Intel® Extended Memory 64 Technology Software Developer's Guide Volume 2 of 2

Revision 1.1

NOTE: This guide consists of volumes 1 and 2. Refer to both volumes when evaluating your design needs.

Part Number: 300835-002

CHAPTER 3
INSTRUCTION SET REFERENCE (M-Z)

Chapter 3 continues the alphabetical discussion of IA-32 instructions (M-Z) started in Chapter 2. To access information on the remainder of the IA-32 instructions (A-M), see *Intel® Extended Memory 64 Technology Software Developer's Guide, Volume 1*.

MASKMOVDQU—Store Selected Bytes of Double Quadword

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
66 0F F7 /r	MASKMOVDQU *xmm1*, *xmm2*	Valid	Valid	Selectively write bytes from *xmm1* to memory location using the byte mask in *xmm2*. The default memory location is specified by DS:EDI

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments. (even if mask is all 0s).
	If the destination operand is in a nonwritable segment.
	If the DS, ES, FS, or GS register contains a null segment selector.
#SS(0)	For an illegal address in the SS segment (even if mask is all 0s).
#PF(fault-code)	For a page fault (implementation specific).
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Real-Address Mode Exceptions

#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH. (even if mask is all 0s).
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault (implementation specific).

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#GP(0)	If the memory address is in a non-canonical form.
#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#PF(fault-code)	For a page fault (implementation specific).
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

MASKMOVQ—Store Selected Bytes of Quadword

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F F7 /r	MASKMOVQ *mm1*, *mm2*	Valid	Valid	Selectively write bytes from *mm1* to memory location using the byte mask in *mm2*. The default memory location is specified by DS:EDI

IA-32e Mode Operation

In 64-bit mode, memory address is specified by RDI.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments. (even if mask is all 0s).
	If the destination operand is in a nonwritable segment.
	If the DS, ES, FS, or GS register contains a null segment selector.
#SS(0)	For an illegal address in the SS segment (even if mask is all 0s).
#PF(fault-code)	For a page fault (implementation specific).
#NM	If TS in CR0 is set.
#MF	If there is a pending FPU exception.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.
	If Mod field of the ModR/M byte not 11B
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH. (even if mask is all 0s).
#NM	If TS in CR0 is set.
#MF	If there is a pending FPU exception.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault (implementation specific).
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#GP(0)	If the memory address is in a non-canonical form.
#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#PF(fault-code)	For a page fault (implementation specific).
#NM	If TS in CR0 is set.
#MF	If there is a pending FPU exception.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.
	If Mod field of the ModR/M byte not 11B
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

MAXPD—Return Maximum Packed Double-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
66 0F 5F /r	MAXPD *xmm1*, *xmm2/m128*	Valid	Valid	Return the maximum double-precision floating-point values between *xmm2/m128* and *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

Invalid (including QNaN source operand), Denormal.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

MAXPS—Return Maximum Packed Single-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 5F /r	MAXPS *xmm1, xmm2/m128*	Valid	Valid	Return the maximum single-precision floating-point values between *xmm2/m128* and *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

Invalid (including QNaN source operand), Denormal.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

MAXSD—Return Maximum Scalar Double-Precision Floating-Point Value

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
F2 0F 5F /r	MAXSD *xmm1, xmm2/m64*	Valid	Valid	Return the maximum scalar double-precision floating-point value between *xmm2/mem64* and *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

Invalid (including QNaN source operand), Denormal.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

MAXSS—Return Maximum Scalar Single-Precision Floating-Point Value

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
F3 0F 5F /r	MAXSS *xmm1, xmm2/m32*	Valid	Valid	Return the maximum scalar single-precision floating-point value between *xmm2/mem32* and *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

Invalid (including QNaN source operand), Denormal.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

MFENCE—Memory Fence

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F AE /6	MFENCE	Valid	Valid	Serializes load and store operations.

Intel C/C++ Compiler Intrinsic Equivalent

void_mm_mfence(void)

IA-32e Mode Operation

Same as legacy.

Exceptions (All Modes of Operation)

None.

MINPD—Return Minimum Packed Double-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
66 0F 5D /r	MINPD *xmm1*, *xmm2/ m128*	Valid	Valid	Return the minimum double-precision floating-point values between *xmm2/m128* and *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

Invalid (including QNaN source operand), Denormal.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

MINPS—Return Minimum Packed Single-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 5D /r	MINPS *xmm1, xmm2/m128*	Valid	Valid	Return the minimum single-precision floating-point values between *xmm2/m128* and *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

Invalid (including QNaN source operand), Denormal.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

MINSD—Return Minimum Scalar Double-Precision Floating-Point Value

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
F2 0F 5D /r	MINSD *xmm1, xmm2/m64*	Valid	Valid	Return the minimum scalar double-precision floating-point value between *xmm2/mem64* and *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

Invalid (including QNaN source operand), Denormal.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

MINSS—Return Minimum Scalar Single-Precision Floating-Point Value

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
F3 0F 5D /r	MINSS *xmm1*, *xmm2/m32*	Valid	Valid	Return the minimum scalar single-precision floating-point value between *xmm2/mem32* and *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

Invalid (including QNaN source operand), Denormal.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

MONITOR—Setup Monitor Address

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
OF 01 *C8*	MONITOR EAX, *ECX, EDX*	Valid	Valid	Sets up a linear address range to be monitored by hardware and activates the monitor. The address range should be a write-back memory caching type. The default address is DS:EAX

Flags Affected

None.

IA-32e Mode Operation

Same as protected mode.

Protected Mode Exceptions

#GP(0)	If the current privilege level is not 0.
	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If ECX != 0.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	For a page fault.
#UD	If CPUID.MONITOR (ECX bit 3) = 0.
	If the F3H, F2H, 66H or LOCK prefix is used.

Real-Address Mode Exceptions

#GP(0)	If ECX != 0.
#UD	If CPUID.MONITOR (ECX bit 3) = 0.
	If the F3H, F2H, 66H or LOCK prefix is used.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#GP(0)	If the current privilege level is not 0.
	If the memory address is in a non-canonical form.
	If ECX != 0.
#PF(fault-code)	For a page fault.
#UD	If CPUID.MONITOR (ECX bit 3) = 0.
	If the F3H, F2H, 66H or LOCK prefix is used.

MOV—Move

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
88 /*r*	MOV *r/m8,r8*	Valid	Valid	Move *r8* to *r/m8*
REX + 88 /*r*	MOV *r/m8***,r8****	Valid	N.E.	Move *r8* to *r/m8*
89 /*r*	MOV *r/m16,r16*	Valid	Valid	Move *r16* to *r/m16*
89 /*r*	MOV *r/m32,r32*	Valid	Valid	Move *r32* to *r/m32*
REX.W + 89 /*r*	MOV *r/m64,r64*	Valid	N.E.	Move *r64* to *r/m64*
8A /*r*	MOV *r8,r/m8*	Valid	Valid	Move *r/m8* to *r8*
REX + 8A /*r*	MOV *r8***,r/m8****	Valid	N.E.	Move *r/m8* to *r8*
8B /*r*	MOV *r16,r/m16*	Valid	Valid	Move *r/m16* to *r16*
8B /*r*	MOV *r32,r/m32*	Valid	Valid	Move *r/m32* to *r32*
REX.W + 8B /*r*	MOV *r64,r/m64*	Valid	N.E.	Move *r/m64* to *r64*
8C /*r*	MOV *r/m16,Sreg***	Valid	Valid	Move segment register to *r/m16*
REX.W + 8C /*r*	MOV *r/m64,Sreg***	Valid	Valid	Move zero extended 16-bit segment register to *r/m64*
8E /*r*	MOV *Sreg,r/m16***	Valid	Valid	Move *r/m16* to segment register
REX.W + 8E /*r*	MOV *Sreg,r/m64***	Valid	Valid	Move *lower 16 bits of r/m64* to segment register
A0	MOV AL,*moffs8**	Valid	Valid	Move byte at (*seg:offset*) to AL
REX.W + A0	MOV AL,*moffs8**	Valid	N.E.	Move byte at (*offset*) to AL
A1	MOV AX,*moffs16**	Valid	Valid	Move word at (*seg:offset*) to AX
A1	MOV EAX,*moffs32**	Valid	Valid	Move doubleword at (*seg:offset*) to EAX
REX.W + A1	MOV RAX,*moffs64**	Valid	N.E.	Move quadword at (*offset*) to RAX
A2	MOV *moffs8*,AL	Valid	Valid	Move AL to (*seg:offset*)
REX.W + A2	MOV *moffs8****,AL	Valid	N.E.	Move AL to (*offset*)
A3	MOV *moffs16**,AX	Valid	Valid	Move AX to (*seg:offset*)
A3	MOV *moffs32**,EAX	Valid	Valid	Move EAX to (*seg:offset*)
REX.W + A3	MOV *moffs64**,RAX	Valid	N.E.	Move RAX to (*offset*)
B0+ *rb*	MOV *r8,imm8*	Valid	Valid	Move *imm8* to *r8*
REX + B0+ *rb*	MOV *r8***,imm8*	Valid	N.E.	Move *imm8* to *r8*
B8+ *rw*	MOV *r16,imm16*	Valid	Valid	Move *imm16* to *r16*
B8+ *rd*	MOV *r32,imm32*	Valid	Valid	Move *imm32* to *r32*
REX.W + B8+ *rd*	MOV *r64,imm64*	Valid	N.E.	Move *imm64* to *r64*
C6 /*0*	MOV *r/m8,imm8*	Valid	Valid	Move *imm8* to *r/m8*
REX + C6 /*0*	MOV *r/m8***,imm8*	Valid	N.E.	Move *imm8* to *r/m8*
C7 /*0*	MOV *r/m16,imm16*	Valid	Valid	Move *imm16* to *r/m16*
C7 /*0*	MOV *r/m32,imm32*	Valid	Valid	Move *imm32* to *r/m32*
REX.W + C7 /*0*	MOV *r/m64,imm32*	Valid	N.E.	Move *imm32 zero extended to 64-bits* to *r/m64*

* The *moffs8*, *moffs16*, *moffs32* and *moffs64* operands specify a simple offset relative to the segment base, where 8, 16, 32 and 64 refer to the size of the data. The address-size attribute of the instruction determines the size of the offset, either 16, 32 or 64 bits.

** In 32-bit mode, the assembler may insert the 16-bit operand-size prefix with this instruction (see the following “Description” section for further information).

*** In 64-bit mode, r/m8 can not be encoded to access the following byte registers if an REX prefix is used: AH, BH, CH, DH. Also refer to Section 1.4.2.2.

Flags Affected

None.

IA-32e Mode Operation

Promoted to 64-bits.

Default operand size 32-bits.

Enables access to new registers R8-R15.

Protected Mode Exceptions

#GP(0)	If attempt is made to load SS register with null segment selector.
	If the destination operand is in a nonwritable segment.
	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the DS, ES, FS, or GS register contains a null segment selector.
#GP(selector)	If segment selector index is outside descriptor table limits.
	If the SS register is being loaded and the segment selector's RPL and the segment descriptor's DPL are not equal to the CPL.
	If the SS register is being loaded and the segment pointed to is a nonwritable data segment.
	If the DS, ES, FS, or GS register is being loaded and the segment pointed to is not a data or readable code segment.
	If the DS, ES, FS, or GS register is being loaded and the segment pointed to is a data or nonconforming code segment, but both the RPL and the CPL are greater than the DPL.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#SS(selector)	If the SS register is being loaded and the segment pointed to is marked not present.
#NP	If the DS, ES, FS, or GS register is being loaded and the segment pointed to is marked not present.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.
#UD	If attempt is made to load the CS register.

Real-Address Mode Exceptions

#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS	If a memory operand effective address is outside the SS segment limit.
#UD	If attempt is made to load the CS register.

Virtual-8086 Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.
#UD	If attempt is made to load the CS register.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#GP(0)	If the memory address is in a non-canonical form.
	If an attempt is made to load SS register with null segment selector when CPL = 3.
	If an attempt is made to load SS register with null segment selector when CPL < 3 and CPL != RPL.
#GP(selector)	If segment selector index is outside descriptor table limits.
	If the memory access to the descriptor table is non-canonical.
	If the SS register is being loaded and the segment selector's RPL and the segment descriptor's DPL are not equal to the CPL.
	If the SS register is being loaded and the segment pointed to is a nonwritable data segment.
	If the DS, ES, FS, or GS register is being loaded and the segment pointed to is not a data or readable code segment.
	If the DS, ES, FS, or GS register is being loaded and the segment pointed to is a data or nonconforming code segment, but both the RPL and the CPL are greater than the DPL.
#SS(U)	If the stack address is in a non-canonical form.
#SS(selector)	If the SS register is being loaded and the segment pointed to is marked not present.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.
#UD	If attempt is made to load the CS register.

MOV—Move to/from Control Registers

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 22 /*r*	MOV CR0,*r32*	Valid	Valid	Move *r32* to CR0
REX.W + 0F 22 /*r*	MOV CR0,*r64*	Valid	N.E.	Move *r64* to extended CR0.
0F 22 /*r*	MOV CR2,*r32*	Valid	Valid	Move *r32* to CR2
REX.W + 0F 22 /*r*	MOV CR2,*r64*	Valid	N.E.	Move *r64* to extended CR2.
0F 22 /*r*	MOV CR3,*r32*	Valid	Valid	Move *r32* to CR3
REX.W + 0F 22 /*r*	MOV CR3,*r64*	Valid	N.E.	Move *r64* to extended CR3.
0F 22 /*r*	MOV CR4,*r32*	Valid	Valid	Move *r32* to CR4
REX.W + 0F 22 /*r*	MOV CR4,*r64*	Valid	N.E.	Move *r64* to extended CR4.
0F 20 /*r*	MOV *r32,*CR0	Valid	Valid	Move CR0 to *r32*
REX.W + 0F 20 /*r*	MOV *r64,*CR0	Valid	N.E.	Move extended CR0 to *r64*.
0F 20 /*r*	MOV *r32,*CR2	Valid	Valid	Move CR2 to *r32*
REX.W + 0F 20 /*r*	MOV *r64,*CR2	Valid	N.E.	Move extended CR2 to *r64*.
0F 20 /*r*	MOV *r32,*CR3	Valid	Valid	Move CR3 to *r32*
REX.W + 0F 20 /*r*	MOV *r64,*CR3	Valid	N.E.	Move extended CR3 to *r64*.
0F 20 /*r*	MOV *r32,*CR4	Valid	Valid	Move CR4 to *r32*
REX.W + 0F 20 /*r*	MOV *r64,*CR4	Valid	N.E.	Move extended CR4 to *r64*.
0F 20 /*r*	MOV *r32,*CR8	Valid	N.E.	Move CR8 to *r32*
REX.W + 0F 20 /*r*	MOV *r64,*CR8	Valid	N.E.	Move extended CR8 to *r64*.

Flags Affected

The OF, SF, ZF, AF, PF, and CF flags are undefined.

IA-32e Mode Operation

Promoted to 64-bits.

Operand size fixed at 64-bits (see Control registers section).

Enables access to new registers R8-R15.

Protected Mode Exceptions

#GP(0) If the current privilege level is not 0.

If an attempt is made to write invalid bit combinations in CR0 (such as setting the PG flag to 1 when the PE flag is set to 0, or setting the CD flag to 0 when the NW flag is set to 1).

If an attempt is made to write a 1 to any reserved bit in CR4.

Attempting to activate IA-32e mode (MOV CR0) with a 286 TSS in TR.

Attempting to activate IA-32e mode (MOV CR0) with CR4.PAE not set.

Attempting to activate IA-32e mode (MOV CR0) with a CS that has the L-bit set.

Real-Address Mode Exceptions

#GP If an attempt is made to write a 1 to any reserved bit in CR4.

If an attempt is made to write invalid bit combinations in CR0 (such as setting the PG flag to 1 when the PE flag is set to 0, or setting the CD flag to 0 when the NW flag is set to 1).

Virtual-8086 Mode Exceptions

#GP(0) These instructions cannot be executed in virtual-8086 mode.

Compatibility Mode Exceptions

#GP(0)	If the current privilege level is not 0.
	If an attempt is made to write invalid bit combinations in CR0 (such as setting the PG flag to 1 when the PE flag is set to 0, or setting the CD flag to 0 when the NW flag is set to 1).
	If an attempt is made to write a 1 to any reserved bit in CR3.
	If an attempt is made to leave IA-32e mode by clearing CR4.PAE.

64-Bit Mode Exceptions

#GP(0)	If the current privilege level is not 0.
	If an attempt is made to write invalid bit combinations in CR0 (such as setting the PG flag to 1 when the PE flag is set to 0, or setting the CD flag to 0 when the NW flag is set to 1).
	Attempting to clear CR0.PG.
	If an attempt is made to write a 1 to any reserved bit in CR4.
	If an attempt is made to write a 1 to any reserved bit in CR8.
	If an attempt is made to write a 1 to any reserved bit in CR3.
	If an attempt is made to leave IA-32e mode by clearing CR4.PAE.

MOV—Move to/from Debug Registers

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 21/*r*	MOV *r32,* DR0-DR7	Valid	Valid	Move debug register to *r32*
REX.W + 0F 21/*r*	MOV *r64,* DR0-DR7	Valid	N.E.	Move extended debug register to *r64*.
0F 23 /*r*	MOV DR0-DR7,*r32*	Valid	Valid	Move *r32* to debug register
REX.W + 0F 23 /*r*	MOV DR0-DR7,*r64*	Valid	N.E.	Move *r64* to extended debug register.

Flags Affected

The OF, SF, ZF, AF, PF, and CF flags are undefined.

IA-32e Mode Operation

Promoted to 64-bits.

Operand size fixed at 64-bits (see Debug registers section).

Enables access to new registers R8-R15.

Protected Mode Exceptions

#GP(0)	If the current privilege level is not 0.
#UD	If the DE (debug extensions) bit of CR4 is set and a MOV instruction is executed involving DR4 or DR5.
#DB	If any debug register is accessed while the GD flag in debug register DR7 is set.

Real-Address Mode Exceptions

#UD	If the DE (debug extensions) bit of CR4 is set and a MOV instruction is executed involving DR4 or DR5.
#DB	If any debug register is accessed while the GD flag in debug register DR7 is set.

Virtual-8086 Mode Exceptions

#GP(0)	The debug registers cannot be loaded or read when in virtual-8086 mode.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#GP(0)	If the current privilege level is not 0.
#UD	If the DE (debug extensions) bit of CR4 is set and a MOV instruction is executed involving DR4 or DR5.
#DB	If any debug register is accessed while the GD flag in debug register DR7 is set.

MOVAPD—Move Aligned Packed Double-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
66 0F 28 /r	MOVAPD *xmm1*, xmm2/m128	Valid	Valid	Move packed double-precision floating-point values from *xmm2/m128* to *xmm1*.
66 0F 29 /r	MOVAPD *xmm2/m128*, *xmm1*	Valid	Valid	Move packed double-precision floating-point values from *xmm1* to *xmm2/m128*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If the destination operand is in a nonwritable segment.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

MOVAPS—Move Aligned Packed Single-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 28 /r	MOVAPS *xmm1*, xmm2/m128	Valid	Valid	Move packed single-precision floating-point values from *xmm2/m128* to *xmm1*.
0F 29 /r	MOVAPS *xmm2/m128*, *xmm1*	Valid	Valid	Move packed single-precision floating-point values from *xmm1* to *xmm2/m128*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If the destination operand is in a nonwritable segment.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

MOVD/MOVQ—Move Doubleword

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 6E /r	MOVD *mm, r/m32*	Valid	Valid	Move doubleword from *r/m32* to *mm*.
REX.W + 0F 6E /r	MOVQ *mm, r/m64*	Valid	N.E.	Move quadword from *r/m64* to *mm*.
0F 7E /r	MOVD *r/m32, mm*	Valid	Valid	Move doubleword from *mm* to *r/m32*.
REX.W + 0F 7E /r	MOVQ *r/m64, mm*	Valid	N.E.	Move quadword from *mm* to *r/m64*.
66 0F 6E /r	MOVD xmm, r/m32	Valid	Valid	Move doubleword from *r/m32* to *xmm*.
REX.W + 66 0F 6E /r	MOVQ xmm, r/m64	Valid	N.E.	Move quadword from *r/m64* to *xmm*.
66 0F 7E /r	MOVD r/m32, xmm	Valid	Valid	Move doubleword from *xmm* register to *r/m32*.
REX.W + 66 0F 7E /r	MOVQ r/m64, xmm	Valid	N.E.	Move quadword from *xmm* register to *r/m64*.

Flags Affected

None.

IA-32e Mode Operation

Promoted to 64-bits.

Default operand size 32-bits.

Enables access to new registers R8-R15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	If the destination operand is in a nonwritable segment.
	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(XMM register operations only.) If OSFXSR in CR4 is 0.
	(XMM register operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(MMX register operations only.) If there is a pending FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(XMM register operations only.) If OSFXSR in CR4 is 0.
	(XMM register operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(MMX register operations only.) If there is a pending FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#UD	If EM in CR0 is set.
	(XMM register operations only.) If OSFXSR in CR4 is 0.
	(XMM register operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(MMX register operations only.) If there is a pending FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

MOVDDUP—Move One Double-Precision Floating-Point Value and Duplicate

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
F2 0F 12 /r	MOVDDUP *xmm1*, xmm2/m64	Valid	Valid	Move one double-precision floating-point value from the lower 64-bit operand in *xmm2/m64* to *xmm1* and duplicate.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE3 is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE3 is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE3 is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

MOVDQA—Move Aligned Double Quadword

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
66 0F 6F /r	MOVDQA *xmm1, xmm2/m128*	Valid	Valid	Move aligned double quadword from *xmm2/m128* to *xmm1*.
66 0F 7F /r	MOVDQA *xmm2/m128*, *xmm1*	Valid	Valid	Move aligned double quadword from *xmm1* to *xmm2/m128*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the destination operand is in a nonwritable segment.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#PF(fault-code)	If a page fault occurs.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

MOVDQU—Move Unaligned Double Quadword

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
F3 0F 6F /r	MOVDQU *xmm1*, *xmm2/m128*	Valid	Valid	Move unaligned double quadword from *xmm2/m128* to *xmm1*.
F3 0F 7F /r	MOVDQU *xmm2/m128*, *xmm1*	Valid	Valid	Move unaligned double quadword from *xmm1* to *xmm2/m128*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the destination operand is in a nonwritable segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#PF(fault-code)	If a page fault occurs.

Real-Address Mode Exceptions

#GP(0)	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

MOVDQ2Q—Move Quadword from XMM to MMX Register

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
F2 0F D6	MOVDQ2Q *mm*, *xmm*	Valid	Valid	Move low quadword from *xmm* to *mmx* register.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#MF	If there is a pending x87 FPU exception.

Real-Address Mode Exceptions

Same exceptions as in Protected Mode

Virtual-8086 Mode Exceptions

Same exceptions as in Protected Mode

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

Same as for protected mode exceptions.

MOVHLPS— Move Packed Single-Precision Floating-Point Values High to Low

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
OF 12 /r	MOVHLPS *xmm1, xmm2*	Valid	Valid	Move two packed single-precision floating-point values from high quadword of *xmm2* to low quadword of *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Real Address Mode Exceptions

Same exceptions as in Protected Mode.

Virtual 8086 Mode Exceptions

Same exceptions as in Protected Mode.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

Same as for protected mode exceptions.

MOVHPD—Move High Packed Double-Precision Floating-Point Value

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
66 0F 16 /r	MOVHPD *xmm*, *m64*	Valid	Valid	Move double-precision floating-point value from *m64* to high quadword of *xmm*.
66 0F 17 /r	MOVHPD *m64*, *xmm*	Valid	Valid	Move double-precision floating-point value from high quadword of *xmm* to *m64*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If the destination operand is in a nonwritable segment.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Same exceptions as in Protected Mode.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

MOVHPS—Move High Packed Single-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 16 /r	MOVHPS *xmm*, *m64*	Valid	Valid	Move two packed single-precision floating-point values from *m64* to high quadword of *xmm*.
0F 17 /r	MOVHPS *m64*, *xmm*	Valid	Valid	Move two packed single-precision floating-point values from high quadword of *xmm* to *m64*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If the destination operand is in a nonwritable segment.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

MOVLHPS—Move Packed Single-Precision Floating-Point Values Low to High

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
OF 16 /r	MOVLHPS *xmm1, xmm2*	Valid	Valid	Move two packed single-precision floating-point values from low quadword of *xmm2* to high quadword of *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Real Address Mode Exceptions

Same exceptions as in Protected Mode.

Virtual 8086 Mode Exceptions

Same exceptions as in Protected Mode.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

Same as for protected mode exceptions.

MOVLPD—Move Low Packed Double-Precision Floating-Point Value

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
66 0F 12 /r	MOVLPD *xmm*, *m64*	Valid	Valid	Move double-precision floating-point value from *m64* to low quadword of *xmm* register.
66 0F 13 /r	MOVLPD *m64*, *xmm*	Valid	Valid	Move double-precision floating-point nvalue from low quadword of *xmm* register to *m64*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If the destination operand is in a nonwritable segment.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

MOVLPS—Move Low Packed Single-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 12 /r	MOVLPS *xmm*, *m64*	Valid	Valid	Move two packed single-precision floating-point values from *m64* to low quadword of *xmm*.
0F 13 /r	MOVLPS *m64*, *xmm*	Valid	Valid	Move two packed single-precision floating-point values from low quadword of *xmm* to *m64*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If the destination operand is in a nonwritable segment.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

MOVMSKPD—Extract Packed Double-Precision Floating-Point Sign Mask

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
66 0F 50 /r	MOVMSKPD *r32*, *xmm*	Valid	Valid	Extract 2-bit sign mask of from *xmm* and store in *r32*.
66 + REX.W 0F 50 /r	MOVMSKPD *r64*, *xmm*	Valid	N.E.	Extract 2-bit sign mask of from *xmm* and store in *r64*. Zero extend 32-bit results to 64-bits.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Real-Address Mode Exceptions

Same exceptions as in Protected Mode.

Virtual-8086 Mode Exceptions

Same exceptions as in Protected Mode.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

Same as for protected mode exceptions.

MOVMSKPS—Extract Packed Single-Precision Floating-Point Sign Mask

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 50 /r	MOVMSKPS *r32*, *xmm*	Valid	Valid	Extract 4-bit sign mask of from *xmm* and store in *r32*.
REX.W + 0F 50 /r	MOVMSKPS *r64*, *xmm*	Valid	N.E.	Extract 4-bit sign mask of from *xmm* and store in *r64*. Zero extend 32-bit results to 64-bits.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Real-Address Mode Exceptions

Same exceptions as in Protected Mode.

Virtual 8086 Mode Exceptions

Same exceptions as in Protected Mode.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

Same as for protected mode exceptions.

MOVNTDQ—Store Double Quadword Using Non-Temporal Hint

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
66 0F E7 /r	MOVNTDQ *m128*, xmm	Valid	Valid	Move double quadword from *xmm* to *m128* using non-temporal hint.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If the destination operand is in a nonwritable segment.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

MOVNTI—Store Doubleword/Quadword Using Non-Temporal Hint

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F C3 /r	MOVNTI *m32*, *r32*	Valid	Valid	Move doubleword from *r32* to *m32* using non-temporal hint.
REX.W + 0F C3 /r	MOVNTI *m64, r64*	Valid	N.E.	Move quadword from *r64* to *m64* using non-temporal hint.

IIA-32e Mode Operation

Promoted to 64-bits. Default operand size 32-bits. Enables access to new registers R8-R15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If the destination operand is in a nonwritable segment.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#UD	If CPUID feature flag SSE2 is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#UD	If CPUID feature flag SSE2 is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	For a page fault.
#UD	If CPUID feature flag SSE2 is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

MOVNTPD—Store Packed Double-Precision Floating-Point Values Using Non-Temporal Hint

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
66 0F 2B /r	MOVNTPD *m128*, *xmm*	Valid	Valid	Move packed double-precision floating-point values from *xmm* to *m128* using non-temporal hint.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If the destination operand is in a nonwritable segment.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

MOVNTPS—Store Packed Single-Precision Floating-Point Values Using Non-Temporal Hint

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 2B /r	MOVNTPS *m128*, *xmm*	Valid	Valid	Move packed single-precision floating-point values from *xmm* to *m128* using non-temporal hint.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If the destination operand is in a nonwritable segment.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

MOVNTQ—Store of Quadword Using Non-Temporal Hint

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F E7 /r	MOVNTQ *m64*, *mm*	Valid	Valid	Move quadword from *mm* to *m64* using non-temporal hint.

IA-32e Mode Operation

Same as legacy mode.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	If the destination operand is in a nonwritable segment.
	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#MF	If there is a pending x87 FPU exception.
#UD	If EM in CR0 is set.
	If CPUID feature flag SSE is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#MF	If there is a pending x87 FPU exception.
#UD	If EM in CR0 is set.
	If CPUID feature flag SSE is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#GP(0)	If the destination operand is in a nonwritable segment.
#PF(fault-code)	For a page fault.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#MF	If there is a pending x87 FPU exception.
#UD	If EM in CR0 is set.
	If CPUID feature flag SSE is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

MOVQ—Move Quadword

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 6F /r	MOVQ *mm, mm/m64*	Valid	Valid	Move quadword from *mm/m64* to *mm*.
0F 7F /r	MOVQ *mm/m64, mm*	Valid	Valid	Move quadword from *mm* to *mm/m64*.
F3 0F 7E	MOVQ *xmm1*, xmm2/m64	Valid	Valid	Move quadword from *xmm2/mem64* to *xmm1*.
66 0F D6	MOVQ *xmm2/m64*, *xmm1*	Valid	Valid	Move quadword from *xmm1* to *xmm2/mem64*.

Flags Affected

None.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	If the destination operand is in a nonwritable segment.
	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(XMM register operations only.) If OSFXSR in CR4 is 0.
	(XMM register operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(MMX register operations only.) If there is a pending FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(XMM register operations only.) If OSFXSR in CR4 is 0.
	(XMM register operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(MMX register operations only.) If there is a pending FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the destination operand is in a nonwritable segment.
	If the memory address is in a non-canonical form.
#UD	If EM in CR0 is set.
	(XMM register operations only.) If OSFXSR in CR4 is 0.
	(XMM register operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(MMX register operations only.) If there is a pending FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

MOVQ2DQ—Move Quadword from MMX to XMM Register

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
F3 0F D6	MOVQ2DQ *xmm*, *mm*	Valid	Valid	Move quadword from *mmx* to low quadword of *xmm*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#MF	If there is a pending x87 FPU exception.

Real-Address Mode Exceptions

Same exceptions as in Protected Mode.

Virtual-8086 Mode Exceptions

Same exceptions as in Protected Mode.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

Same as for protected mode exceptions.

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
A4	MOVS m8, m8	Valid	Valid	For legacy mode, Move byte from address DS:(E)SI to ES:(E)DI. For 64-bit mode move byte from address (R)SI to (R)DI.
A5	MOVS m16, m16	Valid	Valid	For legacy mode, move word from address DS:(E)SI to ES:(E)DI. For 64-bit mode move word at address (R)SI to (R)DI.
A5	MOVS m32, m32	Valid	Valid	For legacy mode, move dword from address DS:(E)SI to ES:(E)DI. For 64-bit mode move dword from address (R)SI to (R)DI.
REX.W + A5	MOVS m64, m64	Valid	N.E.	Move qword from address (R)SI to (R)DI.
A4	MOVSB	Valid	Valid	For legacy mode, Move byte from address DS:(E)SI to ES:(E)DI. For 64-bit mode move byte from address (R)SI to (R)DI.
A5	MOVSW	Valid	Valid	For legacy mode, move word from address DS:(E)SI to ES:(E)DI. For 64-bit mode move word at address (R)SI to (R)DI.
A5	MOVSD	Valid	Valid	For legacy mode, move dword from address DS:(E)SI to ES:(E)DI. For 64-bit mode move dword from address (R)SI to (R)DI.
REX.W + A5	MOVSQ	Valid	N.E.	Move qword from address (R)SI to (R)DI.

Flags Affected

None.

IA-32e Mode Operation

Promoted to 64-bits.

Default operand size fixed at 32-bits.

Protected Mode Exceptions

#GP(0)	If the destination is located in a nonwritable segment.
	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the DS, ES, FS, or GS register contains a null segment selector.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS	If a memory operand effective address is outside the SS segment limit.

Virtual-8086 Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

MOVSD—Move Scalar Double-Precision Floating-Point Value

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
F2 0F 10 /r	MOVSD *xmm1*, *xmm2/m64*	Valid	Valid	Move scalar double-precision floating-point value from *xmm2/m64* to *xmm1* register.
F2 0F 11 /r	MOVSD *xmm2/m64*, *xmm1*	Valid	Valid	Move scalar double-precision floating-point value from *xmm1* register to *xmm2/m64*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	If the destination is located in a nonwritable segment.
	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

MOVSHDUP—Move Packed Single-Precision FP Values High and Duplicate

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
F3 0F 16 /r	MOVSHDUP *xmm1*, xmm2/ m128	Valid	Valid	Move two single-precision floating-point values from the higher 32-bit operand of each qword in *xmm2/m128* to *xmm1* and duplicate each 32-bit operand to the lower 32-bits of each qword.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE3 is 0.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE3 is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE3 is 0.

MOVSLDUP—Move Packed Single-Precision FP Values Low and Duplicate

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
F3 0F 12 /r	MOVSLDUP *xmm1*, xmm2/m128	Valid	Valid	Move two single-precision floating-point values from the lower 32-bit operand of each qword in *xmm2/m128* to *xmm1* and duplicate each 32-bit operand to the higher 32-bits of each qword.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE3 is 0.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE3 is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE3 is 0.

MOVSS—Move Scalar Single--Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
F3 0F 10 /r	MOVSS *xmm1*, *xmm2/m32*	Valid	Valid	Move scalar single-precision floating-point value from *xmm2/m32* to *xmm1* register.
F3 0F 11 /r	MOVSS *xmm2/m32*, *xmm*	Valid	Valid	Move scalar single-precision floating-point value from *xmm1* register to *xmm2/m32*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

MOVSX/MOVSXD—Move with Sign-Extension

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F BE /*r*	MOVSX *r16,r/m8*	Valid	Valid	Move byte to word with sign-extension
0F BE /*r*	MOVSX *r32,r/m8*	Valid	Valid	Move byte to doubleword with sign-extension
REX + 0F BE /*r*	MOVSX *r64,r/m8**	Valid	N.E.	Move byte to quadword with sign-extension
0F BF /*r*	MOVSX *r32,r/m16*	Valid	Valid	Move word to doubleword, with sign-extension
REX.W + 0F BF /*r*	MOVSX *r64,r/m16*	Valid	N.E.	Move word to quadword with sign-extension
REX.W** + 63 /*r*	MOVSXD *r64,r/m32*	Valid	N.E.	Move double word to quadword with sign-extension

* In 64-bit mode, r/m8 can not be encoded to access the following byte registers if an REX prefix is used: AH, BH, CH, DH. Also refer to Section 1.4.2.2.

** The use of MOVSXD without REX.W in 64-bit mode is discouraged, Regular MOV should be used instead of using MOVSXD without REX.W.

Flags Affected

None.

IA-32e Mode Operation

Promoted to 64-bits.

Default operand size 32-bits.

Enables access to new registers R8-R15.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the DS, ES, FS, or GS register contains a null segment selector.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS	If a memory operand effective address is outside the SS segment limit.

Virtual-8086 Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.#PF(fault-code)If a page fault occurs.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

MOVUPD—Move Unaligned Packed Double-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
66 0F 10 /r	MOVUPD *xmm1*, xmm2/m128	Valid	Valid	Move packed double-precision floating-point values from *xmm2/m128* to xmm1.
66 0F 11 /r	MOVUPD xmm2/m128, xmm	Valid	Valid	Move packed double-precision floating-point values from xmm1 to *xmm2/m128*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	If the destination operand is in a nonwritable segment.
	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Real-Address Mode Exceptions

#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

MOVUPS—Move Unaligned Packed Single-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 10 /r	MOVUPS *xmm1, xmm2/m128*	Valid	Valid	Move packed single-precision floating-point values from *xmm2/m128* to xmm1.
0F 11 /r	MOVUPS *xmm2/m128, xmm1*	Valid	Valid	Move packed single-precision floating-point values from xmm1 to *xmm2/m128*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	If the destination operand is in a nonwritable segment.
	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Real-Address Mode Exceptions

#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

MOVZX—Move with Zero-Extend

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F B6 /*r*	MOVZX *r16,r/m8*	Valid	Valid	Move byte to word with zero-extension
0F B6 /*r*	MOVZX *r32,r/m8*	Valid	Valid	Move byte to doubleword, zero-extension
REX + 0F B6 /*r*	MOVZX *r64,r/m8**	Valid	N.E.	Move byte to quadword, zero-extension
0F B7 /*r*	MOVZX *r32,r/m16*	Valid	Valid	Move word to doubleword, zero-extension
REX.W + 0F B7 /*r*	MOVZX *r64,r/m32*	Valid	N.E.	Move doubleword to quadword, zero-extension

* In 64-bit mode, r/m8 can not be encoded to access the following byte registers if an REX prefix is used: AH, BH, CH, DH. Also refer to Section 1.4.2.2.

Flags Affected

None.

IA-32e Mode Operation

Promoted to 64-bits.

Default operand size 32-bits.

Enables access to new registers R8-R15.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the DS, ES, FS, or GS register contains a null segment selector.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS	If a memory operand effective address is outside the SS segment limit.

Virtual-8086 Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

MUL—Unsigned Multiply

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
F6 /4	MUL *r/m8*	Valid	Valid	Unsigned multiply (AX ← AL ∗ *r/m8*)
REX + F6 /4	MUL *r/m8**	Valid	N.E.	Unsigned multiply (AX ← AL ∗ *r/m8*)
F7 /4	MUL *r/m16*	Valid	Valid	Unsigned multiply (DX:AX ← AX ∗ *r/m16*)
F7 /4	MUL *r/m32*	Valid	Valid	Unsigned multiply (EDX:EAX ← EAX ∗ *r/m32*)
REX.W + F7 /4	MUL *r/m64*	Valid	N.E.	Unsigned multiply (RDX:RAX ← RAX ∗ *r/m64*

* In 64-bit mode, r/m8 can not be encoded to access the following byte registers if an REX prefix is used: AH, BH, CH, DH. Also refer to Section 1.4.2.2.

Flags Affected

The OF and CF flags are cleared to 0 if the upper half of the result is 0; otherwise, they are set to 1. The SF, ZF, AF, and PF flags are undefined.

IA-32e Mode Operation

Promoted to 64-bits.

Default operand size 32-bits.

RDX:RAX = RAX * Quadword in register or memory.

Enables access to new registers R8-R15.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the DS, ES, FS, or GS register contains a null segment selector.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS	If a memory operand effective address is outside the SS segment limit.

Virtual-8086 Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

MULPD—Multiply Packed Double-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
66 0F 59 /r	MULPD *xmm1, xmm2/m128*	Valid	Valid	Multiply packed double-precision floating-point values in *xmm2/m128* by *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

Overflow, Underflow, Invalid, Precision, Denormal.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

MULPS—Multiply Packed Single-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 59 /r	MULPS *xmm1*, *xmm2/m128*	Valid	Valid	Multiply packed single-precision floating-point values in *xmm2/mem* by *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

Overflow, Underflow, Invalid, Precision, Denormal.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

MULSD—Multiply Scalar Double-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
F2 0F 59 /r	MULSD *xmm1*, *xmm2/m64*	Valid	Valid	Multiply the low double-precision floating-point value in *xmm2/mem64* by low double-precision floating-point value in *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

Overflow, Underflow, Invalid, Precision, Denormal.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC	For unaligned memory reference if the current privilege level is 3.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

MULSS—Multiply Scalar Single-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
F3 0F 59 /r	MULSS *xmm1*, *xmm2/ m32*	Valid	Valid	Multiply the low single-precision floating-point value in *xmm2/mem* by the low single-precision floating-point value in *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

Overflow, Underflow, Invalid, Precision, Denormal.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC	For unaligned memory reference if the current privilege level is 3.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

MWAIT—Monitor Wait

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
OF 01 *C9*	MWAIT EAX, *ECX*	Valid	Valid	A hint that allow the processor to stop instruction execution and enter an implementation-dependent optimized state until occurrence of a class of events.

Flags Affected

None.

IA-32e Mode Operation

Same as in protected mode.

Protected Mode Exceptions

#GP(0)	If the current privilege level is not 0.
	If ECX<> 0.
#UD	If CPUID.MONITOR (ECX bit 3) = 0.
	If the F3H, F2H, 66H or LOCK prefix is used.

Real-Address Mode Exceptions

#GP(0)	If the current privilege level is not 0.
	If ECX<> 0.
#UD	If CPUID.MONITOR (ECX bit 3) = 0.
	If the F3H, F2H, 66H or LOCK prefix is used.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#GP(0)	If the current privilege level is not 0.
	If ECX != 0.
#UD	If CPUID.MONITOR (ECX bit 3) = 0.
	If the F3H, F2H, 66H or LOCK prefix is used.

NEG—Two's Complement Negation

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
F6 /3	NEG *r/m8*	Valid	Valid	Two's complement negate *r/m8*
REX + F6 /3	NEG *r/m8**	Valid	N.E.	Two's complement negate *r/m8*
F7 /3	NEG *r/m16*	Valid	Valid	Two's complement negate *r/m16*
F7 /3	NEG *r/m32*	Valid	Valid	Two's complement negate *r/m32*
REX.W + F7 /3	NEG *r/m64*	Valid	N.E.	Two's complement negate *r/m64*

* In 64-bit mode, r/m8 can not be encoded to access the following byte registers if an REX prefix is used: AH, BH, CH, DH. Also refer to Section 1.4.2.2.

Flags Affected

The CF flag cleared to 0 if the source operand is 0; otherwise it is set to 1. The OF, SF, ZF, AF, and PF flags are set according to the result.

IA-32e Mode Operation

Promoted to 64-bits.

Default operand size 32-bits.

Enables access to new registers R8-R15.

Protected Mode Exceptions

#GP(0)	If the destination is located in a nonwritable segment.
	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the DS, ES, FS, or GS register contains a null segment selector.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS	If a memory operand effective address is outside the SS segment limit.

Virtual-8086 Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	For a page fault.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

NOP—No Operation

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
90	NOP	Valid	Valid	No operation

Flags Affected

None.

IA-32e Mode Operation

Same as legacy mode

Exceptions (All Operating Modes)

None.

NOT—One's Complement Negation

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
F6 /2	NOT *r/m8*	Valid	Valid	Reverse each bit of *r/m8*
REX + F6 /2	NOT *r/m8**	Valid	N.E.	Reverse each bit of *r/m8*
F7 /2	NOT *r/m16*	Valid	Valid	Reverse each bit of *r/m16*
F7 /2	NOT *r/m32*	Valid	Valid	Reverse each bit of *r/m32*
REX.W + F7 /2	NOT *r/m64*	Valid	N.E.	Reverse each bit of *r/m64*

* In 64-bit mode, r/m8 can not be encoded to access the following byte registers if an REX prefix is used: AH, BH, CH, DH. Also refer to Section 1.4.2.2.

Flags Affected

None.

IA-32e Mode Operation

Promoted to 64-bits.

Default operand size 32-bits.

Enables access to new registers R8-R15.

Protected Mode Exceptions

#GP(0)	If the destination operand points to a nonwritable segment.
	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the DS, ES, FS, or GS register contains a null segment selector.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS	If a memory operand effective address is outside the SS segment limit.

Virtual-8086 Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

OR—Logical Inclusive OR

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0C *ib*	OR AL,*imm8*	Valid	Valid	AL OR *imm8*
0D *iw*	OR AX,*imm16*	Valid	Valid	AX OR *imm16*
0D *id*	OR EAX,*imm32*	Valid	Valid	EAX OR *imm32*
REX.W + 0D *id*	OR RAX,*imm64*	Valid	N.E.	RAX OR *imm32 (sign-extended)*
80 /1 *ib*	OR *r/m8,imm8*	Valid	Valid	*r/m8* OR *imm8*
REX + 80 /1 *ib*	OR *r/m8*,imm8*	Valid	N.E.	*r/m8* OR *imm8*
81 /1 *iw*	OR *r/m16,imm16*	Valid	Valid	*r/m16* OR *imm16*
81 /1 *id*	OR *r/m32,imm32*	Valid	Valid	*r/m32* OR *imm32*
REX.W + 81 /1 *id*	OR *r/m64,imm32*	Valid	N.E.	*r/m64* OR *imm32 (sign-extended)*
83 /1 *ib*	OR *r/m16,imm8*	Valid	Valid	*r/m16* OR *imm8 (sign-extended)*
83 /1 *ib*	OR *r/m32,imm8*	Valid	Valid	*r/m32* OR *imm8 (sign-extended)*
REX.W + 83 /1 *ib*	OR *r/m64,imm8*	Valid	N.E.	*r/m64* OR *imm8 (sign-extended)*
08 /*r*	OR *r/m8,r8*	Valid	Valid	*r/m8* OR *r8*
REX + 08 /*r*	OR *r/m8*,r8**	Valid	N.E.	*r/m8* OR *r8*
09 /*r*	OR *r/m16,r16*	Valid	Valid	*r/m16* OR *r16*
09 /*r*	OR *r/m32,r32*	Valid	Valid	*r/m32* OR *r32*
REX.W + 09 /*r*	OR *r/m64,r64*	Valid	N.E.	*r/m64* OR *r64*
0A /*r*	OR *r8,r/m8*	Valid	Valid	*r8* OR *r/m8*
REX + 0A /*r*	OR *r8*,r/m8**	Valid	N.E.	*r8* OR *r/m8*
0B /*r*	OR *r16,r/m16*	Valid	Valid	*r16* OR *r/m16*
0B /*r*	OR *r32,r/m32*	Valid	Valid	*r32* OR *r/m32*
REX.W + 0B /*r*	OR *r64,r/m64*	Valid	N.E.	*r64* OR *r/m64*

* In 64-bit mode, r/m8 can not be encoded to access the following byte registers if an REX prefix is used: AH, BH, CH, DH. Also refer to Section 1.4.2.2.

Flags Affected

The OF and CF flags are cleared; the SF, ZF, and PF flags are set according to the result. The state of the AF flag is undefined.

IA-32e Mode Operation

Promoted to 64-bits.

Default operand size 32-bits.

Enables access to new registers R8-R15.

Protected Mode Exceptions

#GP(0)	If the destination operand points to a nonwritable segment.
	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the DS, ES, FS, or GS register contains a null segment selector.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS	If a memory operand effective address is outside the SS segment limit.

Virtual-8086 Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

ORPD—Bitwise Logical OR of Double-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
66 0F 56 /r	ORPD *xmm1, xmm2/m128*	Valid	Valid	Bitwise OR of *xmm2/m128* and *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

ORPS—Bitwise Logical OR of Single-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 56 /r	ORPS *xmm1*, *xmm2/m128*	Valid	Valid	Bitwise OR of *xmm2/m128* and *xmm1*

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

OUT—Output to Port

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
E6 *ib*	OUT *imm8*, AL	Valid	Valid	Output byte in AL to I/O port address *imm8*
E7 *ib*	OUT *imm8*, AX	Valid	Valid	Output word in AX to I/O port address *imm8*
E7 *ib*	OUT *imm8*, EAX	Valid	Valid	Output doubleword in EAX to I/O port address *imm8*
REX.W + E7 *ib*	OUT *imm8*, RAX	N.P.	N.E.	REX does not change ensuing instruction
EE	OUT DX, AL	Valid	Valid	Output byte in AL to I/O port address in DX
EF	OUT DX, AX	Valid	Valid	Output word in AX to I/O port address in DX
EF	OUT DX, EAX	Valid	Valid	Output doubleword in EAX to I/O port address in DX
REX.W + EF	OUT DX, RAX	N.P.	N.E.	REX does not change ensuing instruction)

Flags Affected

None.

IA-32e Mode Operation

Same as legacy mode

Protected Mode Exceptions

#GP(0) If the CPL is greater than (has less privilege) the I/O privilege level (IOPL) and any of the corresponding I/O permission bits in TSS for the I/O port being accessed is 1.

Real-Address Mode Exceptions

None.

Virtual-8086 Mode Exceptions

#GP(0) If any of the I/O permission bits in the TSS for the I/O port being accessed is 1.

Compatibility Mode Exceptions

Same as protected mode exceptions.

64-Bit Mode Exceptions

#GP(0) If the CPL is greater than (has less privilege) the I/O privilege level (IOPL) and any of the corresponding I/O permission bits in TSS for the I/O port being accessed is 1.

OUTS/OUTSB/OUTSW/OUTSD—Output String to Port

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
6E	OUTS DX, m8	Valid	Valid	Output byte from memory location specified in DS:(E)SI to I/O port specified in DX
REX.W + 6E	OUTS DX, m8	Valid	N.E.	Output byte from memory location specified in RSI to I/O port specified in DX
6F	OUTS DX, m16	Valid	Valid	Output word from memory location specified in DS:(E)SI to I/O port specified in DX
6F	OUTS DX, m32	Valid	Valid	Output doubleword from memory location specified in DS:(E)SI to I/O port specified in DX
REX.W + 6F	OUTS DX, m32	N.P.	N.E.	Output default size from memory location specified in RSI to I/O port specified in DX
6E	OUTSB	Valid	Valid	Output byte from memory location specified in DS:(E)SI to I/O port specified in DX
REX.W + 6E	OUTSB	Valid	N.E.	Output byte from memory location specified in RSI to I/O port specified in DX
6F	OUTSW	Valid	Valid	Output word from memory location specified in DS:(E)SI to I/O port specified in DX
6F	OUTSD	Valid	Valid	Output doubleword from memory location specified in DS:(E)SI to I/O port specified in DX
REX.W + 6F	OUTSD	N.P.	N.E.	Output default size from memory location specified in RSI to I/O port specified in DX

Flags Affected

None.

IA-32e Mode Operation

Default operand size is 32 bits and is not promoted by REX.W.

64-bit mode enables the use of RSI.

Protected Mode Exceptions

#GP(0)	If the CPL is greater than (has less privilege) the I/O privilege level (IOPL) and any of the corresponding I/O permission bits in TSS for the I/O port being accessed is 1.
	If a memory operand effective address is outside the limit of the CS, DS, ES, FS, or GS segment.
	If the segment register contains a null segment selector.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS	If a memory operand effective address is outside the SS segment limit.

Virtual-8086 Mode Exceptions

#GP(0)	If any of the I/O permission bits in the TSS for the I/O port being accessed is 1.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the CPL is greater than (has less privilege) the I/O privilege level (IOPL) and any of the corresponding I/O permission bits in TSS for the I/O port being accessed is 1.
	If the memory address is in a non-canonical form.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

PACKSSWB/PACKSSDW—Pack with Signed Saturation

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 63 /r	PACKSSWB *mm1, mm2/m64*	Valid	Valid	Converts 4 packed signed word integers from *mm1* and from *mm2/m64* into 8 packed signed byte integers in *mm1* using signed saturation.
66 0F 63 /r	PACKSSWB *xmm1, xmm2/m128*	Valid	Valid	Converts 8 packed signed word integers from *xmm1* and from *xxm2/m128* into 16 packed signed byte integers in *xxm1* using signed saturation.
0F 6B /r	PACKSSDW *mm1, mm2/m64*	Valid	Valid	Converts 2 packed signed doubleword integers from *mm1* and from *mm2/m64* into 4 packed signed word integers in *mm1* using signed saturation.
66 0F 6B /r	PACKSSDW *xmm1, xmm2/m128*	Valid	Valid	Converts 4 packed signed doubleword integers from *xmm1* and from *xxm2/m128* into 8 packed signed word integers in *xxm1* using signed saturation.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

PACKUSWB—Pack with Unsigned Saturation

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 67 /r	PACKUSWB *mm, mm/m64*	Valid	Valid	Converts 4 signed word integers from *mm* and 4 signed word integers from *mm/m64* into 8 unsigned byte integers in *mm* using unsigned saturation.
66 0F 67 /r	PACKUSWB *xmm1, xmm2/m128*	Valid	Valid	Converts 8 signed word integers from *xmm1* and 8 signed word integers from *xmm2/m128* into 16 unsigned byte integers in *xmm1* using unsigned saturation.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

PADDB/PADDW/PADDD—Add Packed Integers

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F FC /r	PADDB *mm, mm/m64*	Valid	Valid	Add packed byte integers from *mm/m64* and *mm*.
66 0F FC /r	PADDB *xmm1,xmm2/m128*	Valid	Valid	Add packed byte integers from *xmm2/m128* and *xmm1*.
0F FD /r	PADDW *mm, mm/m64*	Valid	Valid	Add packed word integers from *mm/m64* and *mm*.
66 0F FD /r	PADDW *xmm1, xmm2/m128*	Valid	Valid	Add packed word integers from *xmm2/ m128* and *xmm1*.
0F FE /r	PADDD *mm, mm/m64*	Valid	Valid	Add packed doubleword integers from *mm/ m64* and *mm*.
66 0F FE /r	PADDD *xmm1, xmm2/m128*	Valid	Valid	Add packed doubleword integers from *xmm2/m128* and *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

PADDQ—Add Packed Quadword Integers

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F D4 /r	PADDQ *mm1,mm2/m64*	Valid	Valid	Add quadword integer *mm2/m64* to *mm1*
66 0F D4 /r	PADDQ *xmm1,xmm2/m128*	Valid	Valid	Add packed quadword integers *xmm2/m128* to *xmm1*

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Numeric Exceptions

None.

PADDSB/PADDSW—Add Packed Signed Integers with Signed Saturation

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F EC /r	PADDSB *mm, mm/m64*	Valid	Valid	Add packed signed byte integers from *mm/m64 and mm* and saturate the results.
66 0F EC /r	PADDSB *xmm1, xmm2/ m128*	Valid	Valid	Add packed signed byte integers from *xmm2/m128* and *xmm1* saturate the results.
0F ED /r	PADDSW *mm, mm/m64*	Valid	Valid	Add packed signed word integers from *mm/m64 and mm* and saturate the results.
66 0F ED /r	PADDSW *xmm1, xmm2/m128*	Valid	Valid	Add packed signed word integers from *xmm2/m128* and *xmm1* and saturate the results.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

PADDUSB/PADDUSW—Add Packed Unsigned Integers with Unsigned Saturation

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F DC /r	PADDUSB *mm, mm/m64*	Valid	Valid	Add packed unsigned byte integers from *mm/m64 and mm* and saturate the results.
66 0F DC /r	PADDUSB xmm1, xmm2/ m128	Valid	Valid	Add packed unsigned byte integers from *xmm2/ m128* and *xmm1* saturate the results.
0F DD /r	PADDUSW *mm, mm/m64*	Valid	Valid	Add packed unsigned word integers from *mm/m64 and mm* and saturate the results.
66 0F DD /r	PADDUSW xmm1, xmm2/ m128	Valid	Valid	Add packed unsigned word integers from *xmm2/ m128* to *xmm1* and saturate the results.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Numeric Exceptions

None.

PAND—Logical AND

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F DB /r	PAND *mm, mm/m64*	Valid	Valid	Bitwise AND *mm/m64* and mm.
66 0F DB /r	PAND *xmm1*, xmm2/m128	Valid	Valid	Bitwise AND of *xmm2/m128* and *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Numeric Exceptions

None.

PANDN—Logical AND NOT

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F DF /r	PANDN *mm, mm/m64*	Valid	Valid	Bitwise AND NOT of *mm/m64* and *mm*.
66 0F DF /r	PANDN *xmm1, xmm2/m128*	Valid	Valid	Bitwise AND NOT of *xmm2/m128* and *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Numeric Exceptions

None.

PAUSE—Spin Loop Hint

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
F3 90	PAUSE	Valid	Valid	Gives hint to processor that improves performance of spin-wait loops.

IA-32e Mode Operation

Same as legacy mode.

Exceptions (All Operating Modes)

None.

Numeric Exceptions

None.

PAVGB/PAVGW—Average Packed Integers

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F E0 /r	PAVGB *mm1, mm2/ m64*	Valid	Valid	Average packed unsigned byte integers from mm2/ m64 and mm1 with rounding.
66 0F E0, /r	PAVGB *xmm1*, *xmm2/ m128*	Valid	Valid	Average packed unsigned byte integers from *xmm2/ m128* and *xmm1* with rounding.
0F E3 /r	PAVGW *mm1, mm2/ m64*	Valid	Valid	Average packed unsigned word integers from mm2/ m64 and mm1 with rounding.
66 0F E3 /r	PAVGW *xmm1*, *xmm2/ m128*	Valid	Valid	Average packed unsigned word integers from *xmm2/m128* and *xmm1* with rounding.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Numeric Exceptions

None.

PCMPEQB/PCMPEQW/PCMPEQD— Compare Packed Data for Equal

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 74 /r	PCMPEQB *mm, mm/m64*	Valid	Valid	Compare packed bytes in *mm/m64* and *mm* for equality.
66 0F 74 /r	PCMPEQB *xmm1*, *xmm2/m128*	Valid	Valid	Compare packed bytes in *xmm2/m128* and xmm1 for equality.
0F 75 /r	PCMPEQW *mm, mm/m64*	Valid	Valid	Compare packed words in *mm/m64* and *mm* for equality.
66 0F 75 /r	PCMPEQW *xmm1*, *xmm2/m128*	Valid	Valid	Compare packed words in *xmm2/m128* and xmm1 for equality.
0F 76 /r	PCMPEQD *mm, mm/m64*	Valid	Valid	Compare packed doublewords in *mm/m64* and *mm* for equality.
66 0F 76 /r	PCMPEQD *xmm1*, *xmm2/m128*	Valid	Valid	Compare packed doublewords in *xmm2/m128* and xmm1 for equality.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Numeric Exceptions

None.

PCMPGTB/PCMPGTW/PCMPGTD—Compare Packed Signed Integers for Greater Than

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 64 /r	PCMPGTB *mm, mm/m64*	Valid	Valid	Compare packed signed byte integers in *mm* and *mm/m64* for greater than.
66 0F 64 /r	PCMPGTB *xmm1*, *xmm2/m128*	Valid	Valid	Compare packed signed byte integers in *xmm1* and *xmm2/m128* for greater than.
0F 65 /r	PCMPGTW *mm, mm/m64*	Valid	Valid	Compare packed signed word integers in *mm* and *mm/m64* for greater than.
66 0F 65 /r	PCMPGTW *xmm1*, *xmm2/m128*	Valid	Valid	Compare packed signed word integers in *xmm1* and *xmm2/m128* for greater than.
0F 66 /r	PCMPGTD *mm, mm/m64*	Valid	Valid	Compare packed signed doubleword integers in *mm* and *mm/m64* for greater than.
66 0F 66 /r	PCMPGTD *xmm1*, *xmm2/m128*	Valid	Valid	Compare packed signed doubleword integers in *xmm1* and *xmm2/m128* for greater than.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Numeric Exceptions

None.

PEXTRW—Extract Word

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F C5 /r ib	PEXTRW *r32*, *mm*, *imm8*	Valid	Valid	Extract the word specified by *imm8* from *mm* and move it to *r32*.
REX.W + 0F C5 /r ib	PEXTRW *r64*, *mm*, *imm8*	Valid	N.E.	Extract the word specified by *imm8* from *mm* and move it to *r16*. Zero Extend 16-bit results to 64-bits
66 0F C5 /r ib	PEXTRW *r32*, *xmm*, *imm8*	Valid	Valid	Extract the word specified by *imm8* from *xmm* and move it to a *r32*.
REX.W + 66 0F C5 /r ib	PEXTRW *r64*, *xmm*, *imm8*	Valid	N.E.	Extract the word specified by *imm8* from *xmm* and move it to a *r16*. Zero extend 16-bit results to 64-bits

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Real-Address Mode Exceptions

#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Numeric Exceptions

None.

PINSRW—Insert Word

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F C4 /r ib	PINSRW *mm*, *r32/m16, imm8*	Valid	Valid	Insert the low word from *r32* or from *m16* into *mm* at the word position specified by *imm8*
REX.W + 0F C4 /r ib	PINSRW *mm*, *r64/m16, imm8*	Valid	N.E.	Insert the low word from *r64* or from *m16* into *mm* at the word position specified by *imm8*
66 0F C4 /r ib	PINSRW xmm, *r32/m16*, imm8	Valid	Valid	Move the low word of *r32* or from *m16* into xmm at the word position specified by *imm8*.
REX.W + 66 0F C4 /r ib	PINSRW xmm, *r64/m16*, imm8	Valid	N.E.	Move the low word of *r64* or from *m16* into xmm at the word position specified by *imm8*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Numeric Exceptions

None.

PMADDWD—Multiply and Add Packed Integers

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F F5 /r	PMADDWD *mm, mm/m64*	Valid	Valid	Multiply the packed words in *mm* by the packed words in *mm/m64*, add adjacent doubleword results, and store in *mm*.
66 0F F5 /r	PMADDWD *xmm1, xmm2/m128*	Valid	Valid	Multiply the packed word integers in *xmm1* by the packed word integers in *xmm2/m128*, add adjacent doubleword results, and store in *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Numeric Exceptions

None.

PMAXSW—Maximum of Packed Signed Word Integers

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F EE /r	PMAXSW *mm1, mm2/m64*	Valid	Valid	Compare signed word integers in *mm2/m64* and *mm1* and return maximum values.
66 0F EE /r	PMAXSW *xmm1*, *xmm2/m128*	Valid	Valid	Compare signed word integers in *xmm2/m128* and *xmm1* and return maximum values.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Numeric Exceptions

None.

PMAXUB—Maximum of Packed Unsigned Byte Integers

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F DE /r	PMAXUB *mm1, mm2/m64*	Valid	Valid	Compare unsigned byte integers in *mm2/m64* and *mm1* and returns maximum values.
66 0F DE /r	PMAXUB *xmm1*, *xmm2/m128*	Valid	Valid	Compare unsigned byte integers in *xmm2/m128* and *xmm1* and returns maximum values.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Exceptions

None.

PMINSW—Minimum of Packed Signed Word Integers

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F EA /r	PMINSW *mm1, mm2/m64*	Valid	Valid	Compare signed word integers in *mm2/m64* and *mm1* and return minimum values.
66 0F EA /r	PMINSW *xmm1*, *xmm2/m128*	Valid	Valid	Compare signed word integers in *xmm2/m128* and *xmm1* and return minimum values.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Numeric Exceptions

None.

PMINUB—Minimum of Packed Unsigned Byte Integers

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F DA /r	PMINUB *mm1, mm2/ m64*	Valid	Valid	Compare unsigned byte integers in *mm2/m64* and *mm1* and returns minimum values.
66 0F DA /r	PMINUB *xmm1, xmm2/ m128*	Valid	Valid	Compare unsigned byte integers in *xmm2/m128* and *xmm1* and returns minimum values.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Numeric Exceptions

None.

PMOVMSKB—Move Byte Mask

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F D7 /r	PMOVMSKB *r32*, *mm*	Valid	Valid	Move a byte mask of *mm* to *r32*.
REX.W + 0F D7 /r	PMOVMSKB *r64*, *mm*	Valid	N.E.	Move a byte mask of *mm* to *the lower 32-bits of r64 and zero upper 32-bits*.
66 0F D7 /r	PMOVMSKB *r32*, *xmm*	Valid	Valid	Move a byte mask of *xmm* to *r32*.
REX.W + 66 0F D7 /r	PMOVMSKB *r64*, *xmm*	Valid	N.E.	Move a byte mask of x*mm* to *the lower 32-bits of r64 and zero upper 32-bits*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Real-Address Mode Exceptions

Same exceptions as in Protected Mode.

Virtual-8086 Mode Exceptions

Same exceptions as in Protected Mode.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

Same as for protected mode exceptions.

Numeric Exceptions

None.

PMULHUW—Multiply Packed Unsigned Integers and Store High Result

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F E4 /r	PMULHUW *mm1*, *mm2/m64*	Valid	Valid	Multiply the packed unsigned word integers in *mm1* register and *mm2/m64*, and store the high 16 bits of the results in *mm1*.
66 0F E4 /r	PMULHUW *xmm1*, *xmm2/m128*	Valid	Valid	Multiply the packed unsigned word integers in *xmm1* and *xmm2/m128*, and store the high 16 bits of the results in *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

IFlags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Numeric Exceptions

None.

PMULHW—Multiply Packed Signed Integers and Store High Result

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F E5 /r	PMULHW *mm, mm/m64*	Valid	Valid	Multiply the packed signed word integers in *mm1* register and *mm2/m64*, and store the high 16 bits of the results in *mm1*.
66 0F E5 /r	PMULHW *xmm1*, *xmm2/m128*	Valid	Valid	Multiply the packed signed word integers in *xmm1* and *xmm2/m128*, and store the high 16 bits of the results in *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#GP(0)	If the memory address is in a non-canonical form.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Numeric Exceptions

None.

PMULLW—Multiply Packed Signed Integers and Store Low Result

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F D5 /r	PMULLW *mm, mm/m64*	Valid	Valid	Multiply the packed signed word integers in *mm1* register and *mm2/m64*, and store the low 16 bits of the results in *mm1*.
66 0F D5 /r	PMULLW *xmm1*, *xmm2/m128*	Valid	Valid	Multiply the packed signed word integers in *xmm1* and *xmm2/m128*, and store the low 16 bits of the results in *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Numeric Exceptions

None.

PMULUDQ—Multiply Packed Unsigned Doubleword Integers

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F F4 /r	PMULUDQ *mm1*, *mm2/m64*	Valid	Valid	Multiply unsigned doubleword integer in *mm1* by unsigned doubleword integer in *mm2/m64*, and store the quadword result in *mm1*.
66 OF F4 /r	PMULUDQ *xmm1*, *xmm2/m128*	Valid	Valid	Multiply packed unsigned doubleword integers in *xmm1* by packed unsigned doubleword integers in *xmm2/m128*, and store the quadword results in *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

POP—Pop a Value from the Stack

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
8F /0	POP *m16*	Valid	Valid	Pop top of stack into *m16*; increment stack pointer
8F /0	POP *m32*	N.E.	Valid	Pop top of stack into *m32*; increment stack pointer
REX.W + 8F /0	POP *m64*	Valid	N.E.	Pop top of stack into *m64*; increment stack pointer. Cannot encode 32-bit operand size.
58+ *rw*	POP *r16*	Valid	Valid	Pop top of stack into *r16*; increment stack pointer
58+ *rd*	POP *r32*	N.E.	Valid	Pop top of stack into *r32*; increment stack pointer
REX.W + 58+ *rd*	POP *r64*	Valid	N.E.	Pop top of stack into *r64*; increment stack pointer. Cannot encode 32-bit operand size.
1F	POP DS	Inv.	Valid	Pop top of stack into DS; increment stack pointer
07	POP ES	Inv.	Valid	Pop top of stack into ES; increment stack pointer
17	POP SS	Inv.	Valid	Pop top of stack into SS; increment stack pointer
0F A1	POP FS	Valid	Valid	Pop top of stack into FS; increment stack pointer by 16 bits.
0F A1	POP FS	N.E.	Valid	Pop top of stack into FS; increment stack pointer by 32 bits.
0F A1	POP FS	Valid	Valid	Pop top of stack into FS; increment stack pointer by 64 bits.
0F A9	POP GS	Valid	Valid	Pop top of stack into GS; increment stack pointer by 16 bits.
0F A9	POP GS	N.E.	Valid	Pop top of stack into GS; increment stack pointer by 32 bits.
0F A9	POP GS	Valid	Valid	Pop top of stack into GS; increment stack pointer by 64 bits.

Flags Affected

None.

IA-32e Mode Operation

See Table above.

Protected Mode Exceptions

#GP(0)	If attempt is made to load SS register with null segment selector.
	If the destination operand is in a nonwritable segment.
	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the DS, ES, FS, or GS register is used to access memory and it contains a null segment selector.
#GP(selector)	If segment selector index is outside descriptor table limits.
	If the SS register is being loaded and the segment selector's RPL and the segment descriptor's DPL are not equal to the CPL.
	If the SS register is being loaded and the segment pointed to is a nonwritable data segment.
	If the DS, ES, FS, or GS register is being loaded and the segment pointed to is not a data or readable code segment.
	If the DS, ES, FS, or GS register is being loaded and the segment pointed to is a data or nonconforming code segment, but both the RPL and the CPL are greater than the DPL.
#SS(0)	If the current top of stack is not within the stack segment.
	If a memory operand effective address is outside the SS segment limit.
#SS(selector)	If the SS register is being loaded and the segment pointed to is marked not present.
#NP	If the DS, ES, FS, or GS register is being loaded and the segment pointed to is marked not present.

#PF(fault-code)	If a page fault occurs.
#AC(0)	If an unaligned memory reference is made while the current privilege level is 3 and alignment checking is enabled.

Real-Address Mode Exceptions

#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.

Virtual-8086 Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If an unaligned memory reference is made while alignment checking is enabled.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#GP(0)	If the memory address is in a non-canonical form.
#SS(U)	If the stack address is in a non-canonical form.
#GP(selector)	If the descriptor is outside the descriptor table limit.
	If the FS or GS register is being loaded and the segment pointed to is not a data or readable code segment.
	If the FS or GS register is being loaded and the segment pointed to is a data or nonconforming code segment, but both the RPL and the CPL are greater than the DPL.
#AC(0)	If an unaligned memory reference is made while alignment checking is enabled.
#PF(fault-code)	If a page fault occurs.
#NP	If the FS or GS register is being loaded and the segment pointed to is marked not present.

POPA/POPAD—Pop All General-Purpose Registers

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
61	POPA	Inv.	Valid	Pop DI, SI, BP, BX, DX, CX, and AX
61	POPAD	Inv.	Valid	Pop EDI, ESI, EBP, EBX, EDX, ECX, and EAX

Flags Affected

None.

IA-32e Mode Operation

Invalid in 64-bit mode.

Protected Mode Exceptions

#SS(0)	If the starting or ending stack address is not within the stack segment.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If an unaligned memory reference is made while the current privilege level is 3 and alignment checking is enabled.

Real-Address Mode Exceptions

#SS	If the starting or ending stack address is not within the stack segment.

Virtual-8086 Mode Exceptions

#SS(0)	If the starting or ending stack address is not within the stack segment.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If an unaligned memory reference is made while alignment checking is enabled.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#AC(0)	If an unaligned memory reference is made while alignment checking is enabled.
#PF(fault-code)	If a page fault occurs.

POPF/POPFD—Pop Stack into EFLAGS Register

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
9D	POPF	Valid	Valid	Pop top of stack into lower 16 bits of EFLAGS.
9D	POPFD	N.E.	Valid	Pop top of stack into EFLAGS.
REX.W + 9D	POPFQ	Valid	N.E.	Pop top of stack and zero-extend into RFLAGS.

Flags Affected

All flags except the reserved bits and the VM bit.

IA-32e Mode Operation

Promoted to 64-bits.

Default operand size 64-bits. Pops 64 bits from stack and loads lower 32 bits into EFLAGS zero extending the upper 32-bits of RFLAGS.

Cannot encode 32-bit operand size.

Protected Mode Exceptions

#SS(0)	If the top of stack is not within the stack segment.
#PF(fault-code)	If a page fault occurs.

#AC(0)If an unaligned memory reference is made while the current privilege level is 3 and alignment checking is enabled.

Real-Address Mode Exceptions

#SS	If the top of stack is not within the stack segment.

Virtual-8086 Mode Exceptions

#GP(0)	If the I/O privilege level is less than 3.
	If an attempt is made to execute the POPF/POPFD instruction with an operand-size override prefix.
#SS(0)	If the top of stack is not within the stack segment.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If an unaligned memory reference is made while alignment checking is enabled.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#GP(0)	If the memory address is in a non-canonical form.
#SS(U)	If the stack address is in a non-canonical form.

POR—Bitwise Logical OR

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F EB /r	POR *mm, mm/m64*	Valid	Valid	Bitwise OR of *mm/m64* and *mm*.
66 0F EB /r	POR *xmm1*, *xmm2/m128*	Valid	Valid	Bitwise OR of *xmm2/m128* and *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Intel C/C++ Compiler Intrinsic Equivalent

POR __m64 _mm_or_si64(__m64 m1, __m64 m2)

POR __m128i _mm_or_si128(__m128i m1, __m128i m2)

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Numeric Exceptions

None.

PREFETCH*h*—Prefetch Data Into Caches

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 18 /1	PREFETCHT0 *m8*	Valid	Valid	Move data from *m8* closer to the processor using T0 hint.
0F 18 /2	PREFETCHT1 *m8*	Valid	Valid	Move data from *m8* closer to the processor using T1 hint.
0F 18 /3	PREFETCHT2 *m8*	Valid	Valid	Move data from *m8* closer to the processor using T2 hint.
0F 18 /0	PREFETCHNTA *m8*	Valid	Valid	Move data from *m8* closer to the processor using NTA hint.

IA-32e Mode Operation

Same as legacy mode.

Numeric Exceptions

None.

Protected Mode Exceptions

None.

Real Address Mode Exceptions

None.

Virtual 8086 Mode Exceptions

None.

64-Bit Mode Exceptions

None.

PSADBW—Compute Sum of Absolute Differences

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F F6 /r	PSADBW *mm1, mm2/ m64*	Valid	Valid	Computes the absolute differences of the packed unsigned byte integers from *mm2 /m64* and *mm1*; differences are then summed to produce an unsigned word integer result.
66 0F F6 /r	PSADBW *xmm1*, *xmm2/m128*	Valid	Valid	Computes the absolute differences of the packed unsigned byte integers from *xmm2 /m128* and *xmm1*; the 8 low differences and 8 high differences are then summed separately to produce two unsigned word integer results.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Numeric Exceptions

None.

PSHUFD—Shuffle Packed Doublewords

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
66 0F 70 /r ib	PSHUFD *xmm1*, *xmm2/m128*, *imm8*	Valid	Valid	Shuffle the doublewords in *xmm2/m128* based on the encoding in *imm8* and store the result in *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#PF(fault-code)	If a page fault occurs.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#PF(fault-code)	If a page fault occurs.

Numeric Exceptions

None.

PSHUFHW—Shuffle Packed High Words

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
F3 0F 70 /r ib	PSHUFHW *xmm1*, *xmm2/m128*, *imm8*	Valid	Valid	Shuffle the high words in *xmm2/m128* based on the encoding in *imm8* and store the result in *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#PF(fault-code)	If a page fault occurs.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#PF(fault-code)	If a page fault occurs.

Numeric Exceptions

None.

PSHUFLW—Shuffle Packed Low Words

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
F2 0F 70 /r ib	PSHUFLW *xmm1*, *xmm2/m128*, *imm8*	Valid	Valid	Shuffle the low words in *xmm2/m128* based on the encoding in *imm8* and store the result in *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#PF(fault-code)	If a page fault occurs.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#PF(fault-code)	If a page fault occurs.

Numeric Exceptions

None.

PSHUFW—Shuffle Packed Words

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 70 /r ib	PSHUFW *mm1, mm2/ m64, imm8*	Valid	Valid	Shuffle the words in *mm2/m64* based on the encoding in *imm8* and store the result in in *mm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
#NM	If TS in CR0 is set.
#MF	If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
#NM	If TS in CR0 is set.
#MF	If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#UD	If EM in CR0 is set.
#NM	If TS in CR0 is set.
#MF	If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Numeric Exceptions

None.

PSLLDQ—Shift Double Quadword Left Logical

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
66 0F 73 /7 ib	PSLLDQ *xmm1*, *imm8*	Valid	Valid	Shift *xmm1* left by *imm8* bytes while shifting in 0s.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.

Real-Address Mode Exceptions

Same exceptions as in Protected Mode

Virtual-8086 Mode Exceptions

Same exceptions as in Protected Mode

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

Same as for protected mode exceptions.

Numeric Exceptions

None.

PSLLW/PSLLD/PSLLQ—Shift Packed Data Left Logical

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F F1 /r	PSLLW *mm, mm/m64*	Valid	Valid	Shift words in *mm* left *mm/m64* while shifting in 0s.
66 0F F1 /r	PSLLW *xmm1*, *xmm2/m128*	Valid	Valid	Shift words in *xmm1* left by *xmm2/m128* while shifting in 0s.
0F 71 /6 ib	PSLLW *xmm1*, *imm8*	Valid	Valid	Shift words in *mm* left by *imm8* while shifting in 0s.
66 0F 71 /6 ib	PSLLW *xmm1*, *imm8*	Valid	Valid	Shift words in *xmm1* left by *imm8* while shifting in 0s.
0F F2 /r	PSLLD *mm, mm/m64*	Valid	Valid	Shift doublewords in *mm* left by *mm/m64* while shifting in 0s.
66 0F F2 /r	PSLLD *xmm1*, *xmm2/m128*	Valid	Valid	Shift doublewords in *xmm1* left by *xmm2/ m128* while shifting in 0s.
0F 72 /6 ib	PSLLD *mm, imm8*	Valid	Valid	Shift doublewords in *mm* left by *imm8* while shifting in 0s.
66 0F 72 /6 ib	PSLLD *xmm1*, *imm8*	Valid	Valid	Shift doublewords in *xmm1* left by *imm8* while shifting in 0s.
0F F3 /r	PSLLQ *mm, mm/m64*	Valid	Valid	Shift quadword in *mm* left by *mm/m64* while shifting in 0s.
66 0F F3 /r	PSLLQ *xmm1*, *xmm2/m128*	Valid	Valid	Shift quadwords in *xmm1* left by *xmm2/m128* while shifting in 0s.
0F 73 /6 ib	PSLLQ *mm, imm8*	Valid	Valid	Shift quadword in *mm* left by *imm8* while shifting in 0s.
66 0F 73 /6 ib	PSLLQ *xmm1*, *imm8*	Valid	Valid	Shift quadwords in *xmm1* left by *imm8* while shifting in 0s.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Numeric Exceptions

None.

PSRAW/PSRAD—Shift Packed Data Right Arithmetic

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F E1 /r	PSRAW *mm, mm/m64*	Valid	Valid	Shift words in *mm* right by *mm/m64* while shifting in sign bits.
66 0F E1 /r	PSRAW *xmm1*, *xmm2/m128*	Valid	Valid	Shift words in *xmm1* right by *xmm2/m128* while shifting in sign bits.
0F 71 /4 ib	PSRAW *mm, imm8*	Valid	Valid	Shift words in *mm* right by *imm8* while shifting in sign bits
66 0F 71 /4 ib	PSRAW *xmm1*, imm8	Valid	Valid	Shift words in *xmm1* right by imm8 while shifting in sign bits
0F E2 /r	PSRAD *mm, mm/m64*	Valid	Valid	Shift doublewords in *mm* right by *mm/m64* while shifting in sign bits.
66 0F E2 /r	PSRAD *xmm1*, *xmm2/m128*	Valid	Valid	Shift doubleword in *xmm1* right by *xmm2 / m128* while shifting in sign bits.
0F 72 /4 ib	PSRAD *mm, imm8*	Valid	Valid	Shift doublewords in *mm* right by *imm8* while shifting in sign bits.
66 0F 72 /4 ib	PSRAD *xmm1*, imm8	Valid	Valid	Shift doublewords in *xmm1* right by *imm8* while shifting in sign bits.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Numeric Exceptions

None.

PSRLDQ—Shift Double Quadword Right Logical

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
66 0F 73 /3 ib	PSRLDQ *xmm1*, *imm8*	Valid	Valid	Shift *xmm1* right by *imm8* while shifting in 0s.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.

Real-Address Mode Exceptions

Same exceptions as in Protected Mode.

Virtual-8086 Mode Exceptions

Same exceptions as in Protected Mode.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

Same as for protected mode exceptions.

Numeric Exceptions

None.

PSRLW/PSRLD/PSRLQ—Shift Packed Data Right Logical

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F D1 /r	PSRLW *mm, mm/m64*	Valid	Valid	Shift words in *mm* right by amount specified in *mm/m64* while shifting in 0s.
66 0F D1 /r	PSRLW *xmm1*, *xmm2/m128*	Valid	Valid	Shift words in *xmm1* right by amount specified in *xmm2/m128* while shifting in 0s.
0F 71 /2 ib	PSRLW *mm, imm8*	Valid	Valid	Shift words in *mm* right by *imm8* while shifting in 0s.
66 0F 71 /2 ib	PSRLW *xmm1*, *imm8*	Valid	Valid	Shift words in *xmm1* right by *imm8* while shifting in 0s.
0F D2 /r	PSRLD *mm, mm/m64*	Valid	Valid	Shift doublewords in *mm* right by amount specified in *mm/m64* while shifting in 0s.
66 0F D2 /r	PSRLD *xmm1*, *xmm2/m128*	Valid	Valid	Shift doublewords in *xmm1* right by amount specified in *xmm2 /m128* while shifting in 0s.
0F 72 /2 ib	PSRLD *mm, imm8*	Valid	Valid	Shift doublewords in *mm* right by *imm8* while shifting in 0s.
66 0F 72 /2 ib	PSRLD *xmm1*, imm8	Valid	Valid	Shift doublewords in *xmm1* right by imm8 while shifting in 0s.
0F D3 /r	PSRLQ *mm, mm/m64*	Valid	Valid	Shift *mm* right by amount specified in *mm/m64* while shifting in 0s.
66 0F D3 /r	PSRLQ *xmm1*, *xmm2/m128*	Valid	Valid	Shift quadwords in *xmm1* right by amount specified in *xmm2/m128* while shifting in 0s.
0F 73 /2 ib	PSRLQ *mm, imm8*	Valid	Valid	Shift *mm* right by *imm8* while shifting in 0s.
66 0F 73 /2 ib	PSRLQ *xmm1*, imm8	Valid	Valid	Shift quadwords in *xmm1* right by *imm8* while shifting in 0s.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.

64-Bit Mode Exceptions

#GP(0)	If the memory address is in a non-canonical form.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Numeric Exceptions

None.

PSUBB/PSUBW/PSUBD—Subtract Packed Integers

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F F8 /r	PSUBB *mm, mm/m64*	Valid	Valid	Subtract packed byte integers in *mm/m64* from packed byte integers in *mm*.
66 0F F8 /r	PSUBB *xmm1*, *xmm2/ m128*	Valid	Valid	Subtract packed byte integers in *xmm2/m128* from packed byte integers in *xmm1*.
0F F9 /r	PSUBW *mm, mm/m64*	Valid	Valid	Subtract packed word integers in *mm/m64* from packed word integers in *mm*.
66 0F F9 /r	PSUBW *xmm1*, *xmm2/ m128*	Valid	Valid	Subtract packed word integers in *xmm2/m128* from packed word integers in *xmm1*.
0F FA /r	PSUBD *mm, mm/m64*	Valid	Valid	Subtract packed doubleword integers in *mm/m64* from packed doubleword integers in *mm*.
66 0F FA /r	PSUBD *xmm1*, *xmm2/ m128*	Valid	Valid	Subtract packed doubleword integers in *xmm2/ mem128* from packed doubleword integers in *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Numeric Exceptions

None.

PSUBQ—Subtract Packed Quadword Integers

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F FB /r	PSUBQ *mm1*, *mm2/m64*	Valid	Valid	Subtract quadword integer in *mm1* from *mm2 /m64*.
66 0F FB /r	PSUBQ *xmm1*, *xmm2/m128*	Valid	Valid	Subtract packed quadword integers in *xmm1* from *xmm2 /m128*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Numeric Exceptions

None.

PSUBSB/PSUBSW—Subtract Packed Signed Integers with Signed Saturation

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F E8 /r	PSUBSB *mm, mm/m64*	Valid	Valid	Subtract signed packed bytes in *mm/m64* from signed packed bytes in *mm* and saturate results.
66 0F E8 /r	PSUBSB *xmm1*, *xmm2/m128*	Valid	Valid	Subtract packed signed byte integers in *xmm2/m128* from packed signed byte integers in xmm1 and saturate results.
0F E9 /r	PSUBSW *mm, mm/m64*	Valid	Valid	Subtract signed packed words in *mm/m64* from signed packed words in *mm* and saturate results.
66 0F E9 /r	PSUBSW *xmm1*, *xmm2/m128*	Valid	Valid	Subtract packed signed word integers in *xmm2/m128* from packed signed word integers in xmm1 and saturate results.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#GP(0)	If the memory address is in a non-canonical form.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Numeric Exceptions

None.

PSUBUSB/PSUBUSW—Subtract Packed Unsigned Integers with Unsigned Saturation

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F D8 /r	PSUBUSB *mm, mm/m64*	Valid	Valid	Subtract unsigned packed bytes in *mm/m64* from unsigned packed bytes in *mm* and saturate result.
66 0F D8 /r	PSUBUSB *xmm1*, *xmm2/ m128*	Valid	Valid	Subtract packed unsigned byte integers in *xmm2/ m128* from packed unsigned byte integers in xmm1 and saturate result.
0F D9 /r	PSUBUSW *mm, mm/m64*	Valid	Valid	Subtract unsigned packed words in *mm/m64* from unsigned packed words in *mm* and saturate result.
66 0F D9 /r	PSUBUSW *xmm1*, *xmm2/ m128*	Valid	Valid	Subtract packed unsigned word integers in *xmm2/ m128* from packed unsigned word integers in xmm1 and saturate result.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Numeric Exceptions

None.

PUNPCKHBW/PUNPCKHWD/PUNPCKHDQ/PUNPCKHQDQ—Unpack High Data

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 68 /r	PUNPCKHBW *mm, mm/m64*	Valid	Valid	Unpack and interleave high-order bytes from *mm* and *mm/m64* into *mm*.
66 0F 68 /r	PUNPCKHBW *xmm1*, *xmm2/m128*	Valid	Valid	Unpack and interleave high-order bytes from *xmm1* and *xmm2/m128* into *xmm1*.
0F 69 /r	PUNPCKHWD *mm, mm/m64*	Valid	Valid	Unpack and interleave high-order words from *mm* and *mm/m64* into *mm*.
66 0F 69 /r	PUNPCKHWD *xmm1*, *xmm2/m128*	Valid	Valid	Unpack and interleave high-order words from *xmm1* and *xmm2/m128* into *xmm1*.
0F 6A /r	PUNPCKHDQ *mm, mm/m64*	Valid	Valid	Unpack and interleave high-order doublewords from *mm* and *mm/m64* into *mm*.
66 0F 6A /r	PUNPCKHDQ *xmm1*, *xmm2/m128*	Valid	Valid	Unpack and interleave high-order doublewords from *xmm1* and *xmm2/m128* into *xmm1*.
66 0F 6D /r	PUNPCKHQDQ *xmm1*, *xmm2/m128*	Valid	Valid	Unpack and interleave high-order quadwords from *xmm1* and *xmm2/m128* into *xmm1*

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit version only) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit version only) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#GP(0)	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	(128-bit version only) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Numeric Exceptions

None.

PUNPCKLBW/PUNPCKLWD/PUNPCKLDQ/PUNPCKLQDQ— Unpack Low Data

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 60 /r	PUNPCKLBW *mm, mm/m32*	Valid	Valid	Interleave low-order bytes from *mm* and *mm/m32* into *mm*.
66 0F 60 /r	PUNPCKLBW *xmm1*, *xmm2/m128*	Valid	Valid	Interleave low-order bytes from *xmm1* and *xmm2/m128* into *xmm1*.
0F 61 /r	PUNPCKLWD *mm, mm/m32*	Valid	Valid	Interleave low-order words from *mm* and *mm/m32* into *mm*.
66 0F 61 /r	PUNPCKLWD *xmm1*, *xmm2/m128*	Valid	Valid	Interleave low-order words from *xmm1* and *xmm2/m128* into *xmm1*.
0F 62 /r	PUNPCKLDQ *mm, mm/m32*	Valid	Valid	Interleave low-order doublewords from *mm* and *mm/m32* into *mm*.
66 0F 62 /r	PUNPCKLDQ *xmm1*, *xmm2/m128*	Valid	Valid	Interleave low-order doublewords from *xmm1* and *xmm2/m128* into xmm1.
66 0F 6C /r	PUNPCKLQDQ *xmm1*, *xmm2/m128*	Valid	Valid	Interleave low-order quadword from *xmm1* and *xmm2/m128* into *xmm1* register

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit version only) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit version only) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
#NM	(64-bit operations only) If TS in CR0 is set.
#MF	(64-bit operations only) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	(128-bit version only) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#UD	If EM in CR0 is set.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Numeric Exceptions

None.

PUSH—Push Word or Doubleword Onto the Stack

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
FF /6	PUSH *r/m16*	Valid	Valid	Push *r/m16*
FF /6	PUSH *r/m32*	N.E.	Valid	Push *r/m32*
FF /6	PUSH *r/m64*	Valid	N.E.	Push *r/m64.* Default operand size 64-bits.
50+*rw*	PUSH *r16*	Valid	Valid	Push *r16*
50+*rd*	PUSH *r32*	N.E.	Valid	Push *r32*
50+*rd*	PUSH *r64*	Valid	N.E.	Push *r64.* Default operand size 64-bits.
6A	PUSH *imm8*	Valid	Valid	Push *imm8*
68	PUSH *imm16*	Valid	Valid	Push *imm16*
68	PUSH *imm32*	N.E.	Valid	Push *imm32*
68	PUSH *imm64*	Valid	N.E.	Push zero-extended *imm32.* Default operand size 64-bits.
0E	PUSH CS	Inv.	Valid	Push CS
16	PUSH SS	Inv.	Valid	Push SS
1E	PUSH DS	Inv.	Valid	Push DS
06	PUSH ES	Inv.	Valid	Push ES
0F A0	PUSH FS	Valid	Valid	Push FS and decrement stack pointer by 16 bits.
0F A0	PUSH FS	N.E.	Valid	Push FS and decrement stack pointer by 32 bits.
0F A0	PUSH FS	Valid	N.E.	Push FS. Default operand size 64-bits. (66h override causes 16-bit operation)
0F A8	PUSH GS	Valid	Valid	Push GS and decrement stack pointer by 16 bits.
0F A8	PUSH GS	N.E.	Valid	Push GS and decrement stack pointer by 32 bits.
0F A8	PUSH GS	Valid	N.E.	Push GS, Default operand size 64-bits. (66h override causes 16-bit operation

Flags Affected

None.

IA-32e Mode Operation

See Table above.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the DS, ES, FS, or GS register is used to access memory and it contains a null segment selector.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS	If a memory operand effective address is outside the SS segment limit.
	If the new value of the SP or ESP register is outside the stack segment limit.

Virtual-8086 Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#GP(0)	If the memory address is in a non-canonical form.
#SS(U)	If the stack address is in a non-canonical form.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

PUSHA/PUSHAD—Push All General-Purpose Registers

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
60	PUSHA	N.E.	Valid	Push AX, CX, DX, BX, original SP, BP, SI, and DI
60	PUSHAD	N.E.	Valid	Push EAX, ECX, EDX, EBX, original ESP, EBP, ESI, and EDI

Flags Affected

None.

IA-32e Mode Operation

Invalid in 64-bit mode.

Protected Mode Exceptions

#SS(0)	If the starting or ending stack address is outside the stack segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If an unaligned memory reference is made while the current privilege level is 3 and alignment checking is enabled.

Real-Address Mode Exceptions

#GP	If the ESP or SP register contains 7, 9, 11, 13, or 15.

Virtual-8086 Mode Exceptions

#GP(0)	If the ESP or SP register contains 7, 9, 11, 13, or 15.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If an unaligned memory reference is made while alignment checking is enabled.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#

PUSHF/PUSHFD—Push EFLAGS Register onto the Stack

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
9C	PUSHF	Valid	Valid	Push lower 16 bits of EFLAGS
9C	PUSHFD	N.E.	Valid	Push EFLAGS
9C	PUSHFQ	Valid	N.E.	Push RFLAGS

Flags Affected

None.

IA-32e Mode Operation

Promoted to 64-bits.

Default operand size 64-bits. Pushes the 64-bit RFLAGS register.

Cannot encode 32-bit operand size.

Protected Mode Exceptions

#SS(0)	If the new value of the ESP register is outside the stack segment boundary.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If an unaligned memory reference is made while the current privilege level is 3 and alignment checking is enabled.

Real-Address Mode Exceptions

None.

Virtual-8086 Mode Exceptions

#GP(0)	If the I/O privilege level is less than 3.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If an unaligned memory reference is made while alignment checking is enabled.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#GP(0)	If the memory address is in a non-canonical form.
#SS(U)	If the stack address is in a non-canonical form.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If an unaligned memory reference is made while the current privilege level is 3 and alignment checking is enabled.

PXOR—Logical Exclusive OR

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F EF /r	PXOR *mm, mm/m64*	Valid	Valid	Bitwise XOR of *mm/m64* and *mm.*
66 0F EF /r	PXOR *xmm1*, *xmm2/m128*	Valid	Valid	Bitwise XOR of *xmm2/m128* and *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

Flags Affected

None.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside of the effective address space from 0 to FFFFH.
#UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	(128-bit operations only.) If memory operand is not aligned on a 16-byte boundary, regardless of segment.
UD	If EM in CR0 is set.
	(128-bit operations only.) If OSFXSR in CR4 is 0.
	(128-bit operations only.) If CPUID feature flag SSE2 is 0.
#NM	If TS in CR0 is set.
#MF	(64-bit operations only.) If there is a pending x87 FPU exception.
#PF(fault-code)	If a page fault occurs.
#AC(0)	(64-bit operations only.) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Numeric Exceptions

None.

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
D0 /2	RCL *r/m8*, 1	Valid	Valid	Rotate 9 bits (CF, *r/m8*) left once.
REX + D0 /2	RCL *r/m8**, 1	Valid	N.E.	Rotate 9 bits (CF, *r/m8*) left once.
D2 /2	RCL *r/m8*, CL	Valid	Valid	Rotate 9 bits (CF, *r/m8*) left CL times.
REX + D2 /2	RCL *r/m8**, CL	Valid	N.E.	Rotate 9 bits (CF, *r/m8*) left CL times.
C0 /2 *ib*	RCL *r/m8, imm8*	Valid	Valid	Rotate 9 bits (CF, *r/m8*) left *imm8* times.
REX + C0 /2 *ib*	RCL *r/m8*, imm8*	Valid	N.E.	Rotate 9 bits (CF, *r/m8*) left *imm8* times.
D1 /2	RCL *r/m16*, 1	Valid	Valid	Rotate 17 bits (CF, *r/m16*) left once
D3 /2	RCL *r/m16*, CL	Valid	Valid	Rotate 17 bits (CF, *r/m16*) left CL times
C1 /2 *ib*	RCL *r/m16, imm8*	Valid	Valid	Rotate 17 bits (CF, *r/m16*) left *imm8* times
D1 /2	RCL *r/m32*, 1	Valid	Valid	Rotate 33 bits (CF, *r/m32*) left once
REX.W + D1 /2	RCL *r/m64*, 1	Valid	N.E.	Rotate 65 bits (CF, *r/m64*) left once. Uses a 6 bit count.
D3 /2	RCL *r/m32*, CL	Valid	Valid	Rotate 33 bits (CF, *r/m32*) left CL times
REX.W + D3 /2	RCL *r/m64*, CL	Valid	N.E.	Rotate 65 bits (CF, *r/m64*) left CL times. Uses a 6 bit count.
C1 /2 *ib*	RCL *r/m32, imm8*	Valid	Valid	Rotate 33 bits (CF, *r/m32*) left *imm8* times
REX.W + C1 /2 *ib*	RCL *r/m64, imm8*	Valid	N.E.	Rotate 65 bits (CF, *r/m64*) left *imm8* times. Uses a 6 bit count.
D0 /3	RCR *r/m8*, 1	Valid	Valid	Rotate 9 bits (CF, *r/m8*) right once.
REX + D0 /3	RCR *r/m8**, 1	Valid	N.E.	Rotate 9 bits (CF, *r/m8*) right once.
D2 /3	RCR *r/m8*, CL	Valid	Valid	Rotate 9 bits (CF, *r/m8*) right CL times.
REX + D2 /3	RCR *r/m8**, CL	Valid	N.E.	Rotate 9 bits (CF, *r/m8*) right CL times.
C0 /3 *ib*	RCR *r/m8, imm8*	Valid	Valid	Rotate 9 bits (CF, *r/m8*) right *imm8* times.
REX + C0 /3 *ib*	RCR *r/m8*, imm8*	Valid	N.E.	Rotate 9 bits (CF, *r/m8*) right *imm8* times.
D1 /3	RCR *r/m16*, 1	Valid	Valid	Rotate 17 bits (CF, *r/m16*) right once
D3 /3	RCR *r/m16*, CL	Valid	Valid	Rotate 17 bits (CF, *r/m16*) right CL times
C1 /3 *ib*	RCR *r/m16, imm8*	Valid	Valid	Rotate 17 bits (CF, *r/m16*) right *imm8* times
D1 /3	RCR *r/m32*, 1	Valid	Valid	Rotate 33 bits (CF, *r/m32*) right once. Uses a 6 bit count.
REX.W + D1 /3	RCR *r/m64*, 1	Valid	N.E.	Rotate 65 bits (CF, *r/m64*) right once. Uses a 6 bit count.
D3 /3	RCR *r/m32*, CL	Valid	Valid	Rotate 33 bits (CF, *r/m32*) right CL times
REX.W + D3 /3	RCR *r/m64*, CL	Valid	N.E.	Rotate 65 bits (CF, *r/m64*) right CL times. Uses a 6 bit count.
C1 /3 *ib*	RCR *r/m32, imm8*	Valid	Valid	Rotate 33 bits (CF, *r/m32*) right *imm8* times
REX.W + C1 /3 *ib*	RCR *r/m64, imm8*	Valid	N.E.	Rotate 65 bits (CF, *r/m64*) right *imm8* times. Uses a 6 bit count.
D0 /0	ROL *r/m8*, 1	Valid	Valid	Rotate 8 bits *r/m8* left once
REX + D0 /0	ROL *r/m8**, 1	Valid	N.E.	Rotate 8 bits *r/m8* left once
D2 /0	ROL *r/m8*, CL	Valid	Valid	Rotate 8 bits *r/m8* left CL times
REX + D2 /0	ROL *r/m8**, CL	Valid	N.E.	Rotate 8 bits *r/m8* left CL times
C0 /0 *ib*	ROL *r/m8, imm8*	Valid	Valid	Rotate 8 bits *r/m8* left *imm8* times
REX + C0 /0 *ib*	ROL *r/m8*, imm8*	Valid	N.E.	Rotate 8 bits *r/m8* left *imm8* times
D1 /0	ROL *r/m16*, 1	Valid	Valid	Rotate 16 bits *r/m16* left once
D3 /0	ROL *r/m16*, CL	Valid	Valid	Rotate 16 bits *r/m16* left CL times
C1 /0 *ib*	ROL *r/m16, imm8*	Valid	Valid	Rotate 16 bits *r/m16* left *imm8* times
D1 /0	ROL *r/m32*, 1	Valid	Valid	Rotate 32 bits *r/m32* left once
REX.W + D1 /0	ROL *r/m64*, 1	Valid	N.E.	Rotate 64 bits *r/m64* left once. Uses a 6 bit count.
D3 /0	ROL *r/m32*, CL	Valid	Valid	Rotate 32 bits *r/m32* left CL times
REX.W + D3 /0	ROL *r/m64*, CL	Valid	N.E.	Rotate 64 bits *r/m64* left CL times. Uses a 6 bit count.

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
C1 /0 *ib*	ROL *r/m32, imm8*	Valid	Valid	Rotate 32 bits *r/m32* left *imm8* times
C1 /0 *ib*	ROL *r/m64, imm8*	Valid	N.E.	Rotate 64 bits *r/m64* left *imm8* times. Uses a 6 bit count.
D0 /1	ROR *r/m8*, 1	Valid	Valid	Rotate 8 bits *r/m8* right once
REX + D0 /1	ROR *r/m8**, 1	Valid	N.E.	Rotate 8 bits *r/m8* right once
D2 /1	ROR *r/m8*, CL	Valid	Valid	Rotate 8 bits *r/m8* right CL times
REX + D2 /1	ROR *r/m8**, CL	Valid	N.E.	Rotate 8 bits *r/m8* right CL times
C0 /1 *ib*	ROR *r/m8, imm8*	Valid	Valid	Rotate 8 bits *r/m16* right *imm8* times
REX + C0 /1 *ib*	ROR *r/m8*, imm8*	Valid	N.E.	Rotate 8 bits *r/m16* right *imm8* times
D1 /1	ROR *r/m16*, 1	Valid	Valid	Rotate 16 bits *r/m16* right once
D3 /1	ROR *r/m16*, CL	Valid	Valid	Rotate 16 bits *r/m16* right CL times
C1 /1 *ib*	ROR *r/m16, imm8*	Valid	Valid	Rotate 16 bits *r/m16* right *imm8* times
D1 /1	ROR *r/m32*, 1	Valid	Valid	Rotate 32 bits *r/m32* right once
REX.W + D1 /1	ROR *r/m64*, 1	Valid	N.E.	Rotate 64 bits *r/m64* right once. Uses a 6 bit count.
D3 /1	ROR *r/m32*, CL	Valid	Valid	Rotate 32 bits *r/m32* right CL times
REX.W + D3 /1	ROR *r/m64*, CL	Valid	N.E.	Rotate 64 bits *r/m64* right CL times. Uses a 6 bit count.
C1 /1 *ib*	ROR *r/m32, imm8*	Valid	Valid	Rotate 32 bits *r/m32* right *imm8* times
REX.W + C1 /1 *ib*	ROR *r/m64, imm8*	Valid	N.E.	Rotate 64 bits *r/m64* right *imm8* times. Uses a 6 bit count.

* In 64-bit mode, r/m8 can not be encoded to access the following byte registers if an REX prefix is used: AH, BH, CH, DH. Also refer to Section 1.4.2.2.

Flags Affected

The CF flag contains the value of the bit shifted into it. The OF flag is affected only for single-bit rotates (see "Description" above); it is undefined for multi-bit rotates. The SF, ZF, AF, and PF flags are not affected.

IA-32e Mode Operation

See Table above.

Protected Mode Exceptions

#GP(0)	If the source operand is located in a nonwritable segment.
	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the DS, ES, FS, or GS register contains a null segment selector.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS	If a memory operand effective address is outside the SS segment limit.

Virtual-8086 Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#GP(0)	If the source operand is located in a nonwritable segment.
	If the memory address is in a non-canonical form.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

RCPPS—Compute Reciprocals of Packed Single-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 53 /r	RCPPS *xmm1, xmm2/m128*	Valid	Valid	Computes the approximate reciprocals of the packed single-precision floating-point values in *xmm2/m128* and stores the results in *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

RCPSS—Compute Reciprocal of Scalar Single-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
F3 0F 53 /r	RCPSS *xmm1*, xmm2/m32	Valid	Valid	Computes the approximate reciprocal of the scalar single-precision floating-point value in *xmm2/m32* and stores the result in *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.
	For unaligned memory reference if the current privilege level is 3.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

RDMSR—Read from Model Specific Register

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 32	RDMSR	Valid	Valid	Load MSR specified by ECX into EDX:EAX
REX.W + 0F 32	RDMSR	Valid	N.E.	Load MSR specified by RCX into RDX:RAX

Flags Affected

None.

IA-32e Mode Operation

Same as legacy mode. RDX[31:0] contains MSR[63:32], RAX[31:0] contains MSR[31:0]

Protected Mode Exceptions

#GP(0) If the current privilege level is not 0.

If the value in ECX specifies a reserved or unimplemented MSR address.

Real-Address Mode Exceptions

#GP If the value in ECX specifies a reserved or unimplemented MSR address.

Virtual-8086 Mode Exceptions

#GP(0) The RDMSR instruction is not recognized in virtual-8086 mode.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#GP(0) If the current privilege level is not 0.

If the value in ECX or RCX specifies a reserved or unimplemented MSR address.

RDPMC—Read Performance-Monitoring Counters

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 33	RDPMC	Valid	Valid	Read performance-monitoring counter specified by ECX into EDX:EAX
REX.W + 0F 33	RDPMC	Valid	N.E.	Read performance-monitoring counter specified by RCX into RDX:RAX

Flags Affected

None.

IA-32e Mode Operation

Same as legacy mode. RDX[31:0] contains MSR[63:32], RAX[31:0] contains MSR[31:0]

Protected Mode Exceptions

#GP(0) If the current privilege level is not 0 and the PCE flag in the CR4 register is clear.

(P6 family processors and Pentium processors with MMX Technology) If the value in the ECX register is not 0 or 1.

(Pentium 4 processor) If the value in ECX[30:0] is not within the range of 0 through 17.

Real-Address Mode Exceptions

#GP (P6 family processors and Pentium processors with MMX Technology) If the value in the ECX register is not 0 or 1.

(Pentium 4 processor) If the value in ECX[30:0] is not within the range of 0 through 17.

Virtual-8086 Mode Exceptions

#GP(0) If the PCE flag in the CR4 register is clear.

(P6 family processors and Pentium processors with MMX Technology) If the value in the ECX register is not 0 or 1.

(Pentium 4 processor) If the value in ECX[30:0] is not within the range of 0 through 17.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#GP(0) If the current privilege level is not 0 and the PCE flag in the CR4 register is clear.

If the value in ECX[30:0] is not within the range of 0 through 17.

RDTSC—Read Time-Stamp Counter

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 31	RDTSC	Valid	Valid	Read time-stamp counter into EDX:EAX
REX.W + 0F 31	RDTSC	Valid	N.E.	Read time-stamp counter into RDX:RAX

Flags Affected

None.

IA-32e Mode Operation

Same as legacy mode. RDX[31:0] contains MSR[63:32], RAX[31:0] contains MSR[31:0]

Protected Mode Exceptions

#GP(0)	If the TSD flag in register CR4 is set and the CPL is greater than 0.

Real-Address Mode Exceptions

None.

Virtual-8086 Mode Exceptions

#GP(0)	If the TSD flag in register CR4 is set.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

Same as for protected mode exceptions.

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
F3 6C	REP INS *m8*, DX	Valid	Valid	Input (E)CX bytes from port DX into ES:[(E)DI]
REX + F3 6C	REP INS *m8*, DX	Valid	N.E.	Input RCX bytes from port DX into [RDI]
F3 6D	REP INS *m16*, DX	Valid	Valid	Input (E)CX words from port DX into ES:[(E)DI]
F3 6D	REP INS *m32*, DX	Valid	Valid	Input (E)CX doublewords from port DX into ES:[(E)DI]
REX.W + F3 6D	REP INS *r/m32*, DX	Valid	N.E.	Input RCX default size from port DX into [RDI]
F3 A4	REP MOVS *m8, m8*	Valid	Valid	Move (E)CX bytes from DS:[(E)SI] to ES:[(E)DI]
REX.W + F3 A4	REP MOVS *m8, m8*	Valid	N.E.	Move RCX bytes from [RSI] to [RDI]
F3 A5	REP MOVS *m16, m16*	Valid	Valid	Move (E)CX words from DS:[(E)SI] to ES:[(E)DI]
F3 A5	REP MOVS *m32, m32*	Valid	Valid	Move (E)CX doublewords from DS:[(E)SI] to ES:[(E)DI]
REX.W + F3 A5	REP MOVS *m32, m32*	Valid	N.E.	Move ECX quadwords from [ESI] to [EDI]
REX.W + F3 A5	REP MOVS *m64, m64*	Valid	N.E.	Move RCX quadwords from [RSI] to [RDI]
F3 6E	REP OUTS DX, *r/m8*	Valid	Valid	Output (E)CX bytes from DS:[(E)SI] to port DX
REX + F3 6E	REP OUTS DX, *r/m8**	Valid	N.E.	Output RCX bytes from [RSI] to port DX
F3 6F	REP OUTS DX, *r/m16*	Valid	Valid	Output (E)CX words from DS:[(E)SI] to port DX
F3 6F	REP OUTS DX, *r/m32*	Valid	Valid	Output (E)CX doublewords from DS:[(E)SI] to port DX
REX.W + F3 6F	REP OUTS DX, *r/m32*	Valid	N.E.	Output RCX default size from [RSI] to port DX
F3 AC	REP LODS AL	Valid	Valid	Load (E)CX bytes from DS:[(E)SI] to AL
REX.W + F3 AC	REP LODS AL	Valid	N.E.	Load RCX bytes from [RSI] to AL
F3 AD	REP LODS AX	Valid	Valid	Load (E)CX words from DS:[(E)SI] to AX
F3 AD	REP LODS EAX	Valid	Valid	Load (E)CX doublewords from DS:[(E)SI] to EAX
REX.W + F3 AD	REP LODS EAX	Valid	N.E.	Load ECX quadwords from [ESI] to EAX
REX.W + F3 AD	REP LODS RAX	Valid	N.E.	Load RCX quadwords from [RSI] to RAX
F3 AA	REP STOS *m8*	Valid	Valid	Fill (E)CX bytes at ES:[(E)DI] with AL
REX.W + F3 AA	REP STOS *m8*	Valid	N.E.	Fill RCX bytes at [RDI] with AL
F3 AB	REP STOS *m16*	Valid	Valid	Fill (E)CX words at ES:[(E)DI] with AX
F3 AB	REP STOS *m32*	Valid	Valid	Fill (E)CX doublewords at ES:[(E)DI] with EAX
REX.W + F3 AB	REP STOS *m32*	Valid	N.E.	Fill ECX doublewords at [EDI] with EAX
REX.W + F3 AB	REP STOS *m64*	Valid	N.E.	Fill RCX quadwords at [RDI] with RAX
F3 A6	REPE CMPS *m8, m8*	Valid	Valid	Find nonmatching bytes in ES:[(E)DI] and DS:[(E)SI]
REX.W + F3 A6	REPE CMPS *m8, m8*	Valid	N.E.	Find non-matching bytes in [RDI] and [RSI]
F3 A7	REPE CMPS *m16, m16*	Valid	Valid	Find nonmatching words in ES:[(E)DI] and DS:[(E)SI]
F3 A7	REPE CMPS *m32, m32*	Valid	Valid	Find nonmatching doublewords in ES:[(E)DI] and DS:[(E)SI]
REX.W + F3 A7	REPE CMPS *m32, m32*	Valid	N.E.	Find non-matching doublewords in [EDI] and [ESI]
REX.W + F3 A7	REPE CMPS *m64, m64*	Valid	N.E.	Find non-matching quadwords in [RDI] and [RSI]
F3 AE	REPE SCAS *m8*	Valid	Valid	Find non-AL byte starting at ES:[(E)DI]
REX.W + F3 AE	REPE SCAS *m8*	Valid	N.E.	Find non-AL byte starting at [RDI]
F3 AF	REPE SCAS *m16*	Valid	Valid	Find non-AX word starting at ES:[(E)DI]
F3 AF	REPE SCAS *m32*	Valid	Valid	Find non-EAX doubleword starting at ES:[(E)DI]
REX.W + F3 AF	REPE SCAS *m32*	Valid	N.E.	Find non-EAX doubleword starting at [EDI]
REX.W + F3 AF	REPE SCAS *m64*	Valid	N.E.	Find non-RAX quadword starting at [RDI]

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
F2 A6	REPNE CMPS *m8, m8*	Valid	Valid	Find matching bytes in ES:[(E)DI] and DS:[(E)SI]
REX.W + F2 A6	REPNE CMPS *m8, m8*	Valid	N.E.	Find matching bytes in [RDI] and [RSI]
F2 A7	REPNE CMPS *m16, m16*	Valid	Valid	Find matching words in ES:[(E)DI] and DS:[(E)SI]
F2 A7	REPNE CMPS *m32, m32*	Valid	Valid	Find matching doublewords in ES:[(E)DI] and DS:[(E)SI]
REX.W + F2 A7	REPNE CMPS *m32, m32*	Valid	N.E.	Find matching doublewords in [EDI] and [ESI]
REX.W + F2 A7	REPNE CMPS *m64, m64*	Valid	N.E.	Find matching doublewords in [RDI] and [RSI]
F2 AE	REPNE SCAS *m8*	Valid	Valid	Find AL, starting at ES:[(E)DI]
REX.W + F2 AE	REPNE SCAS *m8*	Valid	N.E.	Find AL, starting at [RDI]
F2 AF	REPNE SCAS *m16*	Valid	Valid	Find AX, starting at ES:[(E)DI]
F2 AF	REPNE SCAS *m32*	Valid	Valid	Find EAX, starting at ES:[(E)DI]
REX.W + F2 AF	REPNE SCAS *m32*	Valid	N.E.	Find EAX, starting at [EDI]
REX.W + F2 AF	REPNE SCAS *m64*	Valid	N.E.	Find RAX, starting at [RDI]

* In 64-bit mode, r/m8 can not be encoded to access the following byte registers if an REX prefix is used: AH, BH, CH, DH. Also refer to Section 1.4.2.2.

Flags Affected

None; however, the CMPS and SCAS instructions do set the status flags in the EFLAGS register.

IA-32e Mode Operation

Same as legacy mode.

Default operand size 32-bits

Exceptions (All Operating Modes)

None; however, exceptions can be generated by the instruction a repeat prefix is associated with.

64-Bit Mode Exceptions

#GP(0)	If the memory address is in a non-canonical form.

RET—Return from Procedure

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
C3	RET	Valid	Valid	Near return to calling procedure
CB	RET	Valid	Valid	Far return to calling procedure
C2 *iw*	RET *imm16*	Valid	Valid	Near return to calling procedure and pop *imm16* bytes from stack
CA *iw*	RET *imm16*	Valid	Valid	Far return to calling procedure and pop *imm16* bytes from stack

Flags Affected

None.

IA-32e Mode Operation

Same as legacy mode.

Default operand size 64-bits

Protected Mode Exceptions

#GP(0)	If the return code or stack segment selector null.
	If the return instruction pointer is not within the return code segment limit
#GP(selector)	If the RPL of the return code segment selector is less then the CPL.
	If the return code or stack segment selector index is not within its descriptor table limits.
	If the return code segment descriptor does not indicate a code segment.
	If the return code segment is non-conforming and the segment selector's DPL is not equal to the RPL of the code segment's segment selector
	If the return code segment is conforming and the segment selector's DPL greater than the RPL of the code segment's segment selector
	If the stack segment is not a writable data segment.
	If the stack segment selector RPL is not equal to the RPL of the return code segment selector.
	If the stack segment descriptor DPL is not equal to the RPL of the return code segment selector.
#SS(0)	If the top bytes of stack are not within stack limits.
	If the return stack segment is not present.
#NP(selector)	If the return code segment is not present.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If an unaligned memory access occurs when the CPL is 3 and alignment checking is enabled.

Real-Address Mode Exceptions

#GP	If the return instruction pointer is not within the return code segment limit
#SS	If the top bytes of stack are not within stack limits.

Virtual-8086 Mode Exceptions

#GP(0)	If the return instruction pointer is not within the return code segment limit
#SS(0)	If the top bytes of stack are not within stack limits.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If an unaligned memory access occurs when alignment checking is enabled.

Compatibility Mode Exceptions

Same as 64-bit mode exceptions.

64-Bit Mode Exceptions

#GP(0)	If the return instruction pointer is non-canonical.
	If the return instruction pointer is not within the return code segment limit.
	If the stack segment selector is null going back to compatibility mode.
	If the stack segment selector is null going back to CPL3 64-bit mode.
	If a null stack segment selector RPL is not equal to CPL going back to non-CPL3 64-bit mode.
	If the return code segment selector is null.
#GP(selector)	If the segment descriptor for a code segment does not indicate it is a code segment.
	If the proposed new code segment descriptor has both the D-bit and L-bit set.
	If the DPL for a nonconforming-code segment is not equal to the RPL of the code segment selector.
	If CPL is greater than the RPL of the code segment selector.
	If the DPL of a conforming-code segment is greater than the return code segment selector RPL.
	If a segment selector index is outside its descriptor table limits.
	If a segment descriptor memory address is non-canonical.
	If the stack segment is not a writable data segment.
	If the stack segment descriptor DPL is not equal to the RPL of the return code segment selector.
	If the stack segment selector RPL is not equal to the RPL of the return code segment selector.
#SS(0)	If an attempt to pop a value off the stack violates the SS limit.
	If an attempt to pop a value off the stack causes a non-canonical address to be referenced.
#NP(selector)	If the return code or stack segment is not present.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

ROL/ROR—Rotate

See entry for RCL/RCR/ROL/ROR—Rotate.

RSM—Resume from System Management Mode

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F AA	RSM	Valid	Valid	Resume operation of interrupted program

Flags Affected

All.

IA-32e Mode Operation

Same as legacy IA-32 architecture behavior. New SMM save state map will be used to restore the processor to the operating mode prior to the delivery of the SMI. See Appendix A.

Protected Mode Exceptions

#UD	If an attempt is made to execute this instruction when the processor is not in SMM.

Real-Address Mode Exceptions

#UD	If an attempt is made to execute this instruction when the processor is not in SMM.

Virtual-8086 Mode Exceptions

#UD	If an attempt is made to execute this instruction when the processor is not in SMM.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

Same as for protected mode exceptions.

RSQRTPS—Compute Reciprocals of Square Roots of Packed Single-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 52 /r	RSQRTPS *xmm1*, *xmm2/m128*	Valid	Valid	Computes the approximate reciprocals of the square roots of the packed single-precision floating-point values in *xmm2/m128* and stores the results in *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

RSQRTSS—Compute Reciprocal of Square Root of Scalar Single-Precision Floating-Point Value

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
F3 0F 52 /r	RSQRTSS *xmm1*, *xmm2/m32*	Valid	Valid	Computes the approximate reciprocal of the square root of the low single-precision floating-point value in *xmm2/m32* and stores the results in *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

SAHF—Store AH into Flags

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
9E	SAHF	Inv.	Valid	Loads SF, ZF, AF, PF, and CF from AH into EFLAGS register

Flags Affected

The SF, ZF, AF, PF, and CF flags are loaded with values from the AH register. Bits 1, 3, and 5 of the EFLAGS register are unaffected, with the values remaining 1, 0, and 0, respectively.

IA-32e Mode Operation

Invalid in 64-bit mode.

Protected Mode Exceptions

None

Real-Address Mode Exceptions

None

Virtual-8086 Mode Exceptions

None

Compatibility Mode Exceptions

None

64-Bit Mode Exceptions

#UD	If in 64-bit mode.

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
D0 /4	SAL *r/m8*,1	Valid	Valid	Multiply *r/m8* by 2, once.
REX + D0 /4	SAL *r/m8***,1	Valid	N.E.	Multiply *r/m8* by 2, once.
D2 /4	SAL *r/m8*,CL	Valid	Valid	Multiply *r/m8* by 2, CL times
REX + D2 /4	SAL *r/m8***,CL	Valid	N.E.	Multiply *r/m8* by 2, CL times
C0 /4 *ib*	SAL *r/m8,imm8*	Valid	Valid	Multiply *r/m8* by 2, *imm8* times
REX + C0 /4 *ib*	SAL *r/m8**,*imm8*	Valid	N.E.	Multiply *r/m8* by 2, *imm8* times
D1 /4	SAL *r/m16*,1	Valid	Valid	Multiply *r/m16* by 2, once
D3 /4	SAL *r/m16*,CL	Valid	Valid	Multiply *r/m16* by 2, CL times
C1 /4 *ib*	SAL *r/m16,imm8*	Valid	Valid	Multiply *r/m16* by 2, *imm8* times
D1 /4	SAL *r/m32*,1	Valid	Valid	Multiply *r/m32* by 2, once
REX.W + D1 /4	SAL *r/m64*,1	Valid	N.E.	Multiply *r/m64* by 2, once
D3 /4	SAL *r/m32*,CL	Valid	Valid	Multiply *r/m32* by 2, CL times
REX.W + D3 /4	SAL *r/m64*,CL	Valid	N.E.	Multiply *r/m64* by 2, CL times
C1 /4 *ib*	SAL *r/m32,imm8*	Valid	Valid	Multiply *r/m32* by 2, *imm8* times
REX.W + C1 /4 *ib*	SAL *r/m64, imm8*	Valid	N.E.	Multiply *r/m64* by 2, *imm8* times
D0 /7	SAR *r/m8*,1	Valid	Valid	Signed divide* *r/m8* by 2, once
REX + D0 /7	SAR *r/m8***,1	Valid	N.E.	Signed divide* *r/m8* by 2, once
D2 /7	SAR *r/m8*,CL	Valid	Valid	Signed divide* *r/m8* by 2, CL times
REX + D2 /7	SAR *r/m8***,CL	Valid	N.E.	Signed divide* *r/m8* by 2, CL times
C0 /7 *ib*	SAR *r/m8,imm8*	Valid	Valid	Signed divide* *r/m8* by 2, *imm8* times
REX + C0 /7 *ib*	SAR *r/m8**,*imm8*	Valid	N.E.	Signed divide* *r/m8* by 2, *imm8* times
D1 /7	SAR *r/m16*,1	Valid	Valid	Signed divide* *r/m16* by 2, once
D3 /7	SAR *r/m16*,CL	Valid	Valid	Signed divide* *r/m16* by 2, CL times
C1 /7 *ib*	SAR *r/m16,imm8*	Valid	Valid	Signed divide* *r/m16* by 2, *imm8* times
D1 /7	SAR *r/m32*,1	Valid	Valid	Signed divide* *r/m32* by 2, once
REX.W + D1 /7	SAR *r/m64*,1	Valid	N.E.	Signed divide* *r/m64* by 2, once
D3 /7	SAR *r/m32*,CL	Valid	Valid	Signed divide* *r/m32* by 2, CL times
REX.W + D3 /7	SAR *r/m64*, CL	Valid	N.E.	Signed divide* *r/m64* by 2, CL times
C1 /7 *ib*	SAR *r/m32,imm8*	Valid	Valid	Signed divide* *r/m32* by 2, *imm8* times
REX.W + C1 /7 *ib*	SAR *r/m64,imm8*	Valid	N.E.	Signed divide* *r/m64* by 2, *imm8* times
D0 /4	SHL *r/m8*,1	Valid	Valid	Multiply *r/m8* by 2, once
REX + D0 /4	SHL *r/m8***,1	Valid	N.E.	Multiply *r/m8* by 2, once
D2 /4	SHL *r/m8*,CL	Valid	Valid	Multiply *r/m8* by 2, CL times
REX + D2 /4	SHL *r/m8***,CL	Valid	N.E.	Multiply *r/m8* by 2, CL times
C0 /4 *ib*	SHL *r/m8,imm8*	Valid	Valid	Multiply *r/m8* by 2, *imm8* times
REX + C0 /4 *ib*	SHL *r/m8**,*imm8*	Valid	N.E.	Multiply *r/m8* by 2, *imm8* times
D1 /4	SHL *r/m16*,1	Valid	Valid	Multiply *r/m16* by 2, once
D3 /4	SHL *r/m16*,CL	Valid	Valid	Multiply *r/m16* by 2, CL times
C1 /4 *ib*	SHL *r/m16,imm8*	Valid	Valid	Multiply *r/m16* by 2, *imm8* times
D1 /4	SHL *r/m32*,1	Valid	Valid	Multiply *r/m32* by 2, once
REX.W + D1 /4	SHL *r/m64*,1	Valid	N.E.	Multiply *r/m64* by 2, once
D3 /4	SHL *r/m32*,CL	Valid	Valid	Multiply *r/m32* by 2, CL times
REX.W + D3 /4	SHL *r/m64*,CL	Valid	N.E.	Multiply *r/m64* by 2, CL times
C1 /4 *ib*	SHL *r/m32,imm8*	Valid	Valid	Multiply *r/m32* by 2, *imm8* times
REX.W + C1 /4 *ib*	SHL *r/m64,imm8*	Valid	N.E.	Multiply *r/m64* by 2, *imm8* times
D0 /5	SHR *r/m8*,1	Valid	Valid	Unsigned divide *r/m8* by 2, once
REX + D0 /5	SHR *r/m8***,1	Valid	N.E.	Unsigned divide *r/m8* by 2, once
D2 /5	SHR *r/m8*,CL	Valid	Valid	Unsigned divide *r/m8* by 2, CL times
REX + D2 /5	SHR *r/m8***,CL	Valid	N.E.	Unsigned divide *r/m8* by 2, CL times

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
C0 /5 *ib*	SHR *r/m8,imm8*	Valid	Valid	Unsigned divide *r/m8* by 2, *imm8* times
REX + C0 /5 *ib*	SHR *r/m8**,imm8*	Valid	N.E.	Unsigned divide *r/m8* by 2, *imm8* times
D1 /5	SHR *r/m16*,1	Valid	Valid	Unsigned divide *r/m16* by 2, once
D3 /5	SHR *r/m16*,CL	Valid	Valid	Unsigned divide *r/m16* by 2, CL times
C1 /5 *ib*	SHR *r/m16,imm8*	Valid	Valid	Unsigned divide *r/m16* by 2, *imm8* times
D1 /5	SHR *r/m32*,1	Valid	Valid	Unsigned divide *r/m32* by 2, once
REX.W + D1 /5	SHR *r/m64*,1	Valid	N.E.	Unsigned divide *r/m64* by 2, once
D3 /5	SHR *r/m32*,CL	Valid	Valid	Unsigned divide *r/m32* by 2, CL times
REX.W + D3 /5	SHR *r/m64*,CL	Valid	N.E.	Unsigned divide *r/m64* by 2, CL times
C1 /5 *ib*	SHR *r/m32,imm8*	Valid	Valid	Unsigned divide *r/m32* by 2, *imm8* times
REX.W + C1 /5 *ib*	SHR *r/m64, imm8*	Valid	N.E.	Unsigned divide *r/m64* by 2, *imm8* times

* Not the same form of division as IDIV; rounding is toward negative infinity.

** In 64-bit mode, r/m8 can not be encoded to access the following byte registers if an REX prefix is used: AH, BH, CH, DH. Also refer to Section 1.4.2.2.

Flags Affected

The CF flag contains the value of the last bit shifted out of the destination operand; it is undefined for SHL and SHR instructions where the count is greater than or equal to the size (in bits) of the destination operand. The OF flag is affected only for 1-bit shifts (see "Description" above); otherwise, it is undefined. The SF, ZF, and PF flags are set according to the result. If the count is 0, the flags are not affected. For a non-zero count, the AF flag is undefined.

IA-32e Mode Operation

Same table above.

Protected Mode Exceptions

#GP(0)	If the destination is located in a nonwritable segment.
	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the DS, ES, FS, or GS register contains a null segment selector.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS	If a memory operand effective address is outside the SS segment limit.

Virtual-8086 Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

SBB—Integer Subtraction with Borrow

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
1C *ib*	SBB AL,*imm8*	Valid	Valid	Subtract with borrow *imm8* from AL
1D *iw*	SBB AX,*imm16*	Valid	Valid	Subtract with borrow *imm16* from AX
1D *id*	SBB EAX,*imm32*	Valid	Valid	Subtract with borrow *imm32* from EAX
REX.W + 1D *id*	SBB RAX,*imm32*	Valid	N.E.	Subtract with borrow sign-extended *imm32 to 64-bits* from RAX
80 /3 *ib*	SBB *r/m8,imm8*	Valid	Valid	Subtract with borrow *imm8* from *r/m8*
REX + 80 /3 *ib*	SBB *r/m8*,imm8*	Valid	N.E.	Subtract with borrow *imm8* from *r/m8*
81 /3 *iw*	SBB *r/m16,imm16*	Valid	Valid	Subtract with borrow *imm16* from *r/m16*
81 /3 *id*	SBB *r/m32,imm32*	Valid	Valid	Subtract with borrow *imm32* from *r/m32*
REX.W + 81 /3 *id*	SBB *r/m64,imm32*	Valid	N.E.	Subtract with borrow sign-extended *imm32 to 64-bits* from *r/m64*
83 /3 *ib*	SBB *r/m16,imm8*	Valid	Valid	Subtract with borrow sign-extended *imm8* from *r/m16*
83 /3 *ib*	SBB *r/m32,imm8*	Valid	Valid	Subtract with borrow sign-extended *imm8* from *r/m32*
REX.W + 83 /3 *ib*	SBB *r/m64,imm8*	Valid	N.E.	Subtract with borrow sign-extended *imm8* from *r/m64*
18 /*r*	SBB *r/m8,r8*	Valid	Valid	Subtract with borrow *r8* from *r/m8*
REX + 18 /*r*	SBB *r/m8*,r8*	Valid	N.E.	Subtract with borrow *r8* from *r/m8*
19 /*r*	SBB *r/m16,r16*	Valid	Valid	Subtract with borrow *r16* from *r/m16*
19 /*r*	SBB *r/m32,r32*	Valid	Valid	Subtract with borrow *r32* from *r/m32*
REX.W + 19 /*r*	SBB *r/m64,r64*	Valid	N.E.	Subtract with borrow *r64* from *r/m64*
1A /*r*	SBB *r8,r/m8*	Valid	Valid	Subtract with borrow *r/m8* from *r8*
REX + 1A /*r*	SBB *r8*,r/m8**	Valid	N.E.	Subtract with borrow *r/m8* from *r8*
1B /*r*	SBB *r16,r/m16*	Valid	Valid	Subtract with borrow *r/m16* from *r16*
1B /*r*	SBB *r32,r/m32*	Valid	Valid	Subtract with borrow *r/m32* from *r32*
REX.W + 1B /*r*	SBB *r64,r/m64*	Valid	N.E.	Subtract with borrow *r/m64* from *r64*

* In 64-bit mode, r/m8 can not be encoded to access the following byte registers if an REX prefix is used: AH, BH, CH, DH. Also refer to Section 1.4.2.2.

Flags Affected

The OF, SF, ZF, AF, PF, and CF flags are set according to the result.

IA-32e Mode Operation

Same table above.

Protected Mode Exceptions

#GP(0)	If the destination is located in a nonwritable segment.
	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the DS, ES, FS, or GS register contains a null segment selector.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS	If a memory operand effective address is outside the SS segment limit.

Virtual-8086 Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
AE	SCAS m8	Valid	Valid	Compare AL with byte at ES:(E)DI and set status flags
REX.W + AE	SCAS m8	Valid	N.E.	Compare AL with byte at RDI and set status flags
AF	SCAS m16	Valid	Valid	Compare AX with word at ES:(E)DI and set status flags
AF	SCAS m32	Valid	Valid	Compare EAX with doubleword at ES(E)DI and set status flags
REX.W + AF	SCAS m64	Valid	N.E.	Compare RAX with quadword at RDI and set status flags
AE	SCASB	Valid	Valid	Compare AL with byte at ES:(E)DI and set status flags
REX.W + AE	SCASB	Valid	N.E.	Compare AL with byte at RDI and set status flags
AF	SCASW	Valid	Valid	Compare AX with word at ES:(E)DI and set status flags
AF	SCASD	Valid	Valid	Compare EAX with doubleword at ES:(E)DI and set status flags
REX.W + AF	SCASQ	Valid	N.E.	Compare RAX with quadword at RDI and set status flags

Flags Affected

The OF, SF, ZF, AF, PF, and CF flags are set according to the temporary result of the comparison.

IA-32e Mode Operation

See table above.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the limit of the ES segment.
	If the ES register contains a null segment selector.
	If an illegal memory operand effective address in the ES segment is given.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS	If a memory operand effective address is outside the SS segment limit.

Virtual-8086 Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 97	SETA *r/m8*	Valid	Valid	Set byte if above (CF=0 and ZF=0)
REX + 0F 97	SETA *r/m8**	Valid	N.E.	Set byte if above (CF=0 and ZF=0)
0F 93	SETAE *r/m8*	Valid	Valid	Set byte if above or equal (CF=0)
REX + 0F 93	SETAE *r/m8**	Valid	N.E.	Set byte if above or equal (CF=0)
0F 92	SETB *r/m8*	Valid	Valid	Set byte if below (CF=1)
REX + 0F 92	SETB *r/m8**	Valid	N.E.	Set byte if below (CF=1)
0F 96	SETBE *r/m8*	Valid	Valid	Set byte if below or equal (CF=1 or ZF=1)
REX + 0F 96	SETBE *r/m8**	Valid	N.E.	Set byte if below or equal (CF=1 or ZF=1)
0F 92	SETC *r/m8*	Valid	Valid	Set byte if carry (CF=1)
REX + 0F 92	SETC *r/m8**	Valid	N.E.	Set byte if carry (CF=1)
0F 94	SETE *r/m8*	Valid	Valid	Set byte if equal (ZF=1)
REX + 0F 94	SETE *r/m8**	Valid	N.E.	Set byte if equal (ZF=1)
0F 9F	SETG *r/m8*	Valid	Valid	Set byte if greater (ZF=0 and SF=OF)
REX + 0F 9F	SETG *r/m8**	Valid	N.E.	Set byte if greater (ZF=0 and SF=OF)
0F 9D	SETGE *r/m8*	Valid	Valid	Set byte if greater or equal (SF=OF)
REX + 0F 9D	SETGE *r/m8**	Valid	N.E.	Set byte if greater or equal (SF=OF)
0F 9C	SETL *r/m8*	Valid	Valid	Set byte if less (SF<>OF)
REX + 0F 9C	SETL *r/m8**	Valid	N.E.	Set byte if less (SF<>OF)
0F 9E	SETLE *r/m8*	Valid	Valid	Set byte if less or equal (ZF=1 or SF<>OF)
REX + 0F 9E	SETLE *r/m8**	Valid	N.E.	Set byte if less or equal (ZF=1 or SF<>OF)
0F 96	SETNA *r/m8*	Valid	Valid	Set byte if not above (CF=1 or ZF=1)
REX + 0F 96	SETNA *r/m8**	Valid	N.E.	Set byte if not above (CF=1 or ZF=1)
0F 92	SETNAE *r/m8*	Valid	Valid	Set byte if not above or equal (CF=1)
REX + 0F 92	SETNAE *r/m8**	Valid	N.E.	Set byte if not above or equal (CF=1)
0F 93	SETNB *r/m8*	Valid	Valid	Set byte if not below (CF=0)
REX + 0F 93	SETNB *r/m8**	Valid	N.E.	Set byte if not below (CF=0)
0F 97	SETNBE *r/m8*	Valid	Valid	Set byte if not below or equal (CF=0 and ZF=0)
REX + 0F 97	SETNBE *r/m8**	Valid	N.E.	Set byte if not below or equal (CF=0 and ZF=0)
0F 93	SETNC *r/m8*	Valid	Valid	Set byte if not carry (CF=0)
REX + 0F 93	SETNC *r/m8**	Valid	N.E.	Set byte if not carry (CF=0)
0F 95	SETNE *r/m8*	Valid	Valid	Set byte if not equal (ZF=0)
REX + 0F 95	SETNE *r/m8**	Valid	N.E.	Set byte if not equal (ZF=0)
0F 9E	SETNG *r/m8*	Valid	Valid	Set byte if not greater (ZF=1 or SF<>OF)
REX + 0F 9E	SETNG *r/m8**	Valid	N.E.	Set byte if not greater (ZF=1 or SF<>OF)
0F 9C	SETNGE *r/m8*	Valid	Valid	Set byte if not greater or equal (SF<>OF)
REX + 0F 9C	SETNGE *r/m8**	Valid	N.E.	Set byte if not greater or equal (SF<>OF)
0F 9D	SETNL *r/m8*	Valid	Valid	Set byte if not less (SF=OF)
REX + 0F 9D	SETNL *r/m8**	Valid	N.E.	Set byte if not less (SF=OF)
0F 9F	SETNLE *r/m8*	Valid	Valid	Set byte if not less or equal (ZF=0 and SF=OF)
REX + 0F 9F	SETNLE *r/m8**	Valid	N.E.	Set byte if not less or equal (ZF=0 and SF=OF)
0F 91	SETNO *r/m8*	Valid	Valid	Set byte if not overflow (OF=0)
REX + 0F 91	SETNO *r/m8**	Valid	N.E.	Set byte if not overflow (OF=0)
0F 9B	SETNP *r/m8*	Valid	Valid	Set byte if not parity (PF=0)
REX + 0F 9B	SETNP *r/m8**	Valid	N.E.	Set byte if not parity (PF=0)
0F 99	SETNS *r/m8*	Valid	Valid	Set byte if not sign (SF=0)
REX + 0F 99	SETNS *r/m8**	Valid	N.E.	Set byte if not sign (SF=0)
0F 95	SETNZ *r/m8*	Valid	Valid	Set byte if not zero (ZF=0)
REX + 0F 95	SETNZ *r/m8**	Valid	N.E.	Set byte if not zero (ZF=0)
0F 90	SETO *r/m8*	Valid	Valid	Set byte if overflow (OF=1)

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
REX + 0F 90	SETO *r/m8**	Valid	N.E.	Set byte if overflow (OF=1)
0F 9A	SETP *r/m8*	Valid	Valid	Set byte if parity (PF=1)
REX + 0F 9A	SETP *r/m8**	Valid	N.E.	Set byte if parity (PF=1)
0F 9A	SETPE *r/m8*	Valid	Valid	Set byte if parity even (PF=1)
REX + 0F 9A	SETPE *r/m8**	Valid	N.E.	Set byte if parity even (PF=1)
0F 9B	SETPO *r/m8*	Valid	Valid	Set byte if parity odd (PF=0)
REX + 0F 9B	SETPO *r/m8**	Valid	N.E.	Set byte if parity odd (PF=0)
0F 98	SETS *r/m8*	Valid	Valid	Set byte if sign (SF=1)
REX + 0F 98	SETS *r/m8**	Valid	N.E.	Set byte if sign (SF=1)
0F 94	SETZ *r/m8*	Valid	Valid	Set byte if zero (ZF=1)
REX + 0F 94	SETZ *r/m8**	Valid	N.E.	Set byte if zero (ZF=1)

* In 64-bit mode, r/m8 can not be encoded to access the following byte registers if an REX prefix is used: AH, BH, CH, DH. Also refer to Section 1.4.2.2.

Flags Affected

None.

IA-32e Mode Operation

Operand size fixed at 8-bits.

Protected Mode Exceptions

#GP(0)	If the destination is located in a nonwritable segment.
	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the DS, ES, FS, or GS register contains a null segment selector.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.

Real-Address Mode Exceptions

#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS	If a memory operand effective address is outside the SS segment limit.

Virtual-8086 Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the destination is located in a nonwritable segment.
	If the memory address is in a non-canonical form.
#PF(fault-code)	If a page fault occurs.

SFENCE—Store Fence

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F AE /7	SFENCE	Valid	Valid	Serializes store operations.

IA-32e Mode Operation

Same as legacy mode.

Intel C/C++ Compiler Intrinsic Equivalent

void_mm_sfence(void)

Exceptions (All Operating Modes)

None.

SGDT/SIDT—Store Global/Interrupt Descriptor Table Register

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 01 /0	SGDT *m*	Valid	Valid	Store GDTR to *m*
0F 01 /1	SIDT *m*	Valid	Valid	Store IDTR to *m*

Flags Affected

None.

IA-32e Mode Operation

Operand size fixed at 8+2 bytes

Stores 8 byte base and 2 byte limit values.

Protected Mode Exceptions

#UD	If the destination operand is a register.
#GP(0)	If the destination is located in a nonwritable segment.
	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the DS, ES, FS, or GS register is used to access memory and it contains a null segment selector.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#UD	If the destination operand is a register.
#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS	If a memory operand effective address is outside the SS segment limit.

Virtual-8086 Mode Exceptions

#UD	If the destination operand is a register.
#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#UD	If the destination operand is a register.
#GP(0)	If the destination is located in a nonwritable segment.
	If the memory address is in a non-canonical form.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

SHL/SHR—Shift Instructions

See entry for SAL/SAR/SHL/SHR—Shift.

SHLD—Double Precision Shift Left

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F A4	SHLD *r/m16, r16, imm8*	Valid	Valid	Shift *r/m16* to left *imm8* places while shifting bits from *r16* in from the right
0F A5	SHLD *r/m16, r16*, CL	Valid	Valid	Shift *r/m16* to left CL places while shifting bits from *r16* in from the right
0F A4	SHLD *r/m32, r32, imm8*	Valid	Valid	Shift *r/m32* to left *imm8* places while shifting bits from *r32* in from the right
REX.W + 0F A4	SHLD *r/m64, r64, imm8*	Valid	N.E.	Shift *r/m64* to left *imm8* places while shifting bits from *r64* in from the right
0F A5	SHLD *r/m32, r32*, CL	Valid	Valid	Shift *r/m32* to left CL places while shifting bits from *r32* in from the right
REX.W + 0F A5	SHLD *r/m64, r64*, CL	Valid	N.E.	Shift *r/m64* to left CL places while shifting bits from *r64* in from the right

Flags Affected

If the count is 1 or greater, the CF flag is filled with the last bit shifted out of the destination operand and the SF, ZF, and PF flags are set according to the value of the result. For a 1-bit shift, the OF flag is set if a sign change occurred; otherwise, it is cleared. For shifts greater than 1 bit, the OF flag is undefined. If a shift occurs, the AF flag is undefined. If the count operand is 0, the flags are not affected. If the count is greater than the operand size, the flags are undefined.

IA-32e Mode Operation

Default operand size 32-bits.

Uses 6 bit count.

Protected Mode Exceptions

#GP(0)	If the destination is located in a nonwritable segment.
	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the DS, ES, FS, or GS register contains a null segment selector.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS	If a memory operand effective address is outside the SS segment limit.

Virtual-8086 Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the destination is located in a nonwritable segment.
	If the memory address is in a non-canonical form.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

SHRD—Double Precision Shift Right

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F AC	SHRD *r/m16, r16, imm8*	Valid	Valid	Shift *r/m16* to right *imm8* places while shifting bits from *r16* in from the left
0F AD	SHRD *r/m16, r16*, CL	Valid	Valid	Shift *r/m16* to right CL places while shifting bits from *r16* in from the left
0F AC	SHRD *r/m32, r32, mm8*	Valid	Valid	Shift *r/m32* to right *imm8* places while shifting bits from *r32* in from the left
REX.W + 0F AC	SHRD *r/m64, r64, imm8*	Valid	N.E.	Shift *r/m64* to right *imm8* places while shifting bits from *r64* in from the left
0F AD	SHRD *r/m32, r32*, CL	Valid	Valid	Shift *r/m32* to right CL places while shifting bits from *r32* in from the left
REX.W + 0F AD	SHRD *r/m64, r64*, CL	Valid	N.E.	Shift *r/m64* to right CL places while shifting bits from *r64* in from the left

Flags Affected

If the count is 1 or greater, the CF flag is filled with the last bit shifted out of the destination operand and the SF, ZF, and PF flags are set according to the value of the result. For a 1-bit shift, the OF flag is set if a sign change occurred; otherwise, it is cleared. For shifts greater than 1 bit, the OF flag is undefined. If a shift occurs, the AF flag is undefined. If the count operand is 0, the flags are not affected. If the count is greater than the operand size, the flags are undefined.

IA-32e Mode Operation

Promoted to 64-bits.

Default operand size 32-bits.

Uses 6 bit count.

Enables access to new registers R8-R15.

Protected Mode Exceptions

#GP(0)	If the destination is located in a nonwritable segment.
	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the DS, ES, FS, or GS register contains a null segment selector.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS	If a memory operand effective address is outside the SS segment limit.

Virtual-8086 Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the destination is located in a nonwritable segment.
	If the memory address is in a non-canonical form.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

SHUFPD—Shuffle Packed Double-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
66 0F C6 /r ib	SHUFPD *xmm1, xmm2/m128, imm8*	Valid	Valid	Shuffle packed double-precision floating-point values selected by *imm8* from *xmm1* and *xmm2/m128* to *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If the memory address is in a non-canonical form.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

SHUFPS—Shuffle Packed Single-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F C6 /r ib	SHUFPS *xmm1*, *xmm2/m128, imm8*	Valid	Valid	Shuffle packed single-precision floating-point values selected by *imm8* from *xmm1* and *xmm1/m128* to *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If the memory address is in a non-canonical form.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

SIDT—Store Interrupt Descriptor Table Register

See entry for SGDT/SIDT—Store Global/Interrupt Descriptor Table Register.

SLDT—Store Local Descriptor Table Register

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 00 /0	SLDT *r/m16*	Valid	Valid	Stores segment selector from LDTR in r/m16

Flags Affected

None.

IA-32e Mode Operation

The behavior of the SLDT instruction is defined by the following examples.

- SLDT r16 operands size 16, store 16-bit selector in r16
- SLDT r32 operands size 32, zero-extend 16-bit selector and store in r32
- SLDT r64 operands size 64, zero-extend 16-bit selector and store in r64
- SLDT m16 operands size 16, store 16-bit selector in m16
- SLDT m16 operands size 32, store 16-bit selector in m16 (not m32)
- SLDT m16 operands size 64, store 16-bit selector in m16 (not m64)

Protected Mode Exceptions

#GP(0)	If the destination is located in a nonwritable segment.
	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the DS, ES, FS, or GS register is used to access memory and it contains a null segment selector.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#UD	The SLDT instruction is not recognized in real-address mode.

Virtual-8086 Mode Exceptions

#UD	The SLDT instruction is not recognized in virtual-8086 mode.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the destination is located in a nonwritable segment.
	If the memory address is in a non-canonical form.
#PF(fault-code)	If a page fault occurs.

#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

SMSW—Store Machine Status Word

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 01 /4	SMSW *r/m16*	Valid	Valid	Store machine status word to *r/m16*

Flags Affected

None.

IA-32e Mode Operation

The behavior of the SMSW instruction is defined by the following examples.

- SMSW r16 operands size 16, store CR0[15:0] in r16
- SMSW r32 operands size 32, zero-extend CR0[31:0], and store in r32
- SMSW r64 operands size 64, zero-extend CR0[63:0], and store in r64
- SMSW m16 operands size 16, store CR0[15:0] in m16
- SMSW m16 operands size 32, store CR0[15:0] in m16 (not m32)
- SMSW m16 operands size 64, store CR0[15:0] in m16 (not m64)

Protected Mode Exceptions

#GP(0)	If the destination is located in a nonwritable segment.
	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the DS, ES, FS, or GS register is used to access memory and it contains a null segment selector.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.

Virtual-8086 Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

SQRTPD—Compute Square Roots of Packed Double-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
66 0F 51 /r	SQRTPD *xmm1*, *xmm2/m128*	Valid	Valid	Computes square roots of the packed double-precision floating-point values in *xmm2/m128* and stores the results in *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

Invalid, Precision, Denormal.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

SQRTPS—Compute Square Roots of Packed Single-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 51 /r	SQRTPS *xmm1*, xmm2/m128	Valid	Valid	Computes square roots of the packed single-precision floating-point values in *xmm2/m128* and stores the results in *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

Invalid, Precision, Denormal.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

SQRTSD—Compute Square Root of Scalar Double-Precision Floating-Point Value

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
F2 0F 51 /r	SQRTSD *xmm1, xmm2/m64*	Valid	Valid	Computes square root of the low double-precision floating-point value in *xmm2/m64* and stores the results in *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

Invalid, Precision, Denormal.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

SQRTSS—Compute Square Root of Scalar Single-Precision Floating-Point Value

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
F3 0F 51 /r	SQRTSS *xmm1, xmm2/m32*	Valid	Valid	Computes square root of the low single-precision floating-point value in *xmm2/m32* and stores the results in *xmm1.*

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

Invalid, Precision, Denormal.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

STC—Set Carry Flag

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
F9	STC	Valid	Valid	Set CF flag

Flags Affected

The CF flag is set. The OF, ZF, SF, AF, and PF flags are unaffected.

IA-32e Mode Operation

Same as legacy mode.

Exceptions (All Operating Modes)

None.

STD—Set Direction Flag

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
FD	STD	Valid	Valid	Set DF flag

Flags Affected

The DF flag is set. The CF, OF, ZF, SF, AF, and PF flags are unaffected.

IA-32e Mode Operation

Same as legacy mode.

Operation

DF ← 1;

Exceptions (All Operating Modes)

None.

STI—Set Interrupt Flag

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
FB	STI	Valid	Valid	Set interrupt flag; external, maskable interrupts enabled at the end of the next instruction

Flags Affected

The IF flag is set to 1.

IA-32e Mode Operation

Same as legacy mode.

Protected Mode Exceptions

#GP(0)	If the CPL is greater (has less privilege) than the IOPL of the current program or procedure.

Real-Address Mode Exceptions

None.

Virtual-8086 Mode Exceptions

#GP(0)	If the CPL is greater (has less privilege) than the IOPL of the current program or procedure.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

Same as for protected mode exceptions.

STMXCSR—Store MXCSR Register State

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F AE /3	STMXCSR *m32*	Valid	Valid	Store contents of MXCSR register to *m32*.

IA-32e Mode Operation

Same as legacy mode.

Exceptions

None.

Numeric Exceptions

None.

Protected Mode Exceptions

#GP(0)	If the destination operand is in a nonwritable segment.
	For an illegal memory operand effective address in the CS, DS, ES, FS, or GS segments.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#UD	If CR0.EM = 1.
#NM	If TS bit in CR0 is set.
#AC	For unaligned memory reference. To enable #AC exceptions, three conditions must be true(CR0.AM is set; EFLAGS.AC is set; current CPL is 3).
#UD	If CR4.OSFXSR(bit 9) = 0.
#UD	If CPUID.SSE(EDX bit 25) = 0.

Real Address Mode Exceptions

#GP(0)	If any part of the operand would lie outside of the effective address space from 0 to 0FFFFH.
#UD	If CR0.EM = 1.
#NM	If TS bit in CR0 is set.
#UD	If CR4.OSFXSR(bit 9) = 0.
#UD	If CPUID.SSE(EDX bit 25) = 0.

Virtual 8086 Mode Exceptions

Same exceptions as in Real Address Mode.

#PF(fault-code)	For a page fault.
#AC	For unaligned memory reference.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	For a page fault.
#UD	If CR0.EM = 1.
#NM	If TS bit in CR0 is set.
#AC	For unaligned memory reference. To enable #AC exceptions, three conditions must be true(CR0.AM is set; EFLAGS.AC is set; current CPL is 3).
#UD	If CR4.OSFXSR(bit 9) = 0.
#UD	If CPUID.SSE(EDX bit 25) = 0.

STOS/STOSB/STOSW/STOSD/STOSQ—Store String

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
AA	STOS m8	Valid	Valid	For legacy mode, store AL at address ES:(E)DI; For 64-bit mode store AL at address (R)DI.
AB	STOS m16	Valid	Valid	For legacy mode, store AX at address ES:(E)DI; For 64-bit mode store AX at address (R)DI.
AB	STOS m32	Valid	Valid	For legacy mode, store EAX at address ES:(E)DI; For 64-bit mode store EAX at address (R)DI.
REX.W + AB	STOS m64	Valid	N.E.	Store RAX at address RDI
AA	STOSB	Valid	Valid	For legacy mode, store AL at address ES:(E)DI; For 64-bit mode store AL at address (R)DI.
AB	STOSw	Valid	Valid	For legacy mode, store AX at address ES:(E)DI; For 64-bit mode store AX at address (R)DI.
AB	STOSD	Valid	Valid	For legacy mode, store EAX at address ES:(E)DI; For 64-bit mode store EAX at address (R)DI.
REX.W + AB	STOSQ	Valid	N.E.	Store RAX at address RDI

Flags Affected

None.

IA-32e Mode Operation

Promoted to 64-bit.

Default operand size 32-bit. Store string doubleword in 32 bit operand size. Store string quadword in 64-bit operand size.

Protected Mode Exceptions

#GP(0)	If the destination is located in a nonwritable segment.
	If a memory operand effective address is outside the limit of the ES segment.
	If the ES register contains a null segment selector.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP	If a memory operand effective address is outside the ES segment limit.

Virtual-8086 Mode Exceptions

#GP(0)	If a memory operand effective address is outside the ES segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

STR—Store Task Register

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 00 /1	STR *r/m16*	Valid	Valid	Stores segment selector from TR in r/m16

Flags Affected

None.

IA-32e Mode Operation

Same as legacy mode.

Memory operand fixed at 16 bits.

Zero extend 2 byte TR selector to 64 bits and store to register operand.

Protected Mode Exceptions

#GP(0)	If the destination is a memory operand that is located in a nonwritable segment or if the effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the DS, ES, FS, or GS register is used to access memory and it contains a null segment selector.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#UD	The STR instruction is not recognized in real-address mode.

Virtual-8086 Mode Exceptions

#UD	The STR instruction is not recognized in virtual-8086 mode.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#GP(0)	If the memory address is in a non-canonical form.
#SS(U)	If the stack address is in a non-canonical form.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

SUB—Subtract

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
2C *ib*	SUB AL,*imm8*	Valid	Valid	Subtract *imm8* from AL
2D *iw*	SUB AX,*imm16*	Valid	Valid	Subtract *imm16* from AX
2D *id*	SUB EAX,*imm32*	Valid	Valid	Subtract *imm32* from EAX
REX.W + 2D *id*	SUB RAX,*imm32*	Valid	N.E.	Subtract *imm32* sign-extended to 64-bits from RAX
80 /5 *ib*	SUB *r/m8,imm8*	Valid	Valid	Subtract *imm8* from *r/m8*
REX + 80 /5 *ib*	SUB *r/m8*,imm8*	Valid	N.E.	Subtract *imm8* from *r/m8*
81 /5 *iw*	SUB *r/m16,imm16*	Valid	Valid	Subtract *imm16* from *r/m16*
81 /5 *id*	SUB *r/m32,imm32*	Valid	Valid	Subtract *imm32* from *r/m32*
REX.W + 81 /5 *id*	SUB *r/m64,imm32*	Valid	N.E.	Subtract *imm32* sign-extended to 64-bits from *r/m64*
83 /5 *ib*	SUB *r/m16,imm8*	Valid	Valid	Subtract sign-extended *imm8* from *r/m16*
83 /5 *ib*	SUB *r/m32,imm8*	Valid	Valid	Subtract sign-extended *imm8* from *r/m32*
REX.W + 83 /5 *ib*	SUB *r/m64,imm8*	Valid	N.E.	Subtract sign-extended *imm8* from *r/m64*
28 /*r*	SUB *r/m8,r8*	Valid	Valid	Subtract *r8* from *r/m8*
REX + 28 /*r*	SUB *r/m8*,r8**	Valid	N.E.	Subtract *r8* from *r/m8*
29 /*r*	SUB *r/m16,r16*	Valid	Valid	Subtract *r16* from *r/m16*
29 /*r*	SUB *r/m32,r32*	Valid	Valid	Subtract *r32* from *r/m32*
REX.W + 29 /*r*	SUB *r/m64,r32*	Valid	N.E.	Subtract *r64* from *r/m64*
2A /*r*	SUB *r8,r/m8*	Valid	Valid	Subtract *r/m8* from *r8*
REX + 2A /*r*	SUB *r8*,r/m8**	Valid	N.E.	Subtract *r/m8* from *r8*
2B /*r*	SUB *r16,r/m16*	Valid	Valid	Subtract *r/m16* from *r16*
2B /*r*	SUB *r32,r/m32*	Valid	Valid	Subtract *r/m32* from *r32*
REX.W + 2B /*r*	SUB *r64,r/m64*	Valid	N.E.	Subtract *r/m64* from *r64*

* In 64-bit mode, r/m8 can not be encoded to access the following byte registers if an REX prefix is used: AH, BH, CH, DH. Also refer to Section 1.4.2.2.

Flags Affected

The OF, SF, ZF, AF, PF, and CF flags are set according to the result.

IA-32e Mode Operation

Promoted to 64-bits

Default operand size 32-bit.

Enables access to new registers R8-R15.

Protected Mode Exceptions

#GP(0)	If the destination is located in a nonwritable segment.
	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the DS, ES, FS, or GS register contains a null segment selector.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS	If a memory operand effective address is outside the SS segment limit.

Virtual-8086 Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

SUBPD—Subtract Packed Double-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
66 0F 5C /r	SUBPD *xmm1, xmm2/m128*	Valid	Valid	Subtract packed double-precision floating-point values in *xmm2/m128* from *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

Overflow, Underflow, Invalid, Precision, Denormal.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

SUBPS—Subtract Packed Single-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 5C /r	SUBPS *xmm1 xmm2/m128*	Valid	Valid	Subtract packed single-precision floating-point values in *xmm2/mem* from *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

Overflow, Underflow, Invalid, Precision, Denormal.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

SUBSD—Subtract Scalar Double-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
F2 0F 5C /r	SUBSD *xmm1*, xmm2/m64	Valid	Valid	Subtracts the low double-precision floating-point values in *xmm2/mem64* from *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

Overflow, Underflow, Invalid, Precision, Denormal.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

SUBSS—Subtract Scalar Single-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
F3 0F 5C /r	SUBSS *xmm1, xmm2/m32*	Valid	Valid	Subtract the lower single-precision floating-point values in *xmm2/m32* from *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

Overflow, Underflow, Invalid, Precision, Denormal.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

SWAPGS—Swap GS Base Register

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 01 /7	SWAPGS	Valid	Inv.	Exchanges the current GS base register value with the value contained in MSR address C0000102h

Description

SWAPGS exchanges the current GS base register value with the value contained in MSR address C0000102H (MSR_KERNELGSbase). The SWAPGS instruction is a privileged instruction intended for use by system software. When using SYSCALL to implement system calls, there is no kernel stack at the OS entry point. Neither is there a straightforward method to obtain a pointer to kernel structures, from which the kernel stack pointer could be read. Thus, the kernel can't save general purpose registers or reference memory. By design, SWAPGS does not require any general purpose registers or memory operands; therefore, no registers need to be saved before using it. SWAPGS exchanges the CPL 0 data pointer from the KernelGSbase MSR with the GS base register. The kernel can then use the GS prefix on normal memory references to access kernel data structures. Similarly, when the OS kernel is entered via an interrupt or exception (where the kernel stack is already set up), SWAPGS can be used to quickly get a pointer to the kernel data structures.

The KernelGSbase MSR itself is only accessible via the normal RDMSR/WRMSR instructions. Those instructions are only accessible at privilege level 0. WRMSR will cause a #GP(0) if the value to be written into KernelGSbase MSR is non-canonical.

Operation

```
IF mode <>64 then #UD;
IF CPL <>0 then #GP (0);
TEMP = GS base;
GS_base = MSR_KernelGSbase;
MSR_KernelGSbase = temp;
```

Flags Affected

None

IA-32e Mode Operation

SWAPGS exchanges the current GS base register value with the value contained in MSR address C0000102H. KernelGSbase is guaranteed to be canonical; so the SWAPGS instruction itself does not perform a canonical check.

Table 3-1. SWAPGS Operation Parameters

Opcode	ModR/M Byte			Instruction	
	MOD	REG	R/M	Not 64-bit Mode	64-bit Mode
OF 01	MOD <> 11	111	xxx	INVLPG	INVLPG
	11	111	000	#UD	SWPGS
	11	111	<>000	#UD	#UD

Protected Mode Exceptions

#UD If Mode <> 64-Bit

Real-Address Mode Exceptions

#UD Instruction not recognized.

Virtual-8086 Mode Exceptions

#UD	Instruction not recognized.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#GP(0)	If CPL <> 0

SYSCALL—Fast System Call

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 05	SYSCALL	Valid	Inv.	Fast call to privilege level 0 system procedures

Description

SYSCALL saves the RIP of the instruction following the SYSCALL into RCX and loads the new RIP from the LSTAR (64-bit mode). Upon return, SYSRET copies the value saved in RCX into the RIP.

In 64-bit mode, SYSCALL saves RFLAGS (lower 32 bit only) in R11. It then masks RFLAGS with an OS-defined value using the SYSCALL_FLAG_MASK (MSR C000_0084). The actual mask value used by the OS is the complement of the value written to the SYSCALL_FLAG_MASK MSR. None of the bits in RFLAGS are automatically cleared, including IF (except RF).

In 64-bit mode, SYSRET will restore RFLAGS from R11 (the lower 32 bits only).

Software should not alter the CS or SS descriptors in a manner that violates the following assumptions made by the SYSCALL and SYSRET instructions:

- The CS and SS base and limit remain the same for all processes, including the operating system (The base is 0H and the limit is FFFFFFFFH).
- The CS of the SYSCALL target has a privilege level of 0.
- The CS of the SYSRET target has a privilege level of 3.

SYSCALL and SYSRET do not check for violations of these assumptions.

Operation

To be supplied.

Flags Affected

All.

IA-32e Mode Operation

Table 3-2 summarizes mode operation for SYSCALL.

Table 3-2. SYSCALL Operation Parameters

Instruction	Legacy Mode	Compatibility Mode	64-Bit Mode
SYSCALL	No	No	64-bit

Protected Mode Exceptions

#UD If Mode <> 64-bit.

Real-Address Mode Exceptions

#UD Instruction is not recognized in this mode.

Virtual-8086 Mode Exceptions

#UD Instruction is not recognized in this mode.

Compatibility Mode Exceptions

#UD	Instruction is not recognized in this mode.

64-Bit Mode Exceptions

#UD	If IA32_EFER.SCE bit is 0

SYSENTER—Fast System Call

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 34	SYSENTER	Valid	Valid	Fast call to privilege level 0 system procedures

Description

To be supplied.

Operation

To be supplied.

Flags Affected

VM, IF, RF.

IA-32e Mode Operation

Table 3-3 summarizes mode operation for SYSENTER.

Table 3-3. SYSENTER Operation Parameters

Instruction	Legacy Mode	Compatibility Mode	64-Bit Mode
SYSENTER	32-bit	64-bit	64-bit

Protected Mode Exceptions

#GP(0) If SYSENTER_CS_MSR contains zero.

Real-Address Mode Exceptions

#GP(0) Instruction not recognized in this mode.

Virtual-8086 Mode Exceptions

#GP(0) If SYSENTER_CS_MSR contains zero.

Compatibility Mode Exceptions

#GP(0) If SYSENTER_CS_MSR contains zero.

64-Bit Mode Exceptions

#GP(0) If SYSENTER_CS_MSR contains zero.

SYSEXIT—Fast Return from Fast System Call

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 35	SYSEXIT	Valid	Valid	Fast return to privilege level 3 user code.

Description

To be supplied.

Operation

To be supplied.

Flags Affected

None.

IA-32e Mode Operation

Table 3-4 summarizes mode operation for SYSEXIT.

Table 3-4. SYSEXIT Operation Parameters

Instruction	Legacy Mode	Compatibility Mode	64-Bit Mode
SYSEXIT	32-bit	32-bit	32-bit and 64-bit

Protected Mode Exceptions

#GP(0) If CPL <> 0

If SYSENTER_CS_MSR contains zero.

Real-Address Mode Exceptions

#GP(0) If protected mode is not enabled.

Virtual-8086 Mode Exceptions

#GP(0) If SYSENTER_CS_MSR contains zero.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#GP(0) If SYSENTER_CS_MSR contains zero.

If CPL <> 0.

If ECX or EDX contains non-canonical address.

SYSRET—Return From Fast System Call

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 07	SYSRET	Valid	Inv.	Return from fast system call

Description

SYSCALL saves the RIP of the instruction following the SYSCALL into RCX and loads the new RIP from the LSTAR (64-bit mode only). Upon return, SYSRET copies the value saved in RCX into the RIP. SYSRET is supported in 64-bit mode and in compatibility mode via operand size.

SYSRET uses the CS value found in SYSRET_CS (MSR STAR[63:48]). In a return to 64-bit mode using Osize 64, SYSRET sets the CS selector value to SYSRET_CS +10h. The SS is set to SYSRET_CS + 8h. Using Osize 32, SYSRET returns to compatibility mode. For the return to compatibility mode, the new SS selector is set to SYSRET_CS + 24H.

It is the responsibility of the OS to keep the descriptors in the GDT/LDT that correspond to the selectors loaded by SYSCALL and SYSRET consistent with the base, limit and attribute values forced by the these instructions.

Software should not alter the CS or SS descriptors in a manner that violates the following assumptions made by the SYSCALL and SYSRET instructions:

- CS and SS base and limit remain the same for all processes, including the operating system
- CS of the SYSCALL target has a privilege level of 0
- CS of the SYSRET target has a privilege level of 3

SYSCALL and SYSRET do not check for violations of these assumptions.

Operation

To be supplied.

Flags Affected

VM, IF, RF.

IA-32e Mode Operation

Table 3-5 summarizes mode operation for SYSRET.

Table 3-5. SYSRET Operation Parameters

Instruction	Legacy Mode	Compatibility Mode	64-Bit Mode
SYSRET	No	No	64-bit

Protected Mode Exceptions

#UD If Mode <> 64-Bit.

Real-Address Mode Exceptions

#UD Instruction not recognized in this mode.

Virtual-8086 Mode Exceptions

#UD Instruction not recognized in this mode.

Compatibility Mode Exceptions

#UD	Instruction not recognized in this mode.

64-Bit Mode Exceptions

#UD	If IA32_EFER.SCE bit is 0.
#GP(0)	If CPL <> 0.
	If ECX contains non-canonical address.

TEST—Logical Compare

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
A8 *ib*	TEST AL,*imm8*	Valid	Valid	AND *imm8* with AL; set SF, ZF, PF according to result
A9 *iw*	TEST AX,*imm16*	Valid	Valid	AND *imm16* with AX; set SF, ZF, PF according to result
A9 *id*	TEST EAX,*imm32*	Valid	Valid	AND *imm32* with EAX; set SF, ZF, PF according to result
REX.W + A9 *id*	TEST RAX,*imm32*	Valid	N.E.	AND *imm32* sign-extended to 64-bits with RAX; set SF, ZF, PF according to result
F6 /0 *ib*	TEST *r/m8,imm8*	Valid	Valid	AND *imm8* with *r/m8*; set SF, ZF, PF according to result
REX + F6 /0 *ib*	TEST *r/m8*,imm8*	Valid	N.E.	AND *imm8* with *r/m8*; set SF, ZF, PF according to result
F7 /0 *iw*	TEST *r/m16,imm16*	Valid	Valid	AND *imm16* with *r/m16*; set SF, ZF, PF according to result
F7 /0 *id*	TEST *r/m32,imm32*	Valid	Valid	AND *imm32* with *r/m32*; set SF, ZF, PF according to result
REX.W + F7 /0 *id*	TEST *r/m64,imm32*	Valid	N.E.	AND *imm32* sign-extended to 64-bits with *r/m64*; set SF, ZF, PF according to result
84 /*r*	TEST *r/m8,r8*	Valid	Valid	AND *r8* with *r/m8*; set SF, ZF, PF according to result
REX + 84 /*r*	TEST *r/m8*,r8**	Valid	N.E.	AND *r8* with *r/m8*; set SF, ZF, PF according to result
85 /*r*	TEST *r/m16,r16*	Valid	Valid	AND *r16* with *r/m16*; set SF, ZF, PF according to result
85 /*r*	TEST *r/m32,r32*	Valid	Valid	AND *r32* with *r/m32*; set SF, ZF, PF according to result
REX.W + 85 /*r*	TEST *r/m64,r64*	Valid	N.E.	AND *r64* with *r/m64*; set SF, ZF, PF according to result

* In 64-bit mode, r/m8 can not be encoded to access the following byte registers if an REX prefix is used: AH, BH, CH, DH. Also refer to Section 1.4.2.2.

Flags Affected

The OF and CF flags are cleared to 0. The SF, ZF, and PF flags are set according to the result (see the "Operation" section above). The state of the AF flag is undefined.

IA-32e Mode Operation

See table above.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the DS, ES, FS, or GS register contains a null segment selector.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS	If a memory operand effective address is outside the SS segment limit.

Virtual-8086 Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

UCOMISD—Unordered Compare Scalar Double-Precision Floating-Point Values and Set EFLAGS

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
66 0F 2E /r	UCOMISD *xmm1*, *xmm2/m64*	Valid	Valid	Compares (unordered) the low double-precision floating-point values in *xmm1* and *xmm2/m64* and set the EFLAGS accordingly.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

Invalid (if SNaN operands), Denormal.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

UCOMISS—Unordered Compare Scalar Single-Precision Floating-Point Values and Set EFLAGS

Description

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 2E /r	UCOMISS *xmm1, xmm2/m32*	Valid	Valid	Compare lower single-precision floating-point value in *xmm1* register with lower single-precision floating-point value in *xmm2/mem* and set the status flags accordingly.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

Invalid (if SNaN operands), Denormal.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#XM	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 1.
#UD	If an unmasked SIMD floating-point exception and OSXMMEXCPT in CR4 is 0.
	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

UD2—Undefined Instruction

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 0B	UD2	Valid	Valid	Raise invalid opcode exception

Flags Affected

None.

IA-32e Mode Operation

Same as legacy mode.

Exceptions (All Operating Modes)

#UD	Instruction is guaranteed to raise an invalid opcode exception in all operating modes).

UNPCKHPD—Unpack and Interleave High Packed Double-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
66 0F 15 /r	UNPCKHPD *xmm1*, *xmm2/ m128*	Valid	Valid	Unpacks and Interleaves double-precision floating-point values from high quadwords of *xmm1* and *xmm2/m128*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#GP(0)	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

UNPCKHPS—Unpack and Interleave High Packed Single-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 15 /r	UNPCKHPS *xmm1, xmm2/ m128*	Valid	Valid	Unpacks and Interleaves single-precision floating-point values from high quadwords of *xmm1* and *xmm2/mem* into *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

UNPCKLPD—Unpack and Interleave Low Packed Double-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
66 0F 14 /r	UNPCKLPD *xmm1*, *xmm2/m128*	Valid	Valid	Unpacks and Interleaves double-precision floating-point values from low quadwords of *xmm1* and *xmm2/m128*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

UNPCKLPS—Unpack and Interleave Low Packed Single-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 14 /r	UNPCKLPS *xmm1, xmm2/m128*	Valid	Valid	Unpacks and Interleaves single-precision floating-point values from low quadwords of *xmm1* and *xmm2/mem* into *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

VERR, VERW—Verify a Segment for Reading or Writing

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 00 /4	VERR *r/m16*	Valid	Valid	Set ZF=1 if segment specified with *r/m16* can be read
0F 00 /5	VERW *r/m16*	Valid	Valid	Set ZF=1 if segment specified with *r/m16* can be written

Flags Affected

The ZF flag is set to 1 if the segment is accessible and readable (VERR) or writable (VERW); otherwise, it is cleared to 0.

IA-32e Mode Operation

Same as legacy.

Operand size fixed at 16-bits.

Protected Mode Exceptions

The only exceptions generated for these instructions are those related to illegal addressing of the source operand.

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the DS, ES, FS, or GS register is used to access memory and it contains a null segment selector.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#UD	The VERR and VERW instructions are not recognized in real-address mode.

Virtual-8086 Mode Exceptions

#UD	The VERR and VERW instructions are not recognized in virtual-8086 mode.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
9B	WAIT	Valid	Valid	Check pending unmasked floating-point exceptions.
9B	FWAIT	Valid	Valid	Check pending unmasked floating-point exceptions.

FPU Flags Affected

The C0, C1, C2, and C3 flags are undefined.

IA-32e Mode Operation

Same as legacy.

Floating-Point Exceptions

None.

Protected Mode Exceptions

#NM	MP and TS in CR0 is set.
#MF	If there is a pending x87 FPU exception.

Real-Address Mode Exceptions

#NM	MP and TS in CR0 is set.
#MF	If there is a pending x87 FPU exception.

Virtual-8086 Mode Exceptions

#NM	MP and TS in CR0 is set.
#MF	If there is a pending x87 FPU exception.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

Same as for protected mode exceptions.

WBINVD—Write Back and Invalidate Cache

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 09	WBINVD	Valid	Valid	Write back and flush Internal caches; initiate writing-back and flushing of external caches.

Flags Affected

None.

IA-32e Mode Operation

Same as legacy.

Protected Mode Exceptions

#GP(0) If the current privilege level is not 0.

Real-Address Mode Exceptions

None.

Virtual-8086 Mode Exceptions

#GP(0) The WBINVD instruction cannot be executed at the virtual-8086 mode.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

Same as for protected mode exceptions.

WRMSR—Write to Model Specific Register

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 30	WRMSR	Valid	Valid	Write the value in EDX:EAX to MSR specified by ECX
REX.W + 0F 30	WRMSR	Valid	N.E.	Write the value in RDX[31:0]:RAX[31:0] to MSR specified by RCX

Flags Affected

None.

IA-32e Mode Operation

Same as legacy. MSR[63:32] = RDX[31:0], MSR[31:0] = RAX[31:0]

Protected Mode Exceptions

#GP(0) If the current privilege level is not 0.

If the value in ECX specifies a reserved or unimplemented MSR address.

Real-Address Mode Exceptions

#GP If the value in ECX specifies a reserved or unimplemented MSR address.

Virtual-8086 Mode Exceptions

#GP(0) The WRMSR instruction is not recognized in virtual-8086 mode.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

Same as for protected mode exceptions.

XADD—Exchange and Add

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F C0 /r	XADD *r/m8, r8*	Valid	Valid	Exchange *r8* and *r/m8*; load sum into *r/m8*.
REX + 0F C0 /r	XADD *r/m8*, r8**	Valid	N.E.	Exchange *r8* and *r/m8*; load sum into *r/m8*.
0F C1 /r	XADD *r/m16, r16*	Valid	Valid	Exchange *r16* and *r/m16*; load sum into *r/m16*.
0F C1 /r	XADD *r/m32, r32*	Valid	Valid	Exchange *r32* and *r/m32*; load sum into *r/m32*.
REX.W + 0F C1 /r	XADD *r/m64, r64*	Valid	N.E.	Exchange *r64* and *r/m64*; load sum into *r/m64*.

* In 64-bit mode, r/m8 can not be encoded to access the following byte registers if an REX prefix is used: AH, BH, CH, DH. Also refer to Section 1.4.2.2.

Flags Affected

The CF, PF, AF, SF, ZF, and OF flags are set according to the result of the addition, which is stored in the destination operand.

IA-32e Mode Operation

Promoted to 64-bits.

Default operand size 32-bits.

Enables access to new registers R8-R15.

Protected Mode Exceptions

#GP(0)	If the destination is located in a nonwritable segment.
	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the DS, ES, FS, or GS register contains a null segment selector.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS	If a memory operand effective address is outside the SS segment limit.

Virtual-8086 Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.

#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

XCHG—Exchange Register/Memory with Register

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
90+*rw*	XCHG AX, *16*	Valid	Valid	Exchange *r16* with AX
90+*rw*	XCHG *r16*, X	Valid	Valid	Exchange AX with *r16*
90+*rd*	XCHG EAX, *r32*	Valid	Valid	Exchange *r32* with EAX
REX.W + 90+*rd*	XCHG RAX, *r64*	Valid	N.E.	Exchange *r64* with RAX
90+*rd*	XCHG *r32*, EAX	Valid	Valid	Exchange EAX with *r32*
REX.W + 90+*rd*	XCHG *r64*, RAX	Valid	N.E.	Exchange RAX with *r64*
86 /*r*	XCHG *r/m8, r8*	Valid	Valid	Exchange *r8* (byte register) with byte from *r/m8*
REX + 86 /*r*	XCHG *r/m8*, r8**	Valid	N.E.	Exchange *r8* (byte register) with byte from *r/m8*
86 /*r*	XCHG *r8, r/m8*	Valid	Valid	Exchange byte from *r/m8* with *r8* (byte register)
REX + 86 /*r*	XCHG *r8*, r/m8**	Valid	N.E.	Exchange byte from *r/m8* with *r8* (byte register)
87 /*r*	XCHG *r/m16, r16*	Valid	Valid	Exchange *r16* with word from *r/m16*
87 /*r*	XCHG *r16, r/m16*	Valid	Valid	Exchange word from *r/m16* with *r16*
87 /*r*	XCHG *r/m32, r32*	Valid	Valid	Exchange *r32* with doubleword from *r/m32*
REX.W + 87 /*r*	XCHG *r/m64, r64*	Valid	N.E.	Exchange *r64* with quadword from *r/m64*
87 /*r*	XCHG *r32, r/m32*	Valid	Valid	Exchange doubleword from *r/m32* with *r32*
REX.W + 87 /*r*	XCHG *r64, r/m64*	Valid	N.E.	Exchange quadword from *r/m64* with *r64*

* In 64-bit mode, r/m8 can not be encoded to access the following byte registers if an REX prefix is used: AH, BH, CH, DH. Also refer to Section 1.4.2.2.

Flags Affected

None.

IA-32e Mode Operation

Promoted to 64-bits.

Default operand size 32-bits.

Enables access to new registers R8-R15.

Protected Mode Exceptions

#GP(0)	If either operand is in a nonwritable segment.
	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the DS, ES, FS, or GS register contains a null segment selector.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS	If a memory operand effective address is outside the SS segment limit.

Virtual-8086 Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

XLAT/XLATB—Table Look-up Translation

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
D7	XLAT *m8*	Valid	Valid	Set AL to memory byte DS:[(E)BX + unsigned AL]
D7	XLATB	Valid	Valid	Set AL to memory byte DS:[(E)BX + unsigned AL]
REX.W + D7	XLATB	Valid	N.E.	Set AL to memory byte [RBX + unsigned AL]

Flags Affected

None.

IA-32e Mode Operation

Same as legacy mode.

Operand size fixed at 8-bits.

Writes AL [reserves RAX[63:8]

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the DS, ES, FS, or GS register contains a null segment selector.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.

Real-Address Mode Exceptions

#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS	If a memory operand effective address is outside the SS segment limit.

Virtual-8086 Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	If a page fault occurs.

XOR—Logical Exclusive OR

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
34 *ib*	XOR AL,*imm8*	Valid	Valid	AL XOR *imm8*
35 *iw*	XOR AX,*imm16*	Valid	Valid	AX XOR *imm16*
35 *id*	XOR EAX,*imm32*	Valid	Valid	EAX XOR *imm32*
REX.W + 35 *id*	XOR RAX,*imm32*	Valid	N.E.	RAX XOR *imm32 (sign-extended)*
80 /6 *ib*	XOR *r/m8,imm8*	Valid	Valid	*r/m8* XOR *imm8*
REX + 80 /6 *ib*	XOR *r/m8*,imm8*	Valid	N.E.	*r/m8* XOR *imm8*
81 /6 *iw*	XOR *r/m16,imm16*	Valid	Valid	*r/m16* XOR *imm16*
81 /6 *id*	XOR *r/m32,imm32*	Valid	Valid	*r/m32* XOR *imm32*
REX.W + 81 /6 *id*	XOR *r/m64,imm32*	Valid	N.E.	*r/m64* XOR *imm32 (sign-extended)*
83 /6 *ib*	XOR *r/m16,imm8*	Valid	Valid	*r/m16* XOR *imm8 (sign-extended)*
83 /6 *ib*	XOR *r/m32,imm8*	Valid	Valid	*r/m32* XOR *imm8 (sign-extended)*
REX.W + 83 /6 *ib*	XOR *r/m64,imm8*	Valid	N.E.	*r/m64* XOR *imm8 (sign-extended)*
30 /*r*	XOR *r/m8,r8*	Valid	Valid	*r/m8* XOR *r8*
REX + 30 /*r*	XOR *r/m8*,r8**	Valid	N.E.	*r/m8* XOR *r8*
31 /*r*	XOR *r/m16,r16*	Valid	Valid	*r/m16* XOR *r16*
31 /*r*	XOR *r/m32,r32*	Valid	Valid	*r/m32* XOR *r32*
REX.W + 31 /*r*	XOR *r/m64,r64*	Valid	N.E.	*r/m64* XOR *r64*
32 /*r*	XOR *r8,r/m8*	Valid	Valid	*r8* XOR *r/m8*
REX + 32 /*r*	XOR *r8*,r/m8**	Valid	N.E.	*r8* XOR *r/m8*
33 /*r*	XOR *r16,r/m16*	Valid	Valid	*r16* XOR *r/m16*
33 /*r*	XOR *r32,r/m32*	Valid	Valid	*r32* XOR *r/m32*
REX.W + 33 /*r*	XOR *r64,r/m64*	Valid	N.E.	*r64* XOR *r/m64*

* In 64-bit mode, r/m8 can not be encoded to access the following byte registers if an REX prefix is used: AH, BH, CH, DH. Also refer to Section 1.4.2.2.

Flags Affected

The OF and CF flags are cleared; the SF, ZF, and PF flags are set according to the result. The state of the AF flag is undefined.

IA-32e Mode Operation

See table above

Protected Mode Exceptions

#GP(0)	If the destination operand points to a nonwritable segment.
	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the DS, ES, FS, or GS register contains a null segment selector.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS	If a memory operand effective address is outside the SS segment limit.

Virtual-8086 Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

XORPD—Bitwise Logical XOR for Double-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
66 0F 57 /r	XORPD *xmm1*, *xmm2/m128*	Valid	Valid	Bitwise exclusive-OR of *xmm2/m128* and *xmm1*

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE2 is 0.

XORPS—Bitwise Logical XOR for Single-Precision Floating-Point Values

Opcode	Instruction	64-Bit Mode	Compat/Leg Mode	Description
0F 57 /r	XORPS *xmm1, xmm2/ m128*	Valid	Valid	Bitwise exclusive-OR of *xmm2/m128* and *xmm1*.

IA-32e Mode Operation

Enables access to XMM8-XMM15.

SIMD Floating-Point Exceptions

None.

Protected Mode Exceptions

#GP(0)	For an illegal memory operand effective address in the CS, DS, ES, FS or GS segments.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#SS(0)	For an illegal address in the SS segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Real-Address Mode Exceptions

#GP(0)	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
	If any part of the operand lies outside the effective address space from 0 to FFFFH.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

Virtual-8086 Mode Exceptions

Same exceptions as in Real Address Mode

#PF(fault-code)	For a page fault.

Compatibility Mode Exceptions

Same as for protected mode exceptions.

64-Bit Mode Exceptions

#SS(0)	If a memory address referencing the SS segment is in a non-canonical form.
#GP(0)	If the memory address is in a non-canonical form.
	If memory operand is not aligned on a 16-byte boundary, regardless of segment.
#PF(fault-code)	For a page fault.
#NM	If TS in CR0 is set.
#UD	If EM in CR0 is set.
	If OSFXSR in CR4 is 0.
	If CPUID feature flag SSE is 0.

CHAPTER 4
SOFTWARE OPTIMIZATION GUIDELINES

4.1. INTRODUCTION

This chapter describes optimization guidelines for software written to run in IA-32e mode. Software that runs in either compatibility mode or legacy modes should follow the guidelines described in *IA-32 Intel® Architecture Optimization Reference Manual*. Consider this chapter to be an addendum to that manual.

4.2. 64-BIT MODE OPTIMIZATION GUIDELINES

Optimization for software to run in 64-bit mode should start with the coding recommendations described in the *IA-32 Intel Architecture Optimization Reference Manual*. However, there are additional considerations that are applicable to 64-bit mode; these are discussed below.

4.2.1. Coding Rules Affecting 64-bit Mode

4.2.1.1. Use Legacy 32-Bit Instructions When The Data Size Is 32 Bits

64-bit mode makes 16 general purpose 64-bit registers available to applications. If application data size is 32 bits, however, there is no need to use 64-bit registers or 64-bit arithmetic.

The default operand size for most instructions is 32 bits. The behavior of those instructions is to make the upper 32 bits all zeros. For example, when zeroing out a register, the following two instruction streams do the same thing but the 32-bit version saves one instruction byte:

32-bit version

```
xor eax, eax     ;Performs xor on lower 32bits and zeros the upper 32 bits.
```

64-bit version

```
xor rax, rax     ;Performs xor on all 64 bits
```

This optimization holds true for the lower 8 general purpose registers: EAX, ECX, EBX, EDX, ESP, EBP, ESI, EDI. To access the data in registers r9-r15, the REX prefix is required. Using the 32-bit form there does not reduce code size.

Assembly/Compiler Coding rule

Use the 32-bit versions of instructions in 64-bit mode to reduce code size unless the 64-bit version is necessary to access 64-bit data or additional registers.

4.2.1.2. Use the Extra Registers to Reduce Register Pressure

64-bit mode makes 8 additional 64-bit general purpose registers and 8 additional XMM registers available to applications. To access the additional registers, the single byte REX prefix is necessary. Using 8 additional registers can prevent the compiler from needing to spill values onto the stack. The potential increase in code size due to the REX prefix, however, can increase cache misses and this can work against the benefit of using extra registers to access the data (when data comes from the register instead of from memory).

When the 8 additional registers are not needed, refrain from using the registers that require REX prefix. This keeps the code size smaller.

Assembly/Compiler Coding rule

When they are needed to reduce register pressure, use the 8 extra general purpose registers and 8 extra XMM registers for floating-point.

4.2.1.3. Use 64-Bit by 64-Bit Multiplies That Produce 128-Bit Results Only When Necessary

Integer multiplies of 64-bit by 64-bit operands that produce a 128-bit result cost more than multiplies that produce a 64-bit result. The upper 64-bits of a result takes longer to compute than the lower 64 bits.

If the compiler can determine at compile time that the result of a multiply will not exceed 64 bits, then the compiler should generate the multiply instruction that produces a 64-bit result. If the compiler or assembly programmer can not determine that the result will be less than 64 bits, then a multiply that produces 128 bits is necessary.

Assembly/Compiler Coding rule

Prefer 64-bit by 64-bit integer multiplies that produce 64-bit results over multiplies that produce 128-bit results.

4.2.1.4. Sign Extension to Full 64-Bits

When in 64-bit mode, the architecture is optimized to sign-extend to 64 bits in a single uop. In 64-bit mode, when the destination is 32 bits, the upper 32 bits must be zeroed.

Zeroing the upper 32 bits requires an extra uop and is less optimal than sign extending to 64 bits. While sign extending to 64 bits makes the instruction one byte longer, it reduces the number of uops that the trace cache has to store, improving performance.

For example, to sign-extend a byte into esi, use:

```
movsx              rsi, BYTE PTR [rax]
```

instead of:

```
movsx              esi, BYTE PTR [rax]
```

If the next instruction uses the 32-bit form of esi register, the result will be the same. This optimization can also be used to break an unintended dependency. For example, if a program writes a 16-bit value to a register and then writes the register with an 8-bit value, if bits 15:8 of the destination are not needed, use the sign-extended version of writes when available.

For example:

```
mov                r8w, r9w  ;Requires a merge to preserve bits 63:15
…..
mov                r8b, r10b ;Requires a merge to preserve bits 63:8
```

Can be replaced with:

```
movsx              r8, r9w   ;If bits 63:8 do not need to be preserved
…..
movsx              r8, r10b  ;If bits 63:8 do not need to be preserved
```

In the above example, the moves to r8w and r8b both require a merge to preserve the rest of the bits in the register. There is an implicit real dependency on r8 between the 'mov r8w, r9w' and 'mov r8b, r10b'. Using movsx breaks the real dependency and leaves only the output dependency, which the processor can eliminate through renaming.

Assembly/Compiler Coding rule

Sign extend to 64-bits instead of sign extending to 32 bits, even when the destination will be used as a 32-bit value.

4.2.2. Alternate Coding Rules for 64-Bit Mode

4.2.2.1. Use 64-Bit Registers Instead of Two 32-Bit Registers for 64-Bit Arithmetic

Legacy 32-bit mode offers the ability to support extended precision integer arithmetic (such as 64-bit arithmetic). However, 64-bit mode offers native support for 64-bit arithmetic. When 64-bit integers are desired, use the 64-bit forms of arithmetic instructions.

In 32-bit legacy mode, getting a 64-bit result from a 32-bit by 32-bit integer multiply requires three registers; the result is stored in 32-bit chunks in the EDX:EAX pair. When the instruction is available in 64-bit mode, using the 32-bit version of the instruction is not the optimal implementation if a 64-bit result is desired. Use the extended registers.

For example, the following code sequence loads the 32-bit values sign-extended into the 64-bit registers and performs a multiply:

```
movsx           rax, DWORD PTR [x]
movsx           rcx, DWORD PTR [y]
imul            rax, rcx
```

The 64-bit version above is more efficient than using the following 32-bit version:

```
mov             eax, DWORD PTR [x]
mov             ecx, DWORD PTR [y]
imul            ecx
```

In the 32-bit case above, EAX is required to be a source. The result ends up in the EDX:EAX pair instead of in a single 64-bit register.

Assembly/Compiler Coding rule

Use the 64-bit versions of multiply for 32-bit integer multiplies that require a 64 bit result.

To add two 64-bit numbers in 32-bit legacy mode, the add instruction followed by the addc instruction is used. For example, to add two 64-bit variables (X and Y), the following four instructions could be used:

```
mov             eax, DWORD PTR [X]
mov             edx, DWORD PTR [X+4]
add             eax, DWORD PTR [Y]
adc             edx, DWORD PTR [Y+4]
```

The result will end up in the two-register EDX:EAX.

In 64-bit mode, the above sequence can be reduced to the following:

```
mov             rax, QWORD PTR [X]
add             rax, QWORD PTR [Y]
```

The result is stored in rax. One register is required instead of two.

Assembly/Compiler Coding rule

Use the 64-bit versions of add for 64-bit adds.

4.2.3. Other Coding Rules

4.2.3.1. Use 32-Bit Versions of CVTSI2SS and CVTSI2SD When Possible

The CVTSI2SS and CVTSI2SD instructions convert a signed integer in a general-purpose register or memory location to a single-precision or double-precision floating-point value. The signed integer can be either 32-bits or 64-bits. The 32-bit version will result in traces delivered out of the trace cache; the 64-bit version will result in a microcode flow from the microcode ROM and will take longer to execute. In most cases, the 32-bit versions of CVTSI2SS and CVTSI2SD is sufficient.

Assembly/Compiler Coding rule

Use the 32-bit versions of CTVSI2SS and CVTSI2SD when possible.

4.2.3.2. Using Software Prefetch

Intel recommends that software developers follow the recommendations in the *IA-32 Intel® Architecture Optimization Reference Manual* when considering the choice of organizing data access patterns to take advantage of the hardware prefetcher (versus using software prefetch).

Assembly/Compiler Coding rule

If software prefetch instructions are necessary, use the prefetch instructions provided by SSE.

APPENDIX A SMRAM STATE SAVE MAP

When the processor initially enters SMM, it writes its state to the state save area of the SMRAM. The state save area begins at [SMBASE + 8000H + 7FFFH] and extends down to [SMBASE + 8000H + 7C00H]. If the processor reports CPUID.80000001.EDX[29] = 1, The layout of the SMRAM state save map is shown in

Offset (Added to SMBASE + 8000H)	Register
7FF8H	CR0
7FF0H	CR3
7FE8	RFLAGS
7FE0H	IA32_EFER
7FD8H	RIP
7FD0H	DR6
7FC8H	DR7
7FC4H	TR SEL
7FC0H	LDTR SEL
7FBCH	GS SEL
7FB8H	FS SEL
7FB4H	DS SEL
7FB0H	SS SEL
7FACH	CS SEL
7FA8H	ES SEL
7FA4H	IO_MISC
7F9CH	IO_MEM_ADDR
7F94H	RDI
7F8CH	RSI
7F84H	RBP
7F7CH	RSP
7F74H	RBX
7F6CH	RDX
7F64H	RCX
7F5CH	RAX
7F54H	R8
7F4CH	R9
7F44H	R10
7F3CH	R11
7F34H	R12
7F2CH	R13
7F24H	R14
7F1CH	R15

Offset (Added to SMBASE + 8000H)	Register
7F08H-7F1BH	Reserved
7F04H	IEDBASE
7F02H	I/O Instruction Restart Field (Word)
7F00H	Auto HALT Restart Field (Word)
7EFCH	SMM Revision Identifier Field (Doubleword)
7EF8H	SMBASE Field (Doubleword)
7EF7H - 7EA8H	Reserved
7EA4H	LDT Info
7EA0H	LDT Limit
7E9CH	LDT Base (Lower 32 bits)
7E98H	IDT Limit
7E94H	IDT Base (Lower 32 bits)
7E90H	GDT Limit
7E8CH	GDT Base (Lower 32 bits)
7E8BH - 7E44H	Reserved
7E40H	CR4
7E3FH - 7DF0H	Reserved
7DE8H	IO_EIP
7DE7H - 7DDCH	Reserved
7DD8H	IDT Base (Upper 32 bits)
7DD4H	LDT Base (Upper 32 bits)
7DD0H	GDT Base (Upper 32 bits)
7DCFH - 7C00H	Reserved

APPENDIX B
MACHINE CHECK ARCHITECTURE SUPPORT

The Intel® Pentium® 4 and Intel® Xeon™ processor families enhance the P6 Machine Check Architecture to include the Enhanced Machine Check State Registers. The purpose of these registers is to capture the major architecturally visible state of the processor at the time a machine check exception occurs. The state captured includes the general-purpose registers, EFLAGS, stack pointer and instruction pointer. These registers are persistent across a warm system reset and are initialized to zero by default at cold reset.

B.1. MACHINE CHECK ARCHITECTURE

Information about machine check architecture extensions (Detection, Number of extended state registers, MSR addresses) is available in the *IA-32 Intel® Architecture Software Developer's Manual, Volume 3*.

B.2. 64-BIT MODE SPECIFIC EXTENSIONS/MODIFICATIONS

The Machine Check Architecture on a processor supporting Intel EM64T is enhanced to save the additional register state as part of the Enhanced Machine Check State Registers. This includes:

- MSRs 0180H through 0189H save the state in RAX RBX, RCX, RDX, RSI, RDI, RBP, RSP, RFLAGS and RIP respectively, at the time the machine check occurs.
- MSRs 018AH - 018FH are reserved.
- MSRs 0190H through 0197H save the state in registers R8 through R15 respectively, at the time the machine check occurs.
- As part of saving architectural state associated with a machine check exception, the operating system should save all the MSRs from 0180H onwards as specified by the count field in IA32_MCG_CAP MSR.

B.3. INTERPRETING THE MCA ERROR CODES

The architectural error codes logged in the machine check banks, specifically in the MCA Error Code field in the MCI_STATUS registers of a processor enabled with Intel EM64T, can be interpreted using techniques outlined in Chapter 14 of the *IA-32 Intel Architecture Software Developer's Manual, Volume 3*.

APPENDIX C
DEBUG SUPPORT

Intel® Pentium® 4 and Intel® Xeon™ processors extended debug mechanisms available on the P6 family of processors. One of the extensions included replacing the Last Branch Model Specific registers (LastBranchToIP and LastBranchFromIP) with a circular stack of four Last Branch Record (LBR) registers. This permits the stack to contain information for the four most recent branches.

Model Specific Registers MSR_LASTBRANCH_0, MSR_LASTBRANCH_1, MSR_LASTBRANCH_2 and MSR_LASTBRANCH_3 form the circular stack of LBRs. Each LBR records the source and the target information for each taken branch executed by the processor. An additional control register (the last branch record pointer MSR_LASTBRANCH_TOS) can be read to determine which physical LBR corresponds to the most recent branch.

For more information about the LBR stack, see the *IA-32 Intel® Architecture Software Developer's Manual, Volume 3*.

C.1. LAST BRANCH RECORD STACK

The number of Last Branch Record (LBR) registers has increased from four to sixteen, starting with IA-32 processors with CPUID signature family 15 and model 3. For these processors, MSR_LASTBRANCH_TOS now contains a 4-bit pointer (bits 0 through 3) to the MSR in the LBR stack that contains the most recent branch, interrupt or exception.

For each LBR, the processor stores the "From" and "To" linear addresses in different MSRs, with the lower 32-bits of the two MSRs storing the "From" and "To" linear addresses, respectively. Addresses for the "From" MSRs (MSR_LASTBRANCH_0_FROM_LIP through MSR_LASTBRANCH_15_FROM_LIP) are 0680H through 068FH. Addresses for the "To" MSRs (MSR_LASTBRANCH_0_TO_LIP through MSR_LASTBRANCH_15_TO_LIP) are 06C0H through 06CFH.

The registers in the LBR MSR stack and the TOS MSR are writable by WRMSR instruction. The Last Branch Record feature is a model-specific feature and can change between processor models and families.

C.1.1. 64-bit Mode Specific Extensions/Modifications

The LBR stack is enhanced to store the entire 64-bit linear addresses of taken branches in both the "From" and "To" MSRs on a processor supporting Intel® EM64T.

C.2. DEBUG - BRANCH TRACE STORE

Pentium 4 and Intel Xeon processors introduced the Branch Trace Store Mechanism (BTS feature). This mechanism allows for saving branch records in a memory resident buffer. Details are available in *IA-32 Intel Architecture Software Developer's Manual, Volume 3*.

The BTS feature is architectural; support for it can be detected using CPUID. If supported, all system software interfaces for setting up BTS are identical across implementations.

C.2.1. 64-bit Mode Extensions/Modifications

On an IA-32 processor supporting Intel EM64T, the BTS feature is extended in 64-bit mode to support the storing of 64-bit "To" & "From" linear addresses of taken branches and the set up of a memory buffer using 64-bit linear addresses. The changes are:

- IA32_DS_MSR is a 64-bit MSR storing the 64-bit linear address of the base of the DS Save Area.

- The DS Save Area is extended to allow for 64-bit pointers into the buffer (see Table C-1).

Table C-1. DS Save Area

Offset	Contents
00H	BTS buffer base
08H	BTS index
10H	BTS Absolute Max
18H	BTS Interrupt Threshold
20H	PEBS Buffer Base
28H	PEBS Index
30H	PEBS Absolute Max
38H	PEBS Interrupt Threshold
40H	PEBS Counter Reset

- The branch record format in the BTS buffer has been extended for 64-bit "From" and "To" linear addresses (see Table C-2).

Table C-2. Branch Trace Record Format

Offset	Contents
00H	64-bit "from" linear address
08H	64-bit "to" linear address
10H	64-bits with the branch prediction information

NOTE

The format for the DS Save area is different for IA-32e and legacy modes. If the operating system chooses to switch between modes, it must tear-down the BTS layout prior to switching modes and re-establish the buffer following the switch.

APPENDIX D
PERFORMANCE MONITORING SUPPORT

Performance monitoring details are provided in the *IA-32 Intel® Architecture Software Developer's Manual, Volume 3*.

D.1. 64-BIT MODE SPECIFIC EXTENSIONS/MODIFICATIONS

There are no 64-bit mode specific extensions/modifications to event counting and imprecise sampling of the Performance Monitoring capabilities.

When an IA-32 processor is in IA-32e mode, the PEBS feature is extended to support both the storing of entire extended architectural state as well as the set up of the memory buffer using 64-bit linear addresses. The relevant changes are:

- IA32_DS_MSR becomes a 64-bit MSR storing the 64-bit linear address of the base of the DS Save Area.
- The DS Save Area is extended to allow for 64-bit pointers into the buffer.
- The PEBS record format in the PEBS buffer is extended for the entire extended state (Table D-1).

Table D-1. PEBS Record Format

Offset	Contents
00H	RFLAGS
08H	RIP
10H	RAX
18H	RBX
20H	RCX
28H	RDX
30H	RSI
38H	RDI
40H	RBP
48H	RSP
50H	R8
...	...
88H	R15

NOTE

Formats for the DS Save area and PEBS records are different in IA-32e mode and legacy mode. If the operating system chooses to it switch between modes, it must tear-down the PEBS set up prior to switching modes and re-establish the buffer following the mode switch.

APPENDIX E
SMRAM STATE SAVE MAP

E.1. SMRAM STATE SAVE MAP

When the processor initially enters SMM, it writes its state to the state save area of the SMRAM. The state save area begins at [SMBASE + 8000H + 7FFFH] and extends down to [SMBASE + 8000H + 7C00H]. If the processor reports CPUID.80000001.EDX[29] = 1, The layout of the SMRAM state save map is shown in Table E-1.

Table E-1. SMRAM State Save Map

Offset (Added to SMBASE + 8000H)	Register
7FF8H	CR0
7FF0H	CR3
7FE8	RFLAGS
7FE0H	IA32_EFER
7FD8H	RIP
7FD0H	DR6
7FC8H	DR7
7FC4H	TR SEL
7FC0H	LDTR SEL
7FBCH	GS SEL
7FB8H	FS SEL
7FB4H	DS SEL
7FB0H	SS SEL
7FACH	CS SEL
7FA8H	ES SEL
7FA4H	IO_MISC
7F9CH	IO_MEM_ADDR
7F94H	RDI
7F8CH	RSI
7F84H	RBP
7F7CH	RSP
7F74H	RBX
7F6CH	RDX
7F64H	RCX
7F5CH	RAX
7F54H	R8
7F4CH	R9
7F44H	R10
7F3CH	R11
7F34H	R12

Table E-1. SMRAM State Save Map (Contd.)

Offset (Added to SMBASE + 8000H)	Register
7F2CH	R13
7F24H	R14
7F1CH	R15
7F08H-7F1BH	Reserved
7F04H	IEDBASE
7F02H	I/O Instruction Restart Field (Word)
7F00H	Auto HALT Restart Field (Word)
7EFCH	SMM Revision Identifier Field (Doubleword)
7EF8H	SMBASE Field (Doubleword)
7EF7H - 7EA8H	Reserved
7EA4H	LDT Info
7EA0H	LDT Limit
7E9CH	LDT Base (Lower 32 bits)
7E98H	IDT Limit
7E94H	IDT Base (Lower 32 bits)
7E90H	GDT Limit
7E8CH	GDT Base (Lower 32 bits)
7E8BH - 7E44H	Reserved
7E40H	CR4
7E3FH - 7DF0H	Reserved
7DE8H	IO_EIP
7DE7H - 7DDCH	Reserved
7DD8H	IDT Base (Upper 32 bits)
7DD4H	LDT Base (Upper 32 bits)
7DD0H	GDT Base (Upper 32 bits)
7DCFH - 7C00H	Reserved

This index covers the material in Volumes 1 and 2 of the specification. Volume 1 houses Chapters 1 and 2. Volume 2 houses the remainder of the specification.

A

B

C

D

E

F

H - I

J

L

M

N

O

P

R

S

T

U

V

W

X

Z