACKUSWB | **mm** | mm/m64 |  |  | mmx |  | 0F | 67 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Pack with Unsigned Saturation |
| PACKUSWB | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 67 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Pack with Unsigned Saturation |
| PADDB | **mm** | mm/m64 |  |  | mmx |  | 0F | FC |  | r | PX+ |  |  |  |  |  |  |  |  |  | Add Packed Integers |
| PADDB | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | FC |  | r | P4+ |  |  |  |  |  |  |  |  |  | Add Packed Integers |
| PADDD | **mm** | mm/m64 |  |  | mmx |  | 0F | FE |  | r | PX+ |  |  |  |  |  |  |  |  |  | Add Packed Integers |
| PADDD | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | FE |  | r | P4+ |  |  |  |  |  |  |  |  |  | Add Packed Integers |
| PADDQ | **mm** | mm/m64 |  |  | sse2 |  | 0F | D4 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Add Packed Quadword Integers |
| PADDQ | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | D4 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Add Packed Quadword Integers |
| PADDSB | **mm** | mm/m64 |  |  | mmx |  | 0F | EC |  | r | PX+ |  |  |  |  |  |  |  |  |  | Add Packed Signed Integers with Signed Saturation |
| PADDSB | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | EC |  | r | P4+ |  |  |  |  |  |  |  |  |  | Add Packed Signed Integers with Signed Saturation |
| PADDSW | **mm** | mm/m64 |  |  | mmx |  | 0F | ED |  | r | PX+ |  |  |  |  |  |  |  |  |  | Add Packed Signed Integers with Signed Saturation |
| PADDSW | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | ED |  | r | P4+ |  |  |  |  |  |  |  |  |  | Add Packed Signed Integers with Signed Saturation |
| PADDUSB | **mm** | mm/m64 |  |  | mmx |  | 0F | DC |  | r | PX+ |  |  |  |  |  |  |  |  |  | Add Packed Unsigned Integers with Unsigned Saturation |
| PADDUSB | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | DC |  | r | P4+ |  |  |  |  |  |  |  |  |  | Add Packed Unsigned Integers with Unsigned Saturation |
| PADDUSW | **mm** | mm/m64 |  |  | mmx |  | 0F | DD |  | r | PX+ |  |  |  |  |  |  |  |  |  | Add Packed Unsigned Integers with Unsigned Saturation |
| PADDUSW | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | DD |  | r | P4+ |  |  |  |  |  |  |  |  |  | Add Packed Unsigned Integers with Unsigned Saturation |
| PADDW | **mm** | mm/m64 |  |  | mmx |  | 0F | FD |  | r | PX+ |  |  |  |  |  |  |  |  |  | Add Packed Integers |
| PADDW | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | FD |  | r | P4+ |  |  |  |  |  |  |  |  |  | Add Packed Integers |
| PALIGNR | **mm** | mm/m64 |  |  | ssse3 |  | 0F | 3A | 0F | r | C2+ |  |  |  |  |  |  |  |  |  | Packed Align Right |
| PALIGNR | **xmm** | xmm/m128 |  |  | ssse3 | 66 | 0F | 3A | 0F | r | C2+ |  |  |  |  |  |  |  |  |  | Packed Align Right |
| PAND | **mm** | mm/m64 |  |  | mmx |  | 0F | DB |  | r | PX+ |  |  |  |  |  |  |  |  |  | Logical AND |
| PAND | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | DB |  | r | P4+ |  |  |  |  |  |  |  |  |  | Logical AND |
| PANDN | **mm** | mm/m64 |  |  | mmx |  | 0F | DF |  | r | PX+ |  |  |  |  |  |  |  |  |  | Logical AND NOT |
| PANDN | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | DF |  | r | P4+ |  |  |  |  |  |  |  |  |  | Logical AND NOT |
| PAUSE |  |  |  |  | sse2 | F3 |  | 90 |  |  | P4+ |  |  |  |  |  |  |  |  |  | Spin Loop Hint |
| PAVGB | **mm** | mm/m64 |  |  | sse1 |  | 0F | E0 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Average Packed Integers |
| PAVGB | **xmm** | xmm/m128 |  |  | sse1 | 66 | 0F | E0 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Average Packed Integers |
| PAVGW | **mm** | mm/m64 |  |  | sse1 |  | 0F | E3 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Average Packed Integers |
| PAVGW | **xmm** | xmm/m128 |  |  | sse1 | 66 | 0F | E3 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Average Packed Integers |
| PBLENDVB | **xmm** | xmm/m128 | *XMM0* |  | sse41 | 66 | 0F | 38 | 10 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Variable Blend Packed Bytes |
| PBLENDW | **xmm** | xmm/m128 | imm8 |  | sse41 | 66 | 0F | 3A | 0E | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Blend Packed Words |
| PCMPEQB | **mm** | mm/m64 |  |  | mmx |  | 0F | 74 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Compare Packed Data for Equal |
| PCMPEQB | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 74 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Compare Packed Data for Equal |
| PCMPEQD | **mm** | mm/m64 |  |  | mmx |  | 0F | 76 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Compare Packed Data for Equal |
| PCMPEQD | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 76 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Compare Packed Data for Equal |
| PCMPEQQ | **xmm** | xmm/m128 |  |  | sse41 | 66 | 0F | 38 | 29 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Compare Packed Qword Data for Equal |
| PCMPEQW | **mm** | mm/m64 |  |  | mmx |  | 0F | 75 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Compare Packed Data for Equal |
| PCMPEQW | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 75 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Compare Packed Data for Equal |
| PCMPESTRI | ***ECX*** | xmm | xmm/m128 | ... | sse42 | 66 | 0F | 3A | 61 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  | o..szapc | o..szapc |  | .....ap. | Packed Compare Explicit Length Strings, Return Index |
| PCMPESTRM | ***XMM0*** | xmm | xmm/m128 | ... | sse42 | 66 | 0F | 3A | 60 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  | o..szapc | o..szapc |  | .....ap. | Packed Compare Explicit Length Strings, Return Mask |
| PCMPGTB | **mm** | mm/m64 |  |  | mmx |  | 0F | 64 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Compare Packed Signed Integers for Greater Than |
| PCMPGTB | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 64 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Compare Packed Signed Integers for Greater Than |
| PCMPGTD | **mm** | mm/m64 |  |  | mmx |  | 0F | 66 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Compare Packed Signed Integers for Greater Than |
| PCMPGTD | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 66 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Compare Packed Signed Integers for Greater Than |
| PCMPGTQ | **xmm** | xmm/m128 |  |  | sse42 | 66 | 0F | 38 | 37 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Compare Packed Qword Data for Greater Than |
| PCMPGTW | **mm** | mm/m64 |  |  | mmx |  | 0F | 65 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Compare Packed Signed Integers for Greater Than |
| PCMPGTW | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 65 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Compare Packed Signed Integers for Greater Than |
| PCMPISTRI | ***ECX*** | xmm | xmm/m128 | imm8 | sse42 | 66 | 0F | 3A | 63 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  | o..szapc | o..szapc |  | .....ap. | Packed Compare Implicit Length Strings, Return Index |
| PCMPISTRM | ***XMM0*** | xmm | xmm/m128 | imm8 | sse42 | 66 | 0F | 3A | 62 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  | o..szapc | o..szapc |  | .....ap. | Packed Compare Implicit Length Strings, Return Mask |
| PEXTRB | **m8** | xmm | imm8 |  | sse41 | 66 | 0F | 3A | 14 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Extract Byte |
| PEXTRB | **r32** | xmm | imm8 |  |
| PEXTRD | **r/m32** | xmm | imm8 |  | sse41 | 66 | 0F | 3A | 16 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Extract Dword/Qword |
| PEXTRQ | **r/m64** | xmm | imm8 |  |
| PEXTRW | **m16** | xmm | imm8 |  | sse41 | 66 | 0F | 3A | 15 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Extract Word |
| PEXTRW | **r32** | xmm | imm8 |  |
| PEXTRW | **r32** | mm | imm8 |  | sse1 |  | 0F | C5 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Extract Word |
| PEXTRW | **r32** | xmm | imm8 |  | sse1 | 66 | 0F | C5 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Extract Word |
| PHADDD | **mm** | mm/m64 |  |  | ssse3 |  | 0F | 38 | 02 | r | C2+ |  |  |  |  |  |  |  |  |  | Packed Horizontal Add |
| PHADDD | **xmm** | xmm/m128 |  |  | ssse3 | 66 | 0F | 38 | 02 | r | C2+ |  |  |  |  |  |  |  |  |  | Packed Horizontal Add |
| PHADDSW | **mm** | mm/m64 |  |  | ssse3 |  | 0F | 38 | 03 | r | C2+ |  |  |  |  |  |  |  |  |  | Packed Horizontal Add and Saturate |
| PHADDSW | **xmm** | xmm/m128 |  |  | ssse3 | 66 | 0F | 38 | 03 | r | C2+ |  |  |  |  |  |  |  |  |  | Packed Horizontal Add and Saturate |
| PHADDW | **mm** | mm/m64 |  |  | ssse3 |  | 0F | 38 | 01 | r | C2+ |  |  |  |  |  |  |  |  |  | Packed Horizontal Add |
| PHADDW | **xmm** | xmm/m128 |  |  | ssse3 | 66 | 0F | 38 | 01 | r | C2+ |  |  |  |  |  |  |  |  |  | Packed Horizontal Add |
| PHMINPOSUW | **xmm** | xmm/m128 |  |  | sse41 | 66 | 0F | 38 | 41 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Packed Horizontal Word Minimum |
| PHSUBD | **mm** | mm/m64 |  |  | ssse3 |  | 0F | 38 | 06 | r | C2+ |  |  |  |  |  |  |  |  |  | Packed Horizontal Subtract |
| PHSUBD | **xmm** | xmm/m128 |  |  | ssse3 | 66 | 0F | 38 | 06 | r | C2+ |  |  |  |  |  |  |  |  |  | Packed Horizontal Subtract |
| PHSUBSW | **mm** | mm/m64 |  |  | ssse3 |  | 0F | 38 | 07 | r | C2+ |  |  |  |  |  |  |  |  |  | Packed Horizontal Subtract and Saturate |
| PHSUBSW | **xmm** | xmm/m128 |  |  | ssse3 | 66 | 0F | 38 | 07 | r | C2+ |  |  |  |  |  |  |  |  |  | Packed Horizontal Subtract and Saturate |
| PHSUBW | **mm** | mm/m64 |  |  | ssse3 |  | 0F | 38 | 05 | r | C2+ |  |  |  |  |  |  |  |  |  | Packed Horizontal Subtract |
| PHSUBW | **xmm** | xmm/m128 |  |  | ssse3 | 66 | 0F | 38 | 05 | r | C2+ |  |  |  |  |  |  |  |  |  | Packed Horizontal Subtract |
| PINSRB | **xmm** | m8 | imm8 |  | sse41 | 66 | 0F | 3A | 20 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Insert Byte |
| PINSRB | **xmm** | r32 | imm8 |  |
| PINSRD | **xmm** | r/m32 | imm8 |  | sse41 | 66 | 0F | 3A | 22 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Insert Dword/Qword |
| PINSRQ | **xmm** | r/m64 | imm8 |  |
| PINSRW | **mm** | r32 | imm8 |  | sse1 |  | 0F | C4 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Insert Word |
| PINSRW | **mm** | m16 | imm8 |  |
| PINSRW | **xmm** | r32 | imm8 |  | sse1 | 66 | 0F | C4 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Insert Word |
| PINSRW | **xmm** | m16 | imm8 |  |
| PMADDUBSW | **mm** | mm/m64 |  |  | ssse3 |  | 0F | 38 | 04 | r | C2+ |  |  |  |  |  |  |  |  |  | Multiply and Add Packed Signed and Unsigned Bytes |
| PMADDUBSW | **xmm** | xmm/m128 |  |  | ssse3 | 66 | 0F | 38 | 04 | r | C2+ |  |  |  |  |  |  |  |  |  | Multiply and Add Packed Signed and Unsigned Bytes |
| PMADDWD | **mm** | mm/m64 |  |  | mmx |  | 0F | F5 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Multiply and Add Packed Integers |
| PMADDWD | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | F5 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Multiply and Add Packed Integers |
| PMAXSB | **xmm** | xmm/m128 |  |  | sse41 | 66 | 0F | 38 | 3C | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Maximum of Packed Signed Byte Integers |
| PMAXSD | **xmm** | xmm/m128 |  |  | sse41 | 66 | 0F | 38 | 3D | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Maximum of Packed Signed Dword Integers |
| PMAXSW | **mm** | mm/m64 |  |  | sse1 |  | 0F | EE |  | r | P3+ |  |  |  |  |  |  |  |  |  | Maximum of Packed Signed Word Integers |
| PMAXSW | **xmm** | xmm/m128 |  |  | sse1 | 66 | 0F | EE |  | r | P3+ |  |  |  |  |  |  |  |  |  | Maximum of Packed Signed Word Integers |
| PMAXUB | **mm** | mm/m64 |  |  | sse1 |  | 0F | DE |  | r | P3+ |  |  |  |  |  |  |  |  |  | Maximum of Packed Unsigned Byte Integers |
| PMAXUB | **xmm** | xmm/m128 |  |  | sse1 | 66 | 0F | DE |  | r | P3+ |  |  |  |  |  |  |  |  |  | Maximum of Packed Unsigned Byte Integers |
| PMAXUD | **xmm** | xmm/m128 |  |  | sse41 | 66 | 0F | 38 | 3F | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Maximum of Packed Unsigned Dword Integers |
| PMAXUW | **xmm** | xmm/m128 |  |  | sse41 | 66 | 0F | 38 | 3E | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Maximum of Packed Unsigned Word Integers |
| PMINSB | **xmm** | xmm/m128 |  |  | sse41 | 66 | 0F | 38 | 38 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Minimum of Packed Signed Byte Integers |
| PMINSD | **xmm** | xmm/m128 |  |  | sse41 | 66 | 0F | 38 | 39 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Minimum of Packed Signed Dword Integers |
| PMINSW | **mm** | mm/m64 |  |  | sse1 |  | 0F | EA |  | r | P3+ |  |  |  |  |  |  |  |  |  | Minimum of Packed Signed Word Integers |
| PMINSW | **xmm** | xmm/m128 |  |  | sse1 | 66 | 0F | EA |  | r | P3+ |  |  |  |  |  |  |  |  |  | Minimum of Packed Signed Word Integers |
| PMINUB | **mm** | mm/m64 |  |  | sse1 |  | 0F | DA |  | r | P3+ |  |  |  |  |  |  |  |  |  | Minimum of Packed Unsigned Byte Integers |
| PMINUB | **xmm** | xmm/m128 |  |  | sse1 | 66 | 0F | DA |  | r | P3+ |  |  |  |  |  |  |  |  |  | Minimum of Packed Unsigned Byte Integers |
| PMINUD | **xmm** | xmm/m128 |  |  | sse41 | 66 | 0F | 38 | 3B | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Minimum of Packed Unsigned Dword Integers |
| PMINUW | **xmm** | xmm/m128 |  |  | sse41 | 66 | 0F | 38 | 3A | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Minimum of Packed Unsigned Word Integers |
| PMOVMSKB | **r32** | mm |  |  | sse1 |  | 0F | D7 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Move Byte Mask |
| PMOVMSKB | **r32** | xmm |  |  | sse1 | 66 | 0F | D7 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Move Byte Mask |
| PMOVSXBD | **xmm** | m32 |  |  | sse41 | 66 | 0F | 38 | 21 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Packed Move with Sign Extend |
| PMOVSXBD | **xmm** | xmm |  |  |
| PMOVSXBQ | **xmm** | m16 |  |  | sse41 | 66 | 0F | 38 | 22 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Packed Move with Sign Extend |
| PMOVSXBQ | **xmm** | xmm |  |  |
| PMOVSXBW | **xmm** | m64 |  |  | sse41 | 66 | 0F | 38 | 20 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Packed Move with Sign Extend |
| PMOVSXBW | **xmm** | xmm |  |  |
| PMOVSXDQ | **xmm** | m64 |  |  | sse41 | 66 | 0F | 38 | 25 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Packed Move with Sign Extend |
| PMOVSXDQ | **xmm** | xmm |  |  |
| PMOVSXWD | **xmm** | m64 |  |  | sse41 | 66 | 0F | 38 | 23 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Packed Move with Sign Extend |
| PMOVSXWD | **xmm** | xmm |  |  |
| PMOVSXWQ | **xmm** | m32 |  |  | sse41 | 66 | 0F | 38 | 24 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Packed Move with Sign Extend |
| PMOVSXWQ | **xmm** | xmm |  |  |
| PMOVZXBD | **xmm** | m32 |  |  | sse41 | 66 | 0F | 38 | 31 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Packed Move with Zero Extend |
| PMOVZXBD | **xmm** | xmm |  |  |
| PMOVZXBQ | **xmm** | m16 |  |  | sse41 | 66 | 0F | 38 | 32 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Packed Move with Zero Extend |
| PMOVZXBQ | **xmm** | xmm |  |  |
| PMOVZXBW | **xmm** | m64 |  |  | sse41 | 66 | 0F | 38 | 30 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Packed Move with Zero Extend |
| PMOVZXBW | **xmm** | xmm |  |  |
| PMOVZXDQ | **xmm** | m64 |  |  | sse41 | 66 | 0F | 38 | 35 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Packed Move with Zero Extend |
| PMOVZXDQ | **xmm** | xmm |  |  |
| PMOVZXWD | **xmm** | m64 |  |  | sse41 | 66 | 0F | 38 | 33 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Packed Move with Zero Extend |
| PMOVZXWD | **xmm** | xmm |  |  |
| PMOVZXWQ | **xmm** | m32 |  |  | sse41 | 66 | 0F | 38 | 34 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Packed Move with Zero Extend |
| PMOVZXWQ | **xmm** | xmm |  |  |
| PMULDQ | **xmm** | xmm/m128 |  |  | sse41 | 66 | 0F | 38 | 28 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Multiply Packed Signed Dword Integers |
| PMULHRSW | **mm** | mm/m64 |  |  | ssse3 |  | 0F | 38 | 0B | r | C2+ |  |  |  |  |  |  |  |  |  | Packed Multiply High with Round and Scale |
| PMULHRSW | **xmm** | xmm/m128 |  |  | ssse3 | 66 | 0F | 38 | 0B | r | C2+ |  |  |  |  |  |  |  |  |  | Packed Multiply High with Round and Scale |
| PMULHUW | **mm** | mm/m64 |  |  | sse1 |  | 0F | E4 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Multiply Packed Unsigned Integers and Store High Result |
| PMULHUW | **xmm** | xmm/m128 |  |  | sse1 | 66 | 0F | E4 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Multiply Packed Unsigned Integers and Store High Result |
| PMULHW | **mm** | mm/m64 |  |  | mmx |  | 0F | E5 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Multiply Packed Signed Integers and Store High Result |
| PMULHW | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | E5 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Multiply Packed Signed Integers and Store High Result |
| PMULLD | **xmm** | xmm/m128 |  |  | sse41 | 66 | 0F | 38 | 40 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Multiply Packed Signed Dword Integers and Store Low Result |
| PMULLW | **mm** | mm/m64 |  |  | mmx |  | 0F | D5 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Multiply Packed Signed Integers and Store Low Result |
| PMULLW | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | D5 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Multiply Packed Signed Integers and Store Low Result |
| PMULUDQ | **mm** | mm/m64 |  |  | sse2 |  | 0F | F4 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Multiply Packed Unsigned DW Integers |
| PMULUDQ | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | F4 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Multiply Packed Unsigned DW Integers |
| POP | **ES** |  |  |  |  |  |  | 07 |  |  |  |  |  |  |  |  |  |  |  |  | Pop a Value from the Stack |
| POP | **SS** |  |  |  |  |  |  | 17 |  |  |  |  |  |  |  |  |  |  |  |  | Pop a Value from the Stack |
| POP | **DS** |  |  |  |  |  |  | 1F |  |  |  |  |  |  |  |  |  |  |  |  | Pop a Value from the Stack |
| POP | **r16/32** |  |  |  |  |  |  | 58+r | |  |  |  |  |  |  |  |  |  |  |  | Pop a Value from the Stack |
| POP | **r/m16/32** |  |  |  |  |  |  | 8F |  | 0 |  |  |  |  |  |  |  |  |  |  | Pop a Value from the Stack |
| POP | **FS** |  |  |  |  |  | 0F | A1 |  |  | 03+ |  |  |  |  |  |  |  |  |  | Pop a Value from the Stack |
| POP | **GS** |  |  |  |  |  | 0F | A9 |  |  | 03+ |  |  |  |  |  |  |  |  |  | Pop a Value from the Stack |
| POPA | ***DI*** | ***SI*** | ***BP*** | ... |  |  |  | 61 |  |  | 01+ |  |  |  |  |  |  |  |  |  | Pop All General-Purpose Registers |
| POPAD | ***EDI*** | ***ESI*** | ***EBP*** | ... |  |  |  | 61 |  |  | 03+ |  |  |  |  |  |  |  |  |  | Pop All General-Purpose Registers |
| POPCNT | **r16/32** | r/m16/32 |  |  |  | F3 | 0F | B8 |  | r | C2++ |  |  |  |  |  | o..szapc |  |  | o..s.apc | Bit Population Count |
| POPF | ***Flags*** |  |  |  |  |  |  | 9D |  |  |  |  |  |  |  |  |  |  |  |  | Pop Stack into FLAGS Register |
| POPFD | ***EFlags*** |  |  |  |  |  |  | 9D |  |  | 03+ |  |  |  |  |  |  |  |  |  | Pop Stack into eFLAGS Register |
| POR | **mm** | mm/m64 |  |  | mmx |  | 0F | EB |  | r | PX+ |  |  |  |  |  |  |  |  |  | Bitwise Logical OR |
| POR | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | EB |  | r | P4+ |  |  |  |  |  |  |  |  |  | Bitwise Logical OR |
| PREFETCHNTA | m8 |  |  |  | sse1 |  | 0F | 18 |  | 0 | P3+ |  |  |  |  |  |  |  |  |  | Prefetch Data Into Caches |
| PREFETCHT0 | m8 |  |  |  | sse1 |  | 0F | 18 |  | 1 | P3+ |  |  |  |  |  |  |  |  |  | Prefetch Data Into Caches |
| PREFETCHT1 | m8 |  |  |  | sse1 |  | 0F | 18 |  | 2 | P3+ |  |  |  |  |  |  |  |  |  | Prefetch Data Into Caches |
| PREFETCHT2 | m8 |  |  |  | sse1 |  | 0F | 18 |  | 3 | P3+ |  |  |  |  |  |  |  |  |  | Prefetch Data Into Caches |
| PSADBW | **mm** | mm/m64 |  |  | sse1 |  | 0F | F6 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Compute Sum of Absolute Differences |
| PSADBW | **xmm** | xmm/m128 |  |  | sse1 | 66 | 0F | F6 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Compute Sum of Absolute Differences |
| PSHUFB | **mm** | mm/m64 |  |  | ssse3 |  | 0F | 38 | 00 | r | C2+ |  |  |  |  |  |  |  |  |  | Packed Shuffle Bytes |
| PSHUFB | **xmm** | xmm/m128 |  |  | ssse3 | 66 | 0F | 38 | 00 | r | C2+ |  |  |  |  |  |  |  |  |  | Packed Shuffle Bytes |
| PSHUFD | **xmm** | xmm/m128 | imm8 |  | sse2 | 66 | 0F | 70 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Shuffle Packed Doublewords |
| PSHUFHW | **xmm** | xmm/m128 | imm8 |  | sse2 | F3 | 0F | 70 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Shuffle Packed High Words |
| PSHUFLW | **xmm** | xmm/m128 | imm8 |  | sse2 | F2 | 0F | 70 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Shuffle Packed Low Words |
| PSHUFW | **mm** | mm/m64 | imm8 |  | sse1 |  | 0F | 70 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Shuffle Packed Words |
| PSIGNB | **mm** | mm/m64 |  |  | ssse3 |  | 0F | 38 | 08 | r | C2+ |  |  |  |  |  |  |  |  |  | Packed SIGN |
| PSIGNB | **xmm** | xmm/m128 |  |  | ssse3 | 66 | 0F | 38 | 08 | r | C2+ |  |  |  |  |  |  |  |  |  | Packed SIGN |
| PSIGND | **mm** | mm/m64 |  |  | ssse3 |  | 0F | 38 | 0A | r | C2+ |  |  |  |  |  |  |  |  |  | Packed SIGN |
| PSIGND | **xmm** | xmm/m128 |  |  | ssse3 | 66 | 0F | 38 | 0A | r | C2+ |  |  |  |  |  |  |  |  |  | Packed SIGN |
| PSIGNW | **mm** | mm/m64 |  |  | ssse3 |  | 0F | 38 | 09 | r | C2+ |  |  |  |  |  |  |  |  |  | Packed SIGN |
| PSIGNW | **xmm** | xmm/m128 |  |  | ssse3 | 66 | 0F | 38 | 09 | r | C2+ |  |  |  |  |  |  |  |  |  | Packed SIGN |
| PSLLD | **mm** | imm8 |  |  | mmx |  | 0F | 72 |  | 6 | PX+ |  |  |  |  |  |  |  |  |  | Shift Packed Data Left Logical |
| PSLLD | **xmm** | imm8 |  |  | sse2 | 66 | 0F | 72 |  | 6 | P4+ |  |  |  |  |  |  |  |  |  | Shift Packed Data Left Logical |
| PSLLD | **mm** | mm/m64 |  |  | mmx |  | 0F | F2 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Shift Packed Data Left Logical |
| PSLLD | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | F2 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Shift Packed Data Left Logical |
| PSLLDQ | **xmm** | imm8 |  |  | sse2 | 66 | 0F | 73 |  | 7 | P4+ |  |  |  |  |  |  |  |  |  | Shift Double Quadword Left Logical |
| PSLLQ | **mm** | imm8 |  |  | mmx |  | 0F | 73 |  | 6 | PX+ |  |  |  |  |  |  |  |  |  | Shift Packed Data Left Logical |
| PSLLQ | **xmm** | imm8 |  |  | sse2 | 66 | 0F | 73 |  | 6 | P4+ |  |  |  |  |  |  |  |  |  | Shift Packed Data Left Logical |
| PSLLQ | **mm** | mm/m64 |  |  | mmx |  | 0F | F3 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Shift Packed Data Left Logical |
| PSLLQ | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | F3 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Shift Packed Data Left Logical |
| PSLLW | **mm** | imm8 |  |  | mmx |  | 0F | 71 |  | 6 | PX+ |  |  |  |  |  |  |  |  |  | Shift Packed Data Left Logical |
| PSLLW | **xmm** | imm8 |  |  | sse2 | 66 | 0F | 71 |  | 6 | P4+ |  |  |  |  |  |  |  |  |  | Shift Packed Data Left Logical |
| PSLLW | **mm** | mm/m64 |  |  | mmx |  | 0F | F1 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Shift Packed Data Left Logical |
| PSLLW | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | F1 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Shift Packed Data Left Logical |
| PSRAD | **mm** | imm8 |  |  | mmx |  | 0F | 72 |  | 4 | PX+ |  |  |  |  |  |  |  |  |  | Shift Packed Data Right Arithmetic |
| PSRAD | **xmm** | imm8 |  |  | sse2 | 66 | 0F | 72 |  | 4 | P4+ |  |  |  |  |  |  |  |  |  | Shift Packed Data Right Arithmetic |
| PSRAD | **mm** | mm/m64 |  |  | mmx |  | 0F | E2 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Shift Packed Data Right Arithmetic |
| PSRAD | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | E2 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Shift Packed Data Right Arithmetic |
| PSRAW | **mm** | imm8 |  |  | mmx |  | 0F | 71 |  | 4 | PX+ |  |  |  |  |  |  |  |  |  | Shift Packed Data Right Arithmetic |
| PSRAW | **xmm** | imm8 |  |  | sse2 | 66 | 0F | 71 |  | 4 | P4+ |  |  |  |  |  |  |  |  |  | Shift Packed Data Right Arithmetic |
| PSRAW | **mm** | mm/m64 |  |  | mmx |  | 0F | E1 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Shift Packed Data Right Arithmetic |
| PSRAW | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | E1 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Shift Packed Data Right Arithmetic |
| PSRLD | **mm** | imm8 |  |  | mmx |  | 0F | 72 |  | 2 | PX+ |  |  |  |  |  |  |  |  |  | Shift Double Quadword Right Logical |
| PSRLD | **xmm** | imm8 |  |  | sse2 | 66 | 0F | 72 |  | 2 | P4+ |  |  |  |  |  |  |  |  |  | Shift Double Quadword Right Logical |
| PSRLD | **mm** | mm/m64 |  |  | mmx |  | 0F | D2 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Shift Packed Data Right Logical |
| PSRLD | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | D2 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Shift Packed Data Right Logical |
| PSRLDQ | **xmm** | imm8 |  |  | sse2 | 66 | 0F | 73 |  | 3 | P4+ |  |  |  |  |  |  |  |  |  | Shift Double Quadword Right Logical |
| PSRLQ | **mm** | imm8 |  |  | mmx |  | 0F | 73 |  | 2 | PX+ |  |  |  |  |  |  |  |  |  | Shift Packed Data Right Logical |
| PSRLQ | **xmm** | imm8 |  |  | sse2 | 66 | 0F | 73 |  | 2 | P4+ |  |  |  |  |  |  |  |  |  | Shift Packed Data Right Logical |
| PSRLQ | **mm** | mm/m64 |  |  | mmx |  | 0F | D3 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Shift Packed Data Right Logical |
| PSRLQ | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | D3 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Shift Packed Data Right Logical |
| PSRLW | **mm** | imm8 |  |  | mmx |  | 0F | 71 |  | 2 | PX+ |  |  |  |  |  |  |  |  |  | Shift Packed Data Right Logical |
| PSRLW | **xmm** | imm8 |  |  | sse2 | 66 | 0F | 71 |  | 2 | P4+ |  |  |  |  |  |  |  |  |  | Shift Packed Data Right Logical |
| PSRLW | **mm** | mm/m64 |  |  | mmx |  | 0F | D1 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Shift Packed Data Right Logical |
| PSRLW | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | D1 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Shift Packed Data Right Logical |
| PSUBB | **mm** | mm/m64 |  |  | mmx |  | 0F | F8 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Subtract Packed Integers |
| PSUBB | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | F8 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Subtract Packed Integers |
| PSUBD | **mm** | mm/m64 |  |  | mmx |  | 0F | FA |  | r | PX+ |  |  |  |  |  |  |  |  |  | Subtract Packed Integers |
| PSUBD | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | FA |  | r | P4+ |  |  |  |  |  |  |  |  |  | Subtract Packed Integers |
| PSUBQ | **mm** | mm/m64 |  |  | sse2 |  | 0F | FB |  | r | P4+ |  |  |  |  |  |  |  |  |  | Subtract Packed Quadword Integers |
| PSUBQ | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | FB |  | r | P4+ |  |  |  |  |  |  |  |  |  | Subtract Packed Quadword Integers |
| PSUBSB | **mm** | mm/m64 |  |  | mmx |  | 0F | E8 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Subtract Packed Signed Integers with Signed Saturation |
| PSUBSB | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | E8 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Subtract Packed Signed Integers with Signed Saturation |
| PSUBSW | **mm** | mm/m64 |  |  | mmx |  | 0F | E9 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Subtract Packed Signed Integers with Signed Saturation |
| PSUBSW | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | E9 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Subtract Packed Signed Integers with Signed Saturation |
| PSUBUSB | **mm** | mm/m64 |  |  | mmx |  | 0F | D8 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Subtract Packed Unsigned Integers with Unsigned Saturation |
| PSUBUSB | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | D8 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Subtract Packed Unsigned Integers with Unsigned Saturation |
| PSUBUSW | **mm** | mm/m64 |  |  | mmx |  | 0F | D9 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Subtract Packed Unsigned Integers with Unsigned Saturation |
| PSUBUSW | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | D9 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Subtract Packed Unsigned Integers with Unsigned Saturation |
| PSUBW | **mm** | mm/m64 |  |  | mmx |  | 0F | F9 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Subtract Packed Integers |
| PSUBW | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | F9 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Subtract Packed Integers |
| PTEST | xmm | xmm/m128 |  |  | sse41 | 66 | 0F | 38 | 17 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  | o..szapc | o..szapc |  | o..s.ap. | Logical Compare |
| PUNPCKHBW | **mm** | mm/m64 |  |  | mmx |  | 0F | 68 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Unpack High Data |
| PUNPCKHBW | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 68 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Unpack High Data |
| PUNPCKHDQ | **mm** | mm/m64 |  |  | mmx |  | 0F | 6A |  | r | PX+ |  |  |  |  |  |  |  |  |  | Unpack High Data |
| PUNPCKHDQ | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 6A |  | r | P4+ |  |  |  |  |  |  |  |  |  | Unpack High Data |
| PUNPCKHQDQ | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 6D |  | r | P4+ |  |  |  |  |  |  |  |  |  | Unpack High Data |
| PUNPCKHWD | **mm** | mm/m64 |  |  | mmx |  | 0F | 69 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Unpack High Data |
| PUNPCKHWD | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 69 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Unpack High Data |
| PUNPCKLBW | **mm** | mm/m64 |  |  | mmx |  | 0F | 60 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Unpack Low Data |
| PUNPCKLBW | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 60 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Unpack Low Data |
| PUNPCKLDQ | **mm** | mm/m64 |  |  | mmx |  | 0F | 62 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Unpack Low Data |
| PUNPCKLDQ | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 62 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Unpack Low Data |
| PUNPCKLQDQ | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 6C |  | r | P4+ |  |  |  |  |  |  |  |  |  | Unpack Low Data |
| PUNPCKLWD | **mm** | mm/m64 |  |  | mmx |  | 0F | 61 |  | r | PX+ |  |  |  |  |  |  |  |  |  | Unpack Low Data |
| PUNPCKLWD | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 61 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Unpack Low Data |
| PUSH | ES |  |  |  |  |  |  | 06 |  |  |  |  |  |  |  |  |  |  |  |  | Push Word, Doubleword or Quadword Onto the Stack |
| PUSH | CS |  |  |  |  |  |  | 0E |  |  |  |  |  |  |  |  |  |  |  |  | Push Word, Doubleword or Quadword Onto the Stack |
| PUSH | SS |  |  |  |  |  |  | 16 |  |  |  |  |  |  |  |  |  |  |  |  | Push Word, Doubleword or Quadword Onto the Stack |
| PUSH | DS |  |  |  |  |  |  | 1E |  |  |  |  |  |  |  |  |  |  |  |  | Push Word, Doubleword or Quadword Onto the Stack |
| PUSH | r16/32 |  |  |  |  |  |  | 50+r | |  |  |  |  |  |  |  |  |  |  |  | Push Word, Doubleword or Quadword Onto the Stack |
| PUSH | imm16/32 |  |  |  |  |  |  | 68 |  |  | 01+ |  |  |  |  |  |  |  |  |  | Push Word, Doubleword or Quadword Onto the Stack |
| PUSH | imm8 |  |  |  |  |  |  | 6A |  |  | 01+ |  |  |  |  |  |  |  |  |  | Push Word, Doubleword or Quadword Onto the Stack |
| PUSH | r/m16/32 |  |  |  |  |  |  | FF |  | 6 |  |  |  |  |  |  |  |  |  |  | Push Word, Doubleword or Quadword Onto the Stack |
| PUSH | FS |  |  |  |  |  | 0F | A0 |  |  | 03+ |  |  |  |  |  |  |  |  |  | Push Word, Doubleword or Quadword Onto the Stack |
| PUSH | GS |  |  |  |  |  | 0F | A8 |  |  | 03+ |  |  |  |  |  |  |  |  |  | Push Word, Doubleword or Quadword Onto the Stack |
| PUSHA | *AX* | *CX* | *DX* | ... |  |  |  | 60 |  |  | 01+ |  |  |  |  |  |  |  |  |  | Push All General-Purpose Registers |
| PUSHAD | *EAX* | *ECX* | *EDX* | ... |  |  |  | 60 |  |  | 03+ |  |  |  |  |  |  |  |  |  | Push All General-Purpose Registers |
| PUSHF | *Flags* |  |  |  |  |  |  | 9C |  |  |  |  |  |  |  |  |  |  |  |  | Push FLAGS Register onto the Stack |
| PUSHFD | *EFlags* |  |  |  |  |  |  | 9C |  |  | 03+ |  |  |  |  |  |  |  |  |  | Push eFLAGS Register onto the Stack |
| PXOR | **mm** | mm/m64 |  |  | mmx |  | 0F | EF |  | r | PX+ |  |  |  |  |  |  |  |  |  | Logical Exclusive OR |
| PXOR | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | EF |  | r | P4+ |  |  |  |  |  |  |  |  |  | Logical Exclusive OR |
| RCL | **r/m8** | imm8 |  |  |  |  |  | C0 |  | 2 | 01+ |  |  |  |  | .......c | o..szapc | o..szapc | o....... |  | Rotate |
| RCL | **r/m16/32** | imm8 |  |  |  |  |  | C1 |  | 2 | 01+ |  |  |  |  | .......c | o..szapc | o..szapc | o....... |  | Rotate |
| RCL | **r/m8** | 1 |  |  |  |  |  | D0 |  | 2 |  |  |  |  |  | .......c | o..szapc | o..szapc |  |  | Rotate |
| RCL | **r/m16/32** | 1 |  |  |  |  |  | D1 |  | 2 |  |  |  |  |  | .......c | o..szapc | o..szapc |  |  | Rotate |
| RCL | **r/m8** | CL |  |  |  |  |  | D2 |  | 2 |  |  |  |  |  | .......c | o..szapc | o..szapc | o....... |  | Rotate |
| RCL | **r/m16/32** | CL |  |  |  |  |  | D3 |  | 2 |  |  |  |  |  | .......c | o..szapc | o..szapc | o....... |  | Rotate |
| RCPPS | **xmm** | xmm/m128 |  |  | sse1 |  | 0F | 53 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Compute Reciprocals of Packed Single-FP Values |
| RCPSS | **xmm** | xmm/m32 |  |  | sse1 | F3 | 0F | 53 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Compute Reciprocal of Scalar Single-FP Values |
| RCR | **r/m8** | imm8 |  |  |  |  |  | C0 |  | 3 | 01+ |  |  |  |  | .......c | o..szapc | o..szapc | o....... |  | Rotate |
| RCR | **r/m16/32** | imm8 |  |  |  |  |  | C1 |  | 3 | 01+ |  |  |  |  | .......c | o..szapc | o..szapc | o....... |  | Rotate |
| RCR | **r/m8** | 1 |  |  |  |  |  | D0 |  | 3 |  |  |  |  |  | .......c | o..szapc | o..szapc |  |  | Rotate |
| RCR | **r/m16/32** | 1 |  |  |  |  |  | D1 |  | 3 |  |  |  |  |  | .......c | o..szapc | o..szapc |  |  | Rotate |
| RCR | **r/m8** | CL |  |  |  |  |  | D2 |  | 3 |  |  |  |  |  | .......c | o..szapc | o..szapc | o....... |  | Rotate |
| RCR | **r/m16/32** | CL |  |  |  |  |  | D3 |  | 3 |  |  |  |  |  | .......c | o..szapc | o..szapc | o....... |  | Rotate |
| RDMSR | ***EAX*** | ***EDX*** | *ECX* | *MSR* |  |  | 0F | 32 |  |  | P1+ |  |  | 0 |  |  |  |  |  |  | Read from Model Specific Register |
| RDPMC | ***EAX*** | ***EDX*** | *PMC* |  |  |  | 0F | 33 |  |  | PX+ |  |  | f[3](http://ref.x86asm.net/coder32-abc.html#cr4_pce) |  |  |  |  |  |  | Read Performance-Monitoring Counters |
| RDTSC | ***EAX*** | ***EDX*** | *IA32\_TIM…* |  |  |  | 0F | 31 |  |  | P1+ |  |  | f[2](http://ref.x86asm.net/coder32-abc.html#cr4_tsd) |  |  |  |  |  |  | Read Time-Stamp Counter |
| RDTSCP | ***EAX*** | ***EDX*** | ***ECX*** | ... |  |  | 0F | 01 | F9 | 7 | C7+ |  |  | f[2](http://ref.x86asm.net/coder32-abc.html#cr4_tsd) |  |  |  |  |  |  | Read Time-Stamp Counter and Processor ID |
| REP | ***eCX*** |  |  |  |  | F2 |  |  |  |  |  | U[10](http://ref.x86asm.net/coder32-abc.html#gen_note_REP_F2_F3) |  |  |  |  |  |  |  |  | Repeat String Operation Prefix |
| REP | ***eCX*** |  |  |  |  | F3 |  |  |  |  |  | D[10](http://ref.x86asm.net/coder32-abc.html#gen_note_REP_F2_F3) |  |  |  |  |  |  |  |  | Repeat String Operation Prefix |
| REPNZ | ***eCX*** |  |  |  |  | F2 |  |  |  |  |  | D[10](http://ref.x86asm.net/coder32-abc.html#gen_note_REP_F2_F3) |  |  |  | ....z... |  |  |  |  | Repeat String Operation Prefix |
| REPNE | ***eCX*** |  |  |  |
| REPZ | ***eCX*** |  |  |  |  | F3 |  |  |  |  |  | D[10](http://ref.x86asm.net/coder32-abc.html#gen_note_REP_F2_F3) |  |  |  | ....z... |  |  |  |  | Repeat String Operation Prefix |
| REPE | ***eCX*** |  |  |  |
| RETF | imm16 |  |  |  |  |  |  | CA |  |  |  |  |  | f |  |  |  |  |  |  | Return from procedure |
| RETF |  |  |  |  |  |  |  | CB |  |  |  |  |  | f |  |  |  |  |  |  | Return from procedure |
| RETN | imm16 |  |  |  |  |  |  | C2 |  |  |  |  |  |  |  |  |  |  |  |  | Return from procedure |
| RETN |  |  |  |  |  |  |  | C3 |  |  |  |  |  |  |  |  |  |  |  |  | Return from procedure |
| ROL | **r/m8** | imm8 |  |  |  |  |  | C0 |  | 0 | 01+ |  |  |  |  |  | o..szapc | o..szapc | o....... |  | Rotate |
| ROL | **r/m16/32** | imm8 |  |  |  |  |  | C1 |  | 0 | 01+ |  |  |  |  |  | o..szapc | o..szapc | o....... |  | Rotate |
| ROL | **r/m8** | 1 |  |  |  |  |  | D0 |  | 0 |  |  |  |  |  |  | o..szapc | o..szapc |  |  | Rotate |
| ROL | **r/m16/32** | 1 |  |  |  |  |  | D1 |  | 0 |  |  |  |  |  |  | o..szapc | o..szapc |  |  | Rotate |
| ROL | **r/m8** | CL |  |  |  |  |  | D2 |  | 0 |  |  |  |  |  |  | o..szapc | o..szapc | o....... |  | Rotate |
| ROL | **r/m16/32** | CL |  |  |  |  |  | D3 |  | 0 |  |  |  |  |  |  | o..szapc | o..szapc | o....... |  | Rotate |
| ROR | **r/m8** | imm8 |  |  |  |  |  | C0 |  | 1 | 01+ |  |  |  |  |  | o..szapc | o..szapc | o....... |  | Rotate |
| ROR | **r/m16/32** | imm8 |  |  |  |  |  | C1 |  | 1 | 01+ |  |  |  |  |  | o..szapc | o..szapc | o....... |  | Rotate |
| ROR | **r/m8** | 1 |  |  |  |  |  | D0 |  | 1 |  |  |  |  |  |  | o..szapc | o..szapc |  |  | Rotate |
| ROR | **r/m16/32** | 1 |  |  |  |  |  | D1 |  | 1 |  |  |  |  |  |  | o..szapc | o..szapc |  |  | Rotate |
| ROR | **r/m8** | CL |  |  |  |  |  | D2 |  | 1 |  |  |  |  |  |  | o..szapc | o..szapc | o....... |  | Rotate |
| ROR | **r/m16/32** | CL |  |  |  |  |  | D3 |  | 1 |  |  |  |  |  |  | o..szapc | o..szapc | o....... |  | Rotate |
| ROUNDPD | **xmm** | xmm/m128 | imm8 |  | sse41 | 66 | 0F | 3A | 09 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Round Packed Double-FP Values |
| ROUNDPS | **xmm** | xmm/m128 | imm8 |  | sse41 | 66 | 0F | 3A | 08 | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Round Packed Single-FP Values |
| ROUNDSD | **xmm** | xmm/m64 | imm8 |  | sse41 | 66 | 0F | 3A | 0B | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Round Scalar Double-FP Values |
| ROUNDSS | **xmm** | xmm/m32 | imm8 |  | sse41 | 66 | 0F | 3A | 0A | r | C2++ | D[24](http://ref.x86asm.net/coder32-abc.html#gen_note_SSE4_amd) |  |  |  |  |  |  |  |  | Round Scalar Single-FP Values |
| RSM | ***Flags*** |  |  |  |  |  | 0F | AA |  |  | 03++ |  | S |  |  |  |  |  |  |  | Resume from System Management Mode |
| RSQRTPS | **xmm** | xmm/m128 |  |  | sse1 |  | 0F | 52 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Compute Recipr. of Square Roots of Packed Single-FP Values |
| RSQRTSS | **xmm** | xmm/m32 |  |  | sse1 | F3 | 0F | 52 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Compute Recipr. of Square Root of Scalar Single-FP Value |
| SAHF | *AH* |  |  |  |  |  |  | 9E |  |  |  |  |  |  |  |  | ...szapc | ...szapc |  |  | Store AH into Flags |
| SAL | **r/m8** | imm8 |  |  |  |  |  | C0 |  | 6 | 01+ | U[2](http://ref.x86asm.net/coder32-abc.html#gen_note_SAL_C0_4_C1_4_D0_4_D1_4) |  |  |  |  | o..szapc | o..sz.pc | o....a.c |  | Shift |
| SHL | **r/m8** | imm8 |  |  |
| SAL | **r/m16/32** | imm8 |  |  |  |  |  | C1 |  | 6 | 01+ | U[2](http://ref.x86asm.net/coder32-abc.html#gen_note_SAL_C0_4_C1_4_D0_4_D1_4) |  |  |  |  | o..szapc | o..sz.pc | o....a.c |  | Shift |
| SHL | **r/m16/32** | imm8 |  |  |
| SAL | **r/m8** | 1 |  |  |  |  |  | D0 |  | 6 |  | U[2](http://ref.x86asm.net/coder32-abc.html#gen_note_SAL_C0_4_C1_4_D0_4_D1_4) |  |  |  |  | o..szapc | o..sz.pc | .....a.. |  | Shift |
| SHL | **r/m8** | 1 |  |  |
| SAL | **r/m16/32** | 1 |  |  |  |  |  | D1 |  | 6 |  | U[2](http://ref.x86asm.net/coder32-abc.html#gen_note_SAL_C0_4_C1_4_D0_4_D1_4) |  |  |  |  | o..szapc | o..sz.pc | .....a.. |  | Shift |
| SHL | **r/m16/32** | 1 |  |  |
| SAL | **r/m8** | CL |  |  |  |  |  | D2 |  | 6 |  | U[2](http://ref.x86asm.net/coder32-abc.html#gen_note_SAL_C0_4_C1_4_D0_4_D1_4) |  |  |  |  | o..szapc | o..sz.pc | o....a.c |  | Shift |
| SHL | **r/m8** | CL |  |  |
| SAL | **r/m16/32** | CL |  |  |  |  |  | D3 |  | 6 |  | U[2](http://ref.x86asm.net/coder32-abc.html#gen_note_SAL_C0_4_C1_4_D0_4_D1_4) |  |  |  |  | o..szapc | o..sz.pc | o....a.c |  | Shift |
| SHL | **r/m16/32** | CL |  |  |
| SALC | ***AL*** |  |  |  |  |  |  | D6 |  |  | 02+ | U[3](http://ref.x86asm.net/coder32-abc.html#gen_note_u_SALC_D6) |  |  |  | .......c |  |  |  |  | Set AL If Carry |
| SETALC | ***AL*** |  |  |  |
| SAR | **r/m8** | imm8 |  |  |  |  |  | C0 |  | 7 | 01+ |  |  |  |  |  | o..szapc | o..sz.pc | o....a.. |  | Shift |
| SAR | **r/m16/32** | imm8 |  |  |  |  |  | C1 |  | 7 | 01+ |  |  |  |  |  | o..szapc | o..sz.pc | o....a.. |  | Shift |
| SAR | **r/m8** | 1 |  |  |  |  |  | D0 |  | 7 |  |  |  |  |  |  | o..szapc | o..sz.pc | .....a.. |  | Shift |
| SAR | **r/m16/32** | 1 |  |  |  |  |  | D1 |  | 7 |  |  |  |  |  |  | o..szapc | o..sz.pc | .....a.. |  | Shift |
| SAR | **r/m8** | CL |  |  |  |  |  | D2 |  | 7 |  |  |  |  |  |  | o..szapc | o..sz.pc | o....a.. |  | Shift |
| SAR | **r/m16/32** | CL |  |  |  |  |  | D3 |  | 7 |  |  |  |  |  |  | o..szapc | o..sz.pc | .....a.. |  | Shift |
| SBB | **r/m8** | r8 |  |  |  |  |  | 18 |  | r |  |  |  |  | L | .......c | o..szapc | o..szapc |  |  | Integer Subtraction with Borrow |
| SBB | **r/m16/32** | r16/32 |  |  |  |  |  | 19 |  | r |  |  |  |  | L | .......c | o..szapc | o..szapc |  |  | Integer Subtraction with Borrow |
| SBB | **r8** | r/m8 |  |  |  |  |  | 1A |  | r |  |  |  |  |  | .......c | o..szapc | o..szapc |  |  | Integer Subtraction with Borrow |
| SBB | **r16/32** | r/m16/32 |  |  |  |  |  | 1B |  | r |  |  |  |  |  | .......c | o..szapc | o..szapc |  |  | Integer Subtraction with Borrow |
| SBB | **AL** | imm8 |  |  |  |  |  | 1C |  |  |  |  |  |  |  | .......c | o..szapc | o..szapc |  |  | Integer Subtraction with Borrow |
| SBB | **eAX** | imm16/32 |  |  |  |  |  | 1D |  |  |  |  |  |  |  | .......c | o..szapc | o..szapc |  |  | Integer Subtraction with Borrow |
| SBB | **r/m8** | imm8 |  |  |  |  |  | 80 |  | 3 |  |  |  |  | L | .......c | o..szapc | o..szapc |  |  | Integer Subtraction with Borrow |
| SBB | **r/m16/32** | imm16/32 |  |  |  |  |  | 81 |  | 3 |  |  |  |  | L | .......c | o..szapc | o..szapc |  |  | Integer Subtraction with Borrow |
| SBB | **r/m8** | imm8 |  |  |  |  |  | 82 |  | 3 |  |  |  |  | L | .......c | o..szapc | o..szapc |  |  | Integer Subtraction with Borrow |
| SBB | **r/m16/32** | imm8 |  |  |  |  |  | 83 |  | 3 |  |  |  |  | L | .......c | o..szapc | o..szapc |  |  | Integer Subtraction with Borrow |
| SCAS | m8 | *AL* |  |  |  |  |  | AE |  |  |  |  |  |  |  | .d...... | o..szapc | o..szapc |  |  | Scan String |
| SCASB | *m8* | *AL* |  |  |
| SCAS | m16 | *AX* |  |  |  |  |  | AF |  |  |  |  |  |  |  | .d...... | o..szapc | o..szapc |  |  | Scan String |
| SCASW | *m16* | *AX* |  |  |
| SCAS | m16/32 | *eAX* |  |  |  |  |  | AF |  |  | 03+ |  |  |  |  | .d...... | o..szapc | o..szapc |  |  | Scan String |
| SCASD | *m32* | *EAX* |  |  |
| SETB | **r/m8** |  |  |  |  |  | 0F | 92 |  | 0 | 03+ | D[17](http://ref.x86asm.net/coder32-abc.html#gen_note_SETcc_0F90-0F9F) |  |  |  | .......c |  |  |  |  | Set Byte on Condition - below/not above or equal/carry (CF=1) |
| SETNAE | **r/m8** |  |  |  |
| SETC | **r/m8** |  |  |  |
| SETBE | **r/m8** |  |  |  |  |  | 0F | 96 |  | 0 | 03+ | D[17](http://ref.x86asm.net/coder32-abc.html#gen_note_SETcc_0F90-0F9F) |  |  |  | ....z..c |  |  |  |  | Set Byte on Condition - below or equal/not above (CF=1 OR ZF=1) |
| SETNA | **r/m8** |  |  |  |
| SETL | **r/m8** |  |  |  |  |  | 0F | 9C |  | 0 | 03+ | D[17](http://ref.x86asm.net/coder32-abc.html#gen_note_SETcc_0F90-0F9F) |  |  |  | o..s.... |  |  |  |  | Set Byte on Condition - less/not greater (SF!=OF) |
| SETNGE | **r/m8** |  |  |  |
| SETLE | **r/m8** |  |  |  |  |  | 0F | 9E |  | 0 | 03+ | D[17](http://ref.x86asm.net/coder32-abc.html#gen_note_SETcc_0F90-0F9F) |  |  |  | o..sz... |  |  |  |  | Set Byte on Condition - less or equal/not greater ((ZF=1) OR (SF!=OF)) |
| SETNG | **r/m8** |  |  |  |
| SETNB | **r/m8** |  |  |  |  |  | 0F | 93 |  | 0 | 03+ | D[17](http://ref.x86asm.net/coder32-abc.html#gen_note_SETcc_0F90-0F9F) |  |  |  | .......c |  |  |  |  | Set Byte on Condition - not below/above or equal/not carry (CF=0) |
| SETAE | **r/m8** |  |  |  |
| SETNC | **r/m8** |  |  |  |
| SETNBE | **r/m8** |  |  |  |  |  | 0F | 97 |  | 0 | 03+ | D[17](http://ref.x86asm.net/coder32-abc.html#gen_note_SETcc_0F90-0F9F) |  |  |  | ....z..c |  |  |  |  | Set Byte on Condition - not below or equal/above (CF=0 AND ZF=0) |
| SETA | **r/m8** |  |  |  |
| SETNL | **r/m8** |  |  |  |  |  | 0F | 9D |  | 0 | 03+ | D[17](http://ref.x86asm.net/coder32-abc.html#gen_note_SETcc_0F90-0F9F) |  |  |  | o..s.... |  |  |  |  | Set Byte on Condition - not less/greater or equal (SF=OF) |
| SETGE | **r/m8** |  |  |  |
| SETNLE | **r/m8** |  |  |  |  |  | 0F | 9F |  | 0 | 03+ | D[17](http://ref.x86asm.net/coder32-abc.html#gen_note_SETcc_0F90-0F9F) |  |  |  | o..sz... |  |  |  |  | Set Byte on Condition - not less nor equal/greater ((ZF=0) AND (SF=OF)) |
| SETG | **r/m8** |  |  |  |
| SETNO | **r/m8** |  |  |  |  |  | 0F | 91 |  | 0 | 03+ | D[17](http://ref.x86asm.net/coder32-abc.html#gen_note_SETcc_0F90-0F9F) |  |  |  | o....... |  |  |  |  | Set Byte on Condition - not overflow (OF=0) |
| SETNP | **r/m8** |  |  |  |  |  | 0F | 9B |  | 0 | 03+ | D[17](http://ref.x86asm.net/coder32-abc.html#gen_note_SETcc_0F90-0F9F) |  |  |  | ......p. |  |  |  |  | Set Byte on Condition - not parity/parity odd (PF=0) |
| SETPO | **r/m8** |  |  |  |
| SETNS | **r/m8** |  |  |  |  |  | 0F | 99 |  | 0 | 03+ | D[17](http://ref.x86asm.net/coder32-abc.html#gen_note_SETcc_0F90-0F9F) |  |  |  | ...s.... |  |  |  |  | Set Byte on Condition - not sign (SF=0) |
| SETNZ | **r/m8** |  |  |  |  |  | 0F | 95 |  | 0 | 03+ | D[17](http://ref.x86asm.net/coder32-abc.html#gen_note_SETcc_0F90-0F9F) |  |  |  | ....z... |  |  |  |  | Set Byte on Condition - not zero/not equal (ZF=0) |
| SETNE | **r/m8** |  |  |  |
| SETO | **r/m8** |  |  |  |  |  | 0F | 90 |  | 0 | 03+ | D[17](http://ref.x86asm.net/coder32-abc.html#gen_note_SETcc_0F90-0F9F) |  |  |  | o....... |  |  |  |  | Set Byte on Condition - overflow (OF=1) |
| SETP | **r/m8** |  |  |  |  |  | 0F | 9A |  | 0 | 03+ | D[17](http://ref.x86asm.net/coder32-abc.html#gen_note_SETcc_0F90-0F9F) |  |  |  | ......p. |  |  |  |  | Set Byte on Condition - parity/parity even (PF=1) |
| SETPE | **r/m8** |  |  |  |
| SETS | **r/m8** |  |  |  |  |  | 0F | 98 |  | 0 | 03+ | D[17](http://ref.x86asm.net/coder32-abc.html#gen_note_SETcc_0F90-0F9F) |  |  |  | ...s.... |  |  |  |  | Set Byte on Condition - sign (SF=1) |
| SETZ | **r/m8** |  |  |  |  |  | 0F | 94 |  | 0 | 03+ | D[17](http://ref.x86asm.net/coder32-abc.html#gen_note_SETcc_0F90-0F9F) |  |  |  | ....z... |  |  |  |  | Set Byte on Condition - zero/equal (ZF=1) |
| SETE | **r/m8** |  |  |  |
| SFENCE |  |  |  |  | sse1 |  | 0F | AE |  | 7 | P3+ |  |  |  |  |  |  |  |  |  | Store Fence |
| SGDT | **m** | *GDTR* |  |  |  |  | 0F | 01 |  | 0 | 02+ |  |  |  |  |  |  |  |  |  | Store Global Descriptor Table Register |
| SHL | **r/m8** | imm8 |  |  |  |  |  | C0 |  | 4 | 01+ |  |  |  |  |  | o..szapc | o..sz.pc | o....a.c |  | Shift |
| SAL | **r/m8** | imm8 |  |  |
| SHL | **r/m16/32** | imm8 |  |  |  |  |  | C1 |  | 4 | 01+ |  |  |  |  |  | o..szapc | o..sz.pc | o....a.c |  | Shift |
| SAL | **r/m16/32** | imm8 |  |  |
| SHL | **r/m8** | 1 |  |  |  |  |  | D0 |  | 4 |  |  |  |  |  |  | o..szapc | o..sz.pc | .....a.. |  | Shift |
| SAL | **r/m8** | 1 |  |  |
| SHL | **r/m16/32** | 1 |  |  |  |  |  | D1 |  | 4 |  |  |  |  |  |  | o..szapc | o..sz.pc | .....a.. |  | Shift |
| SAL | **r/m16/32** | 1 |  |  |
| SHL | **r/m8** | CL |  |  |  |  |  | D2 |  | 4 |  |  |  |  |  |  | o..szapc | o..sz.pc | o....a.c |  | Shift |
| SAL | **r/m8** | CL |  |  |
| SHL | **r/m16/32** | CL |  |  |  |  |  | D3 |  | 4 |  |  |  |  |  |  | o..szapc | o..sz.pc | o....a.c |  | Shift |
| SAL | **r/m16/32** | CL |  |  |
| SHLD | **r/m16/32** | r16/32 | imm8 |  |  |  | 0F | A4 |  | r | 03+ |  |  |  |  |  | o..szapc | o..sz.pc | o....a.c |  | Double Precision Shift Left |
| SHLD | **r/m16/32** | r16/32 | CL |  |  |  | 0F | A5 |  | r | 03+ |  |  |  |  |  | o..szapc | o..sz.pc | o....a.c |  | Double Precision Shift Left |
| SHR | **r/m8** | imm8 |  |  |  |  |  | C0 |  | 5 | 01+ |  |  |  |  |  | o..szapc | o..sz.pc | o....a.c |  | Shift |
| SHR | **r/m16/32** | imm8 |  |  |  |  |  | C1 |  | 5 | 01+ |  |  |  |  |  | o..szapc | o..sz.pc | o....a.c |  | Shift |
| SHR | **r/m8** | 1 |  |  |  |  |  | D0 |  | 5 |  |  |  |  |  |  | o..szapc | o..sz.pc | .....a.. |  | Shift |
| SHR | **r/m16/32** | 1 |  |  |  |  |  | D1 |  | 5 |  |  |  |  |  |  | o..szapc | o..sz.pc | .....a.. |  | Shift |
| SHR | **r/m8** | CL |  |  |  |  |  | D2 |  | 5 |  |  |  |  |  |  | o..szapc | o..sz.pc | o....a.c |  | Shift |
| SHR | **r/m16/32** | CL |  |  |  |  |  | D3 |  | 5 |  |  |  |  |  |  | o..szapc | o..sz.pc | o....a.c |  | Shift |
| SHRD | **r/m16/32** | r16/32 | imm8 |  |  |  | 0F | AC |  | r | 03+ |  |  |  |  |  | o..szapc | o..sz.pc | o....a.c |  | Double Precision Shift Right |
| SHRD | **r/m16/32** | r16/32 | CL |  |  |  | 0F | AD |  | r | 03+ |  |  |  |  |  | o..szapc | o..sz.pc | o....a.c |  | Double Precision Shift Right |
| SHUFPD | **xmm** | xmm/m128 | imm8 |  | sse2 | 66 | 0F | C6 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Shuffle Packed Double-FP Values |
| SHUFPS | **xmm** | xmm/m128 | imm8 |  | sse1 |  | 0F | C6 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Shuffle Packed Single-FP Values |
| SIDT | **m** | *IDTR* |  |  |  |  | 0F | 01 |  | 1 | 02+ |  |  |  |  |  |  |  |  |  | Store Interrupt Descriptor Table Register |
| SLDT | **m16** | *LDTR* |  |  |  |  | 0F | 00 |  | 0 | 02+ |  | P |  |  |  |  |  |  |  | Store Local Descriptor Table Register |
| SLDT | **r16/32** | *LDTR* |  |  |
| SMSW | **m16** | *MSW* |  |  |  |  | 0F | 01 |  | 4 | 02+ | D[12](http://ref.x86asm.net/coder32-abc.html#gen_note_SMSW_0F01_4) |  |  |  |  |  |  |  |  | Store Machine Status Word |
| SMSW | **r16/32** | *MSW* |  |  |
| SQRTPD | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 51 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Compute Square Roots of Packed Double-FP Values |
| SQRTPS | **xmm** | xmm/m128 |  |  | sse1 |  | 0F | 51 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Compute Square Roots of Packed Single-FP Values |
| SQRTSD | **xmm** | xmm/m64 |  |  | sse2 | F2 | 0F | 51 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Compute Square Root of Scalar Double-FP Value |
| SQRTSS | **xmm** | xmm/m32 |  |  | sse1 | F3 | 0F | 51 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Compute Square Root of Scalar Single-FP Value |
| SS | *SS* |  |  |  |  | 36 |  |  |  |  |  |  |  |  |  |  |  |  |  |  | SS segment override prefix |
| STC |  |  |  |  |  |  |  | F9 |  |  |  |  |  |  |  |  | .......c | .......c |  | .......C | Set Carry Flag |
| STD |  |  |  |  |  |  |  | FD |  |  |  |  |  |  |  |  | .d...... | .d...... |  | .D...... | Set Direction Flag |
| STI |  |  |  |  |  |  |  | FB |  |  |  |  |  | f[1](http://ref.x86asm.net/coder32-abc.html#rflags_iopl) |  |  | ..i..... | ..i..... |  | ..I..... | Set Interrupt Flag |
| STMXCSR | **m32** |  |  |  | sse1 |  | 0F | AE |  | 3 | P3+ |  |  |  |  |  |  |  |  |  | Store MXCSR Register State |
| STOS | **m8** | *AL* |  |  |  |  |  | AA |  |  |  |  |  |  |  | .d...... |  |  |  |  | Store String |
| STOSB | ***m8*** | *AL* |  |  |
| STOS | **m16** | *AX* |  |  |  |  |  | AB |  |  |  |  |  |  |  | .d...... |  |  |  |  | Store String |
| STOSW | ***m16*** | *AX* |  |  |
| STOS | **m16/32** | *eAX* |  |  |  |  |  | AB |  |  | 03+ |  |  |  |  | .d...... |  |  |  |  | Store String |
| STOSD | ***m32*** | *EAX* |  |  |
| STR | **m16** | *TR* |  |  |  |  | 0F | 00 |  | 1 | 02+ |  | P |  |  |  |  |  |  |  | Store Task Register |
| STR | **r16/32** | *TR* |  |  |
| SUB | **r/m8** | r8 |  |  |  |  |  | 28 |  | r |  |  |  |  | L |  | o..szapc | o..szapc |  |  | Subtract |
| SUB | **r/m16/32** | r16/32 |  |  |  |  |  | 29 |  | r |  |  |  |  | L |  | o..szapc | o..szapc |  |  | Subtract |
| SUB | **r8** | r/m8 |  |  |  |  |  | 2A |  | r |  |  |  |  |  |  | o..szapc | o..szapc |  |  | Subtract |
| SUB | **r16/32** | r/m16/32 |  |  |  |  |  | 2B |  | r |  |  |  |  |  |  | o..szapc | o..szapc |  |  | Subtract |
| SUB | **AL** | imm8 |  |  |  |  |  | 2C |  |  |  |  |  |  |  |  | o..szapc | o..szapc |  |  | Subtract |
| SUB | **eAX** | imm16/32 |  |  |  |  |  | 2D |  |  |  |  |  |  |  |  | o..szapc | o..szapc |  |  | Subtract |
| SUB | **r/m8** | imm8 |  |  |  |  |  | 80 |  | 5 |  |  |  |  | L |  | o..szapc | o..szapc |  |  | Subtract |
| SUB | **r/m16/32** | imm16/32 |  |  |  |  |  | 81 |  | 5 |  |  |  |  | L |  | o..szapc | o..szapc |  |  | Subtract |
| SUB | **r/m8** | imm8 |  |  |  |  |  | 82 |  | 5 |  |  |  |  | L |  | o..szapc | o..szapc |  |  | Subtract |
| SUB | **r/m16/32** | imm8 |  |  |  |  |  | 83 |  | 5 |  |  |  |  | L |  | o..szapc | o..szapc |  |  | Subtract |
| SUBPD | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 5C |  | r | P4+ |  |  |  |  |  |  |  |  |  | Subtract Packed Double-FP Values |
| SUBPS | **xmm** | xmm/m128 |  |  | sse1 |  | 0F | 5C |  | r | P3+ |  |  |  |  |  |  |  |  |  | Subtract Packed Single-FP Values |
| SUBSD | **xmm** | xmm/m64 |  |  | sse2 | F2 | 0F | 5C |  | r | P4+ |  |  |  |  |  |  |  |  |  | Subtract Scalar Double-FP Values |
| SUBSS | **xmm** | xmm/m32 |  |  | sse1 | F3 | 0F | 5C |  | r | P3+ |  |  |  |  |  |  |  |  |  | Subtract Scalar Single-FP Values |
| SYSENTER | ***SS*** | ***ESP*** | *IA32\_SYS…* | ... |  |  | 0F | 34 |  |  | P2+ |  | P |  |  |  | ..i..... | ..i..... |  | ..i..... | Fast System Call |
| SYSEXIT | ***SS*** | ***eSP*** | *IA32\_SYS…* | ... |  |  | 0F | 35 |  |  | P2+ |  | P | 0 |  |  |  |  |  |  | Fast Return from Fast System Call |
| TEST | r/m8 | r8 |  |  |  |  |  | 84 |  | r |  |  |  |  |  |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical Compare |
| TEST | r/m16/32 | r16/32 |  |  |  |  |  | 85 |  | r |  |  |  |  |  |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical Compare |
| TEST | AL | imm8 |  |  |  |  |  | A8 |  |  |  |  |  |  |  |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical Compare |
| TEST | eAX | imm16/32 |  |  |  |  |  | A9 |  |  |  |  |  |  |  |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical Compare |
| TEST | r/m8 | imm8 |  |  |  |  |  | F6 |  | 0 |  |  |  |  |  |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical Compare |
| TEST | r/m8 | imm8 |  |  |  |  |  | F6 |  | 1 |  | U[11](http://ref.x86asm.net/coder32-abc.html#gen_note_TEST_F6_1_F7_1) |  |  |  |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical Compare |
| TEST | r/m16/32 | imm16/32 |  |  |  |  |  | F7 |  | 0 |  |  |  |  |  |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical Compare |
| TEST | r/m16/32 | imm16/32 |  |  |  |  |  | F7 |  | 1 |  | U[11](http://ref.x86asm.net/coder32-abc.html#gen_note_TEST_F6_1_F7_1) |  |  |  |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical Compare |
| UCOMISD | xmm | xmm/m64 |  |  | sse2 | 66 | 0F | 2E |  | r | P4+ |  |  |  |  |  | ....z.pc | ....z.pc |  |  | Unordered Compare Scalar Double-FP Values and Set EFLAGS |
| UCOMISS | xmm | xmm/m32 |  |  | sse1 |  | 0F | 2E |  | r | P3+ |  |  |  |  |  | ....z.pc | ....z.pc |  |  | Unordered Compare Scalar Single-FP Values and Set EFLAGS |
| *UD* | r | r/m |  |  |  |  | 0F | B9 |  | r | 02+ | M[19](http://ref.x86asm.net/coder32-abc.html#gen_note_sug_UD_0FB9) |  |  |  |  |  |  |  |  | Undefined Instruction |
| UD2 |  |  |  |  |  |  | 0F | 0B |  |  | 02+ |  |  |  |  |  |  |  |  |  | Undefined Instruction |
| UNPCKHPD | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 15 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Unpack and Interleave High Packed Double-FP Values |
| UNPCKHPS | **xmm** | xmm/m64 |  |  | sse1 |  | 0F | 15 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Unpack and Interleave High Packed Single-FP Values |
| UNPCKLPD | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 14 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Unpack and Interleave Low Packed Double-FP Values |
| UNPCKLPS | **xmm** | xmm/m64 |  |  | sse1 |  | 0F | 14 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Unpack and Interleave Low Packed Single-FP Values |
| VERR | r/m16 |  |  |  |  |  | 0F | 00 |  | 4 | 02+ |  | P |  |  |  | ....z... | ....z... |  |  | Verify a Segment for Reading |
| VERW | r/m16 |  |  |  |  |  | 0F | 00 |  | 5 | 02+ |  | P |  |  |  | ....z... | ....z... |  |  | Verify a Segment for Writing |
| VMCALL |  |  |  |  | vmx |  | 0F | 01 | C1 | 0 | P4++ | D[23](http://ref.x86asm.net/coder32-abc.html#gen_note_VMX_vs_SVM) | P | 0 |  |  | o..szapc | o..szapc |  |  | Call to VM Monitor |
| VMCLEAR | **m64** |  |  |  | vmx | 66 | 0F | C7 |  | 6 | P4++ | D[23](http://ref.x86asm.net/coder32-abc.html#gen_note_VMX_vs_SVM) | P | 0 |  |  | o..szapc | o..szapc |  |  | Clear Virtual-Machine Control Structure |
| VMLAUNCH |  |  |  |  | vmx |  | 0F | 01 | C2 | 0 | P4++ | D[23](http://ref.x86asm.net/coder32-abc.html#gen_note_VMX_vs_SVM) | P | 0 |  |  | o..szapc | o..szapc |  |  | Launch Virtual Machine |
| VMPTRLD | m64 |  |  |  | vmx |  | 0F | C7 |  | 6 | P4++ | D[23](http://ref.x86asm.net/coder32-abc.html#gen_note_VMX_vs_SVM) | P | 0 |  |  | o..szapc | o..szapc |  |  | Load Pointer to Virtual-Machine Control Structure |
| VMPTRST | **m64** |  |  |  | vmx |  | 0F | C7 |  | 7 | P4++ | D[23](http://ref.x86asm.net/coder32-abc.html#gen_note_VMX_vs_SVM) | P | 0 |  |  | o..szapc | o..szapc |  |  | Store Pointer to Virtual-Machine Control Structure |
| VMREAD | **r/m32** | r32 |  |  | vmx |  | 0F | 78 |  | r | P4++ | D[23](http://ref.x86asm.net/coder32-abc.html#gen_note_VMX_vs_SVM) | P | 0 |  |  | o..szapc | o..szapc |  |  | Read Field from Virtual-Machine Control Structure |
| VMRESUME |  |  |  |  | vmx |  | 0F | 01 | C3 | 0 | P4++ | D[23](http://ref.x86asm.net/coder32-abc.html#gen_note_VMX_vs_SVM) | P | 0 |  |  | o..szapc | o..szapc |  |  | Resume Virtual Machine |
| VMWRITE | r32 | r/m32 |  |  | vmx |  | 0F | 79 |  | r | P4++ | D[23](http://ref.x86asm.net/coder32-abc.html#gen_note_VMX_vs_SVM) | P | 0 |  |  | o..szapc | o..szapc |  |  | Write Field to Virtual-Machine Control Structure |
| VMXOFF |  |  |  |  | vmx |  | 0F | 01 | C4 | 0 | P4++ | D[23](http://ref.x86asm.net/coder32-abc.html#gen_note_VMX_vs_SVM) | P | 0 |  |  | o..szapc | o..szapc |  |  | Leave VMX Operation |
| VMXON | m64 |  |  |  | vmx | F3 | 0F | C7 |  | 6 | P4++ | D[23](http://ref.x86asm.net/coder32-abc.html#gen_note_VMX_vs_SVM) | P | 0 |  |  | o..szapc | o..szapc |  |  | Enter VMX Operation |
| WBINVD |  |  |  |  |  |  | 0F | 09 |  |  | 04+ |  |  | 0 |  |  |  |  |  |  | Write Back and Invalidate Cache |
| WRMSR | ***MSR*** | *ECX* | *EAX* | *EDX* |  |  | 0F | 30 |  |  | P1+ |  |  | 0 |  |  |  |  |  |  | Write to Model Specific Register |
| XADD | **r/m8** | **r8** |  |  |  |  | 0F | C0 |  | r | 04+ |  |  |  | L |  | o..szapc | o..szapc |  |  | Exchange and Add |
| XADD | **r/m16/32** | **r16/32** |  |  |  |  | 0F | C1 |  | r | 04+ |  |  |  | L |  | o..szapc | o..szapc |  |  | Exchange and Add |
| XCHG | **r8** | **r/m8** |  |  |  |  |  | 86 |  | r |  |  |  |  | L |  |  |  |  |  | Exchange Register/Memory with Register |
| XCHG | **r16/32** | **r/m16/32** |  |  |  |  |  | 87 |  | r |  |  |  |  | L |  |  |  |  |  | Exchange Register/Memory with Register |
| XCHG | **r16/32** | **eAX** |  |  |  |  |  | 90+r | |  |  |  |  |  |  |  |  |  |  |  | Exchange Register/Memory with Register |
| XGETBV | ***EDX*** | ***EAX*** | *ECX* | *XCR* |  |  | 0F | 01 | D0 | 2 | C2++ |  |  |  |  |  |  |  |  |  | Get Value of Extended Control Register |
| XLAT | ***AL*** | m8 |  |  |  |  |  | D7 |  |  |  |  |  |  |  |  |  |  |  |  | Table Look-up Translation |
| XLATB | ***AL*** | *m8* |  |  |
| XOR | **r/m8** | r8 |  |  |  |  |  | 30 |  | r |  |  |  |  | L |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical Exclusive OR |
| XOR | **r/m16/32** | r16/32 |  |  |  |  |  | 31 |  | r |  |  |  |  | L |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical Exclusive OR |
| XOR | **r8** | r/m8 |  |  |  |  |  | 32 |  | r |  |  |  |  |  |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical Exclusive OR |
| XOR | **r16/32** | r/m16/32 |  |  |  |  |  | 33 |  | r |  |  |  |  |  |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical Exclusive OR |
| XOR | **AL** | imm8 |  |  |  |  |  | 34 |  |  |  |  |  |  |  |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical Exclusive OR |
| XOR | **eAX** | imm16/32 |  |  |  |  |  | 35 |  |  |  |  |  |  |  |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical Exclusive OR |
| XOR | **r/m8** | imm8 |  |  |  |  |  | 80 |  | 6 |  |  |  |  | L |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical Exclusive OR |
| XOR | **r/m16/32** | imm16/32 |  |  |  |  |  | 81 |  | 6 |  |  |  |  | L |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical Exclusive OR |
| XOR | **r/m8** | imm8 |  |  |  |  |  | 82 |  | 6 |  |  |  |  | L |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical Exclusive OR |
| XOR | **r/m16/32** | imm8 |  |  |  |  |  | 83 |  | 6 | 03+ |  |  |  | L |  | o..szapc | o..sz.pc | .....a.. | o......c | Logical Exclusive OR |
| XORPD | **xmm** | xmm/m128 |  |  | sse2 | 66 | 0F | 57 |  | r | P4+ |  |  |  |  |  |  |  |  |  | Bitwise Logical XOR for Double-FP Values |
| XORPS | **xmm** | xmm/m128 |  |  | sse1 |  | 0F | 57 |  | r | P3+ |  |  |  |  |  |  |  |  |  | Bitwise Logical XOR for Single-FP Values |
| XRSTOR | ***ST*** | ***ST1*** | ***ST2*** | ... |  |  | 0F | AE |  | 5 | C2++ |  |  |  |  |  |  |  |  |  | Restore Processor Extended States |
| XSAVE | **m** | *EDX* | *EAX* | ... |  |  | 0F | AE |  | 4 | C2++ |  |  |  |  |  |  |  |  |  | Save Processor Extended States |
| XSETBV | ***XCR*** | *ECX* | *EDX* | *EAX* |  |  | 0F | 01 | D1 | 2 | C2++ |  |  | 0 |  |  |  |  |  |  | Set Extended Control Register |

Printing is not enabled. You can order a printed copy in the [store](http://ref.x86asm.net/store/), or get [access to benefits](http://ref.x86asm.net/#Why-to-Contribute---Benefits_desc), which include also printable HTML and PDF files.

**General notes:**

1. *90 NOP*
   1. 90 NOP is not really aliased to XCHG eAX, eAX instruction. This is important in 64-bit mode where the implicit zero-extension to RAX does not happen
2. *SAL*
   1. sandpile.org -- IA-32 architecture -- opcode groups
3. *SALC*
   1. sandpile.org -- IA-32 architecture -- one byte opcodes
   2. AMD64 Architecture Programmer's Manual Volume 3, Table One-Bytes Opcodes
4. *FSTP1*
   1. Christian Ludloff wrote: While FSTP (D9 /3, mod < 11b), FSTP8 (DF /2, mod = 11b), and FSTP9 (DF /3, mod = 11b) do signal stack underflow, FSTP1 (D9 /3, mod = 11b) does not.
5. *FNENI and FNDISI*
   1. INTEL 80287 PROGRAMMER'S REFERENCE MANUAL 1987, Processor Control Instructions: The 8087 instructions FENI and FDISI perform no function in the 80287. If these opcodes are detected in an 80286/80287 instruction stream, the 80287 will perform no specific operation and no internal states will be affected.
6. *FNSETPM*
   1. INTEL 80387 PROGRAMMER'S REFERENCE MANUAL 1987, 6.1.2 Independent of CPU Addressing Modes: Unlike the 80287, the 80387 is not sensitive to the addressing and memory management of the CPU. The 80387 operates the same regardless of whether the 80386 CPU is operating in real-address mode, in protected mode, or in virtual 8086 mode.
7. *FFREEP*
   1. INTEL 80287 PROGRAMMER'S REFERENCE MANUAL 1987, Table A-2. Machine Instruction Decoding Guide: If the 80287 encounters one of these encodings (DF /1, mod = 11b) in the instruction stream, it will execute it as follows: FFREE ST(i) and pop stack
   2. Intel Architecture Optimization Reference Manual PIII, Table C-1 Pentium II and Pentium III Processors Instruction to Decoder Specification
   3. AMD Athlon Processor x86 Code Optimization Guide, Chapter 9, Use FFREEP Macro to Pop One Register from the FPU Stack
   4. sandpile.org -- IA-32 architecture -- ESC (FP) opcodes
8. *X87 aliases*
   1. sandpile.org -- IA-32 architecture -- ESC (FP) opcodes
9. *INT1, ICEBP*
   1. sandpile.org -- IA-32 architecture -- one byte opcodes
   2. AMD64 Architecture Programmer's Manual Volume 3, Table One-Bytes Opcodes
   3. Christian Ludloff wrote: Unlike INT 1 (CDh,01h), INT1 (F1h) doesn't perform the IOPL or DPL check and it can't be redirected via the TSS32.IRB.
10. *REP prefixes*
    1. Flags aren't updated until after the last iteration to make the operation faster
11. *TEST*
    1. sandpile.org -- IA-32 architecture -- opcode groups
    2. Christian Ludloff wrote: While the latest Intel manuals still omit this de-facto standard, the recent x86-64 manuals from AMD document it.
    3. AMD64 Architecture Programmer's Manual Volume 3, Table One-Byte and Two-Byte Opcode ModRM Extensions
12. *SMSW r32/64*
    1. Some processors support reading whole CR0 register, causing a security flaw.
13. *0F0D NOP*
    1. Intel 64 and IA-32 Architecture Software Developer's Manual Volume 2B: Instruction Set Reference, N-Z, Two-byte Opcode Map
    2. AMD architecture maps 3DNow! PREFETCH instructions here
14. *Hintable NOP*
    1. See U.S. Patent 5,701,442
    2. sandpile.org -- IA-32 architecture -- opcode groups
15. *MOV from/to CRn, DRn, TRn*
    1. Christian Ludloff wrote: For the MOVs from/to CRx/DRx/TRx, mod=00b/01b/10b is aliased to 11b.
    2. AMD64 Architecture Programmer's Manual Volume 3, System Instruction Reference: This instruction is always treated as a register-to-register instruction, regardless of the encoding of the MOD field in the MODR/M byte.
16. *GETSEC Leaf Functions*
    1. Intel 64 and IA-32 Architecture Software Developer's Manual Volume 2B: Instruction Set Reference, N-Z: The GETSEC instruction supports multiple leaf functions. Leaf functions are selected by the value in EAX at the time GETSEC is executed. The following leaf functions are available: CAPABILITIES, ENTERACCS, EXITAC, SENTER, SEXIT, PARAMETERS, SMCTRL, WAKEUP. GETSEC instruction operands are specific to selected leaf function.
17. *SETcc*
    1. AMD64 Architecture Programmers Manual Volume 3: General-Purpose and System Instructions: The reg field in the ModR/M byte is unused.
18. *CMPXCHG with memory operand*
    1. Intel 64 and IA-32 Architectures Software Developer's Manual Volume 2A: Instruction Set Reference, A-M: This instruction can be used with a LOCK prefix …. To simplify the interface to the processor's bus, the destination operand receives a write cycle without regard to the result of the comparison.
    2. AMD64 Architecture Programmers Manual Volume 3: General-Purpose and System Instructions: CMPXCHG always does a read-modify-write on the memory operand.
19. *0FB9 UD*
    1. Intel 64 and IA-32 Architecture Software Developer's Manual Volume 2B: Instruction Set Reference, N-Z, Two-byte Opcode Map
    2. sandpile.org -- IA-32 architecture -- two byte opcodes
20. *CMPXCHG8B, CMPXCHG16B*
    1. Intel 64 and IA-32 Architectures Software Developer's Manual Volume 2A: Instruction Set Reference, A-M: This instruction can be used with a LOCK prefix …. To simplify the interface to the processor's bus, the destination operand receives a write cycle without regard to the result of the comparison.
    2. AMD64 Architecture Programmers Manual Volume 3: General-Purpose and System Instructions: The CMPXCHG8B and CMPXCHG16B instructions always do a read-modify-write on the memory operand.
    3. CMPXCHG16B is invalid on early steppings of AMD64 architecture.
21. *BSWAP r16*
    1. Intel 64 and IA-32 Architectures Software Developer's Manual Volume 2A: Instruction Set Reference, A-M: When the BSWAP instruction references a 16-bit register, the result is undefined.
    2. AMD64 Architecture Programmer's Manual Volume 3: General-Purpose and System Instructions: The result of applying the BSWAP instruction to a 16-bit register is undefined.
22. *MASKMOVQ*
    1. Intel 64 and IA-32 Architectures Software Developer's Manual Volume 2A: Instruction Set Reference, A-M: This instruction causes a transition from x87 FPU to MMX technology state.
23. *Intel VMX*
    1. Intel VMX is not binary-compatible with AMD SVM
24. *Intel SSE4*
    1. AMD64 architecture does not support SSE4 instructions but PTEST as part of SSE5

**Notes for the Ring Level, used in case of *f* mark:**

1. rFlags.IOPL
2. CR4.TSD[bit 2]
3. CR4.PCE[bit 8]

Create a hypertext reference to this edition's mnemonic group (append mnemonic's starting letter at the end of the following line):

http://ref.x86asm.net/coder32-abc.html#

**32-bit ModR/M Byte**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| r8(/r) | | | AL | CL | DL | BL | AH | CH | DH | BH |
| r16(/r) | | | AX | CX | DX | BX | SP | BP | SI | DI |
| r32(/r) | | | EAX | ECX | EDX | EBX | ESP | EBP | ESI | EDI |
| mm(/r) | | | MM0 | MM1 | MM2 | MM3 | MM4 | MM5 | MM6 | MM7 |
| xmm(/r) | | | XMM0 | XMM1 | XMM2 | XMM3 | XMM4 | XMM5 | XMM6 | XMM7 |
| sreg | | | ES | CS | SS | DS | FS | GS | res. | res. |
| eee | | | CR0 | invd | CR2 | CR3 | CR4 | invd | invd | invd |
| eee | | | DR0 | DR1 | DR2 | DR3 | DR4[1](http://ref.x86asm.net/coder32-abc.html#modrm_dr4_dr5) | DR5[1](http://ref.x86asm.net/coder32-abc.html#modrm_dr4_dr5) | DR6 | DR7 |
| (In decimal) /digit (Opcode) | | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| (In binary) REG = | | | 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |
| Effective Address | Mod | R/M | Value of ModR/M Byte (in Hex) | | | | | | | |
| [EAX] | 00 | 000 | 00 | 08 | 10 | 18 | 20 | 28 | 30 | 38 |
| [ECX] |  | 001 | 01 | 09 | 11 | 19 | 21 | 29 | 31 | 39 |
| [EDX] |  | 010 | 02 | 0A | 12 | 1A | 22 | 2A | 32 | 3A |
| [EBX] |  | 011 | 03 | 0B | 13 | 1B | 23 | 2B | 33 | 3B |
| [[*sib*](http://ref.x86asm.net/coder32-abc.html#sib_byte_32)] |  | 100 | 04 | 0C | 14 | 1C | 24 | 2C | 34 | 3C |
| disp32 |  | 101 | 05 | 0D | 15 | 1D | 25 | 2D | 35 | 3D |
| [ESI] |  | 110 | 06 | 0E | 16 | 1E | 26 | 2E | 36 | 3E |
| [EDI] |  | 111 | 07 | 0F | 17 | 1F | 27 | 2F | 37 | 3F |
| [EAX]+disp8 | 01 | 000 | 40 | 48 | 50 | 58 | 60 | 68 | 70 | 78 |
| [ECX]+disp8 |  | 001 | 41 | 49 | 51 | 59 | 61 | 69 | 71 | 79 |
| [EDX]+disp8 |  | 010 | 42 | 4A | 52 | 5A | 62 | 6A | 72 | 7A |
| [EBX]+disp8 |  | 011 | 43 | 4B | 53 | 5B | 63 | 6B | 73 | 7B |
| [[*sib*](http://ref.x86asm.net/coder32-abc.html#sib_byte_32)]+disp8 |  | 100 | 44 | 4C | 54 | 5C | 64 | 6C | 74 | 7C |
| [EBP]+disp8 |  | 101 | 45 | 4D | 55 | 5D | 65 | 6D | 75 | 7D |
| [ESI]+disp8 |  | 110 | 46 | 4E | 56 | 5E | 66 | 6E | 76 | 7E |
| [EDI]+disp8 |  | 111 | 47 | 4F | 57 | 5F | 67 | 6F | 77 | 7F |
| [EAX]+disp32 | 10 | 000 | 80 | 88 | 90 | 98 | A0 | A8 | B0 | B8 |
| [ECX]+disp32 |  | 001 | 81 | 89 | 91 | 99 | A1 | A9 | B1 | B9 |
| [EDX]+disp32 |  | 010 | 82 | 8A | 92 | 9A | A2 | AA | B2 | BA |
| [EBX]+disp32 |  | 011 | 83 | 8B | 93 | 9B | A3 | AB | B3 | BB |
| [[*sib*](http://ref.x86asm.net/coder32-abc.html#sib_byte_32)]+disp32 |  | 100 | 84 | 8C | 94 | 9C | A4 | AC | B4 | BC |
| [EBP]+disp32 |  | 101 | 85 | 8D | 95 | 9D | A5 | AD | B5 | BD |
| [ESI]+disp32 |  | 110 | 86 | 8E | 96 | 9E | A6 | AE | B6 | BE |
| [EDI]+disp32 |  | 111 | 87 | 8F | 97 | 9F | A7 | AF | B7 | BF |
| AL/AX/EAX/ST0/MM0/XMM0 | 11 | 000 | C0 | C8 | D0 | D8 | E0 | E8 | F0 | F8 |
| CL/CX/ECX/ST1/MM1/XMM1 |  | 001 | C1 | C9 | D1 | D9 | E1 | E9 | F1 | F9 |
| DL/DX/EDX/ST2/MM2/XMM2 |  | 010 | C2 | CA | D2 | DA | E2 | EA | F2 | FA |
| BL/BX/EBX/ST3/MM3/XMM3 |  | 011 | C3 | CB | D3 | DB | E3 | EB | F3 | FB |
| AH/SP/ESP/ST4/MM4/XMM4 |  | 100 | C4 | CC | D4 | DC | E4 | EC | F4 | FC |
| CH/BP/EBP/ST5/MM5/XMM5 |  | 101 | C5 | CD | D5 | DD | E5 | ED | F5 | FD |
| DH/SI/ESI/ST6/MM6/XMM6 |  | 110 | C6 | CE | D6 | DE | E6 | EE | F6 | FE |
| BH/DI/EDI/ST7/MM7/XMM7 |  | 111 | C7 | CF | D7 | DF | E7 | EF | F7 | FF |

**32-bit SIB Byte**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| r32 | | | EAX | ECX | EDX | EBX | ESP | →[1](http://ref.x86asm.net/coder32-abc.html#sib32_base_101) | ESI | EDI |
| (In decimal) Base = | | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| (In binary) Base = | | | 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |
| Scaled Index | SS | Index | Value of SIB Byte (in Hexadecimal) | | | | | | | |
| [EAX] | 00 | 000 | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 |
| [ECX] |  | 001 | 08 | 09 | 0A | 0B | 0C | 0D | 0E | 0F |
| [EDX] |  | 010 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 |
| [EBX] |  | 011 | 18 | 19 | 1A | 1B | 1C | 1D | 1E | 1F |
| *none* |  | 100 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 |
| [EBP] |  | 101 | 28 | 29 | 2A | 2B | 2C | 2D | 2E | 2F |
| [ESI] |  | 110 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 |
| [EDI] |  | 111 | 38 | 39 | 3A | 3B | 3C | 3D | 3E | 3F |
| [EAX\*2] | 01 | 000 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 |
| [ECX\*2] |  | 001 | 48 | 49 | 4A | 4B | 4C | 4D | 4E | 4F |
| [EDX\*2] |  | 010 | 50 | 51 | 52 | 53 | 54 | 55 | 56 | 57 |
| [EBX\*2] |  | 011 | 58 | 59 | 5A | 5B | 5C | 5D | 5E | 5F |
| *none* |  | 100 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 |
| [EBP\*2] |  | 101 | 68 | 69 | 6A | 6B | 6C | 6D | 6E | 6F |
| [ESI\*2] |  | 110 | 70 | 71 | 72 | 73 | 74 | 75 | 76 | 77 |
| [EDI\*2] |  | 111 | 78 | 79 | 7A | 7B | 7C | 7D | 7E | 7F |
| [EAX\*4] | 10 | 000 | 80 | 81 | 82 | 83 | 84 | 85 | 86 | 87 |
| [ECX\*4] |  | 001 | 88 | 89 | 8A | 8B | 8C | 8D | 8E | 8F |
| [EDX\*4] |  | 010 | 90 | 91 | 92 | 93 | 94 | 95 | 96 | 97 |
| [EBX\*4] |  | 011 | 98 | 99 | 9A | 9B | 9C | 9D | 9E | 9F |
| *none* |  | 100 | A0 | A1 | A2 | A3 | A4 | A5 | A6 | A7 |
| [EBP\*4] |  | 101 | A8 | A9 | AA | AB | AC | AD | AE | AF |
| [ESI\*4] |  | 110 | B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
| [EDI\*4] |  | 111 | B8 | B9 | BA | BB | BC | BD | BE | BF |
| [EAX\*8] | 11 | 000 | C0 | C1 | C2 | C3 | C4 | C5 | C6 | C7 |
| [ECX\*8] |  | 001 | C8 | C9 | CA | CB | CC | CD | CE | CF |
| [EDX\*8] |  | 010 | D0 | D1 | D2 | D3 | D4 | D5 | D6 | D7 |
| [EBX\*8] |  | 011 | D8 | D9 | DA | DB | DC | DD | DE | DF |
| *none* |  | 100 | E0 | E1 | E2 | E3 | E4 | E5 | E6 | E7 |
| [EBP\*8] |  | 101 | E8 | E9 | EA | EB | EC | ED | EE | EF |
| [ESI\*8] |  | 110 | F0 | F1 | F2 | F3 | F4 | F5 | F6 | F7 |
| [EDI\*8] |  | 111 | F8 | F9 | FA | FB | FC | FD | FE | FF |

|  |  |
| --- | --- |
| **SIB Note 1** | |
| Mod bits | base |
| 00 | disp32 |
| 01 | EBP+disp8 |
| 10 | EBP+disp32 |

**16-bit ModR/M Byte**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| r8(/r) | | | AL | CL | DL | BL | AH | CH | DH | BH |
| r16(/r) | | | AX | CX | DX | BX | SP | BP | SI | DI |
| r32(/r) | | | EAX | ECX | EDX | EBX | ESP | EBP | ESI | EDI |
| mm(/r) | | | MM0 | MM1 | MM2 | MM3 | MM4 | MM5 | MM6 | MM7 |
| xmm(/r) | | | XMM0 | XMM1 | XMM2 | XMM3 | XMM4 | XMM5 | XMM6 | XMM7 |
| sreg | | | ES | CS | SS | DS | FS | GS | res. | res. |
| eee | | | CR0 | invd | CR2 | CR3 | CR4 | invd | invd | invd |
| eee | | | DR0 | DR1 | DR2 | DR3 | DR4[1](http://ref.x86asm.net/coder32-abc.html#modrm_dr4_dr5) | DR5[1](http://ref.x86asm.net/coder32-abc.html#modrm_dr4_dr5) | DR6 | DR7 |
| (In decimal) /digit (Opcode) | | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| (In binary) REG = | | | 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |
| Effective Address | Mod | R/M | Value of ModR/M Byte (in Hex) | | | | | | | |
| [BX+SI] | 00 | 000 | 00 | 08 | 10 | 18 | 20 | 28 | 30 | 38 |
| [BX+DI] |  | 001 | 01 | 09 | 11 | 19 | 21 | 29 | 31 | 39 |
| [BP+SI] |  | 010 | 02 | 0A | 12 | 1A | 22 | 2A | 32 | 3A |
| [BP+DI] |  | 011 | 03 | 0B | 13 | 1B | 23 | 2B | 33 | 3B |
| [SI] |  | 100 | 04 | 0C | 14 | 1C | 24 | 2C | 34 | 3C |
| [DI] |  | 101 | 05 | 0D | 15 | 1D | 25 | 2D | 35 | 3D |
| disp16 |  | 110 | 06 | 0E | 16 | 1E | 26 | 2E | 36 | 3E |
| [BX] |  | 111 | 07 | 0F | 17 | 1F | 27 | 2F | 37 | 3F |
| [BX+SI]+disp8 | 01 | 000 | 40 | 48 | 50 | 58 | 60 | 68 | 70 | 78 |
| [BX+DI]+disp8 |  | 001 | 41 | 49 | 51 | 59 | 61 | 69 | 71 | 79 |
| [BP+SI]+disp8 |  | 010 | 42 | 4A | 52 | 5A | 62 | 6A | 72 | 7A |
| [BP+DI]+disp8 |  | 011 | 43 | 4B | 53 | 5B | 63 | 6B | 73 | 7B |
| [SI]+disp8 |  | 100 | 44 | 4C | 54 | 5C | 64 | 6C | 74 | 7C |
| [DI]+disp8 |  | 101 | 45 | 4D | 55 | 5D | 65 | 6D | 75 | 7D |
| [BP]+disp8 |  | 110 | 46 | 4E | 56 | 5E | 66 | 6E | 76 | 7E |
| [BX]+disp8 |  | 111 | 47 | 4F | 57 | 5F | 67 | 6F | 77 | 7F |
| [BX+SI]+disp16 | 10 | 000 | 80 | 88 | 90 | 98 | A0 | A8 | B0 | B8 |
| [BX+DI]+disp16 |  | 001 | 81 | 89 | 91 | 99 | A1 | A9 | B1 | B9 |
| [BP+SI]+disp16 |  | 010 | 82 | 8A | 92 | 9A | A2 | AA | B2 | BA |
| [BP+DI]+disp16 |  | 011 | 83 | 8B | 93 | 9B | A3 | AB | B3 | BB |
| [SI]+disp16 |  | 100 | 84 | 8C | 94 | 9C | A4 | AC | B4 | BC |
| [DI]+disp16 |  | 101 | 85 | 8D | 95 | 9D | A5 | AD | B5 | BD |
| [BP]+disp16 |  | 110 | 86 | 8E | 96 | 9E | A6 | AE | B6 | BE |
| [BX]+disp16 |  | 111 | 87 | 8F | 97 | 9F | A7 | AF | B7 | BF |
| AL/AX/EAX/ST0/MM0/XMM0 | 11 | 000 | C0 | C8 | D0 | D8 | E0 | E8 | F0 | F8 |
| CL/CX/ECX/ST1/MM1/XMM1 |  | 001 | C1 | C9 | D1 | D9 | E1 | E9 | F1 | F9 |
| DL/DX/EDX/ST2/MM2/XMM2 |  | 010 | C2 | CA | D2 | DA | E2 | EA | F2 | FA |
| BL/BX/EBX/ST3/MM3/XMM3 |  | 011 | C3 | CB | D3 | DB | E3 | EB | F3 | FB |
| AH/SP/ESP/ST4/MM4/XMM4 |  | 100 | C4 | CC | D4 | DC | E4 | EC | F4 | FC |
| CH/BP/EBP/ST5/MM5/XMM5 |  | 101 | C5 | CD | D5 | DD | E5 | ED | F5 | FD |
| DH/SI/ESI/ST6/MM6/XMM6 |  | 110 | C6 | CE | D6 | DE | E6 | EE | F6 | FE |
| BH/DI/EDI/ST7/MM7/XMM7 |  | 111 | C7 | CF | D7 | DF | E7 | EF | F7 | FF |

**ModR/M Note 1: Debug Registers DR4 and DR5**

References to debug registers DR4 and DR5 cause an undefined opcode (#UD) exception to be generated when CR4.DE[bit 3] (Debugging Extensions) set; when clear, processor aliases references to registers DR4 and DR5 to DR6 and DR7 for compatibility with software written to run on earlier IA-32 processors.