インテル® エクステンデッド・メモリ 64 テクノロジ・ソフトウェア・デベロッパーズ・ガイド

第 1 巻（全 2 巻）

リビジョン 1.1

注記：
本書は、第 1 巻と第 2 巻で構成されています。ソフトウェアを設計する際は、第 1 巻と第 2 巻の両方を参照してください。

300834-002JA

目 次

この目次には、第1巻と第2巻の両方の項目が含まれています。第1巻には目次と第1章、第2章が収録され、第2巻にはその残りが収録されています。

第 2 章 命令セット・リファレンス（A-L） 2-1

1 はじめに

1.1. インテル® エクステンデッド・メモリ 64 テクノロジ

本書では、64 ビットアドレス拡張技術をサポートするインテル® IA-32 アーキテクチャの拡張について説明する。この拡張には、新しい動作モード、新しい命令、拡張された命令が含まれる。第 1 章は、インテル® エクステンデッド・メモリ 64 テクノロジ（インテル® EM64T）のソフトウェアから見える変更点について説明する。第 2 章と第 3 章は、各種の動作モードでの命令について説明する。第 4 章は、コーディング規則と適用可能な最適化手法について説明する。

インテル EM64T は、インテル IA-32 アーキテクチャを拡張する技術である。このテクノロジを搭載した IA-32 プロセッサは、既存の IA-32 ソフトウェアと互換性がある。このプロセッサ上では、ソフトウェアはより大きなメモリアドレス空間にアクセスできる。また、32 ビット・リニア・アドレス空間向けに開発されたソフトウェアと、64 ビット・リニア・アドレス空間にアクセスするソフトウェアを共存させることができる。

1.2. 動作モード

インテル® EM64T では、IA-32e モードと呼ばれる新しい動作モードが導入されている。IA-32e モードは、互換モードと 64 ビットモードの 2 つのサブモードで構成される。(1) 互換モードでは、64 ビット・オペレーティング・システム上で、従来の 32 ビット・ソフトウェアの大部分が修正なしで動作する。(2) 64 ビットモードでは、64 ビット・オペレーティング・システム上で、64 ビットアドレス空間向けに開発されたアプリケーションが動作する。

インテル EM64T の 64 ビット・サブモードでは、アプリケーションは以下の機能を利用できる。

- 64 ビットのフラットなリニアアドレス指定
- 8 個の新しい汎用レジスタ（GPR）
- ストリーミング SIMD 拡張命令（SSE、SSE2、SSE3）用の 8 個の新しいレジスタ
- 64 ビット長の GPR と命令ポインタ
- 共通のバイトレジスタ・アドレス指定

- 高速の割り込み優先度制御機構
- 新しい命令ポインタ相対アドレス指定モード

インテル EM64T を搭載したプロセッサは、従来の IA-32 モードでも IA-32e モードでも動作する。従来の IA-32 モードでは、プロセッサは、保護モード、実アドレスモード、仮想 8086 モードで動作する。インテル EM64T を搭載したプロセッサは、最初はページングをイネーブルにした従来の保護モードで動作する。次に、IA32-EFER レジスタ内のビットがセットされ、PAE モードがイネーブルになると（1.3.3.1. 節を参照）、プロセッサは IA-32e モードに移行する。表 1-1. は、サポートしている動作モードと、各モードの相違点を示している。

表 1-1. IA-32e モード

モード		必要なオペレーティング・システム	アプリケーションの再コンパイルの必要	デフォルトアドレスサイズ（ビット）	デフォルトオペランド・サイズ（ビット）	レジスタの拡張	GPR の幅（ビット）	SMM によるサポート
IA-32e モード	64 ビットモード	64 ビット OS	あり	64	32	あり	64	あり *
	互換モード		なし	32	32	なし	32	あり
				16	16		16、8	

* SMM は、64 ビット OS と従来の OS の間の移行をサポートする。ただし、SMM 環境内では、PAE と 64 ビットのリニアアドレスは利用できない。

1.2.1. IA-32e モード

IA-32e モードは、64 ビットモードと互換モードの 2 つのサブモードで構成される。IA-32e モードに移行するには、64 ビット対応オペレーティング・システムを起動する必要がある。IA-32e モードへの移行手順については、1.3.3.1. 節と 1.6.3.2. 節で説明する。

1.2.2. 64 ビットモード

64 ビットモードは、64 ビット・オペレーティング・システム上で動作する 64 ビット・アプリケーションによって使用される。64 ビットモードは、以下の機能をサポートしている。

- アーキテクチャ上での 64 ビットのリニアアドレスをサポート。ただし、インテル EM64T 対応の IA-32 プロセッサが実装しているアドレスは、64 ビットより小さい場合がある（1.3.3.3. 節と 1.5.2. 節を参照）。
- 一連の新しいオペコード・プリフィックス（REX）によってアクセス可能なレジスタの拡張
- 64 ビットに拡張された既存の汎用レジスタ（RAX、RBX、RCX、RDX、RSI、RDI、RBP、RSP）

- 8個の新しい汎用レジスタ（R8～R15）
- 8個の新しい128ビット・ストリーミングSIMD拡張命令レジスタ（XMM8～XMM15）
- 64ビット命令ポインタ（RIP）
- 新しいRIP相対データアドレス指定モード
- 単一のコード、データ、スタック空間を持つフラットなアドレス空間の使用
- 拡張された新しい命令
- 64GBを超える物理アドレスのサポート。インテルEM64T対応のIA-32プロセッサの実際の物理アドレスサイズは、プロセッサ・モデルによって異なる。
- 新しい割り込み優先度制御機構

64ビットモードは、オペレーティング・システムによってコード・セグメントごとにイネーブルにされる。64ビットモードでは、デフォルトのアドレスサイズは64ビットで、デフォルトのオペランド・サイズは32ビットである。これらのデフォルト値は、新しいREXオペコード・プリフィックスを使用して、命令ごとに変更できる。REXプリフィックスにより、64ビットモードでの動作中に32ビット・オペランドの指定が可能になる。このメカニズムを利用して、多くの既存の命令が、大きな64ビット・レジスタと64ビットアドレスを使用できるように、変更または再定義されている。

1.2.3. 互換モード

互換モードでは、従来の16ビット・アプリケーションと32ビット・アプリケーションの大部分は、再コンパイルの必要なく、64ビット・オペレーティング・システム上で動作する。なお、互換モードでは、従来のアプリケーションのうち、仮想8086モードで動作するものとハードウェア・タスク管理を使用するものは正常に動作しない。64ビットモードと同じように、互換モードは、オペレーティング・システムによってコード・セグメントごとにイネーブルにされる。したがって、64ビット・アプリケーションと（64ビット向けに再コンパイルされていない）従来の32ビット・アプリケーションは、同時に（64ビットモードの）プロセッサ上で動作できる。

互換モードは、従来の保護モードによく似ている。アプリケーションは、リニアアドレス空間の最初の4GB、標準のIA-32命令プリフィックス、標準のIA-32レジスタにのみアクセスする。互換モードでは、REXプリフィックスは無効である（REXプリフィックスのエンコーディングは、従来のIA-32命令として扱われる）。互換モードでは、16ビットと32ビットのアドレスとオペランド・サイズが使用される。従来の保護モードと同じように、互換モードでは、アプリケーションは物理アドレス拡張機構（PAE）を使用して最大64GBの物理メモリにアクセスできる。

従来の保護モードの以下の機能は、互換モードではサポートされない。

- 仮想8086モード、タスクスイッチ、スタック・パラメータのコピー機能
- オペレーティング・システムの観点から見た、システムデータ構造、アドレス変換、割り込みと例外の処理。これらの構造やイベントの処理には、32ビット・メカニズムではなく64ビット・メカニズムが使用される。

1.2.4. レガシーモード

レガシーモードには、保護モード、実アドレスモード、仮想8086モードが含まれる。これらのモード向けに作成されたソフトウェアは、インテルEM64T対応のプロセッサとの完全な互換性を持つ。

1.2.5. システム管理モード

インテルEM64T対応のプロセッサ内のシステム管理モード（SMM）は、従来のIA-32環境で提供していたシステム管理割り込み（SMI）ハンドラと同じ実行環境を提供する。SMIが伝達されると、プロセッサはSMMに切り替え、SMRAMステート・セーブ・マップに従ってプロセッサ・ステートをセーブする。

SMMは、異なる動作モード（IA-32eモードとレガシーモード）間の移行をサポートする。SMIハンドラは、PSE機構を使用して、すべての物理メモリページにアクセスできる。SMM環境は、PAEがサポートされないため、64ビットのリニアアドレスをサポートしない。

1.3. レジスタセットの変更

本節では、レジスタセットの変更について説明する。表 1-2. は、IA-32e モードで動作するアプリケーションから見たレジスタおよびデータ構造と、従来の IA-32 環境で動作するアプリケーションから見たレジスタおよびデータ構造の比較を示している。従来の環境には、既存の IA-32 プロセッサ内の動作モード、インテル® EM64T 対応のプロセッサ内のレガシーモード、IA-32e 互換モードが含まれる。

互換モードのアプリケーションは、64 ビットモードを認識しない。互換モードで正常に動作しなければならないアプリケーションは、従来の IA-32 保護モード環境で動作するように設計しなければならない。

表 1-2. レジスタセットの変更

ソフトウェアから見えるレジスタ	64 ビットモード			レガシーモードと互換モード		
	名前	数	サイズ（ビット）	名前	数	サイズ（ビット）
汎用レジスタ	RAX、RBX、RCX、RDX、RBP、RSI、RDI、RSP、R8 ～ 15	16	64	EAX、EBX、ECX、EDX、EBP、ESI、EDI、ESP	8	32
命令ポインタ	RIP	1	64	EIP	1	32
フラグ	EFLAGS	1	32	EFLAGS	1	32
FP レジスタ	ST0 ～ 7	8	80	ST0 ～ 7	8	80
マルチメディア・レジスタ	MM0 ～ 7	8	64	MM0 ～ 7	8	64
ストリーミング SIMD レジスタ	XMM0 ～ 15	16	128	XMM0 ～ 7	8	128
スタック幅	-		64	-		16 または 32

1.3.1. 汎用レジスタ（GPR）

レガシーモードまたは互換モードで動作している IA-32 アーキテクチャ内には、8 個の汎用レジスタ（GPR）がある。オペランド・サイズが 16 ビットの場合は、AX、BX、CX、DX、DI、SI、BP、SP が利用できる。オペランド・サイズが 32 ビットの場合は、EAX、EBX、ECX、EDX、EDI、ESI、EBP、ESP が利用できる。

64 ビットモードでは、デフォルト・オペランドは 32 ビットである。ただし、GPR は、32 ビット・オペランドと 64 ビット・オペランドのいずれでも利用できる。32 ビット・オペランドが指定されている場合は、EAX、EBX、ECX、EDX、EDI、ESI、EBP、ESP、R8D ～ R15D が利用できる。64 ビット・オペランドが指定されている場合は、RAX、RBX、RCX、RDX、RDI、RSI、RBP、RSP、R8 ～ R15 が利用できる。R8 ～ R15 が、8 個の新しい GPR である。これらのすべてのレジスタは、バイト、ワード、ダブ

ルワード、クワッドワードの各レベルでアクセスできる。グラニュラリティのレベルをイネーブルにするには、REXプリフィックスを使用する（1.4.2. 節を参照）。

64 ビットモードでは、命令がアクセスできるバイトレジスタに制限がある。命令は、従来の上位バイト（例えば、AH、BH、CH、またはDH）と新しいバイトレジスタのうち1つ（例えば、RAX レジスタの下位バイト）を同時に参照はできない。しかし、命令は、従来の下位バイト（例えば、AL、BL、CL、またはDL）と新しいバイトレジスタ（例えば、R8 レジスタまたはRBP の下位バイト）を同時に参照するのは可能である。アーキテクチャは、REXプリフィックスを持つ任意の命令について、上記の制限を強制するために、上位バイトの参照（AH、BH、CH、DH）を下位バイトの参照（BPL、SPL、DIL、SIL。これらはRBP、RSP、RDI、RSI の下位8 ビットである）に変更する。

64 ビットモードでは、オペランドのサイズによって、デスティネーションGPR 内の有効ビット数が決まる。

- 64 ビット・オペランドは、デスティネーションGPR 内に64 ビットの結果を生成する。
- 32 ビット・オペランドは、デスティネーションGPR 内に、64 ビットに0 で拡張される32 ビットの結果を生成する。
- 8 ビット・オペランドと16 ビット・オペランドは、8 ビットまたは16 ビットの結果を生成する。デスティネーション GPR の上位 56 ビット（結果が 8 ビットの場合）または48 ビット（結果が 16 ビットの場合）は、この演算によって変更されない。8 ビット演算または16 ビット演算の結果を64 ビットアドレス計算に使用する場合は、その結果を明示的にフル64 ビットに符号拡張する必要がある。

32 ビットモードでは、64 ビットGPR の上位32 ビットは未定義であるため、64 ビットモードから32 ビットモード（例えば、レガシーモードまたは互換モード）への切り替えの際に、GPR の上位 32 ビットは維持されない。ソフトウェアは、64 ビットモードから32 ビットモードへの切り替え後に、これらの未定義の上位ビットの値が維持されていると見なしてはならない。これらの値は、ハードウェア・モデル間またはサイクル間で変更されることがある。

1.3.2. ストリーミング SIMD 拡張命令（SSE）レジスタ

互換モードとレガシーモードでは、SSE レジスタは従来の 8 個の 128 ビット・レジスタ（XMM0 ～ XMM7）で構成される。64 ビットモードでは、追加の8 個の128 ビットSSE レジスタを利用できる（XMM8 ～ XMM15）。これらのレジスタへのアクセスは、REX 命令プリフィックスによって命令ごとに制御される。

XMM レジスタは、どのモードでも、SSE、SSE2、SSE3 で使用できる。

1.3.3. システムレジスタ

インテル® EM64T では、新しいレジスタが導入され、既存のシステムレジスタが変更されている。変更されたレジスタは、以下のとおりである。

- MSR。拡張機能イネーブル MSR（IA32_EFER）は、インテル EM64T の機能の制御、イネーブル、ディスエーブル用のビットを格納する。KernelGSbase MSR については、1.4.4. 節を参照のこと。STAR、LSTAR、CSTAR、FMASK の各 MSR については、1.6.9.3. 節を参照のこと。FS.base MSR と GS.base MSR については、1.6.4.2. 節を参照のこと。
- **制御レジスタ**。すべての制御レジスタは、64 ビットに拡張されている。新しい制御レジスタ（タスク・プライオリティ・レジスタ : CR8 または TPR）が追加されている。
- **ディスクリプタ・テーブル・レジスタ**。グローバル・ディスクリプタ・テーブル・レジスタ（GDTR）と割り込みディスクリプタ・テーブル・レジスタ（IDTR）は、フル 64 ビットのベースアドレスを格納できるように、10 バイトに拡張されている。ローカル・ディスクリプタ・テーブル・レジスタ（LDTR）とタスクレジスタ（TR）も、フル 64 ビットのベースアドレスを格納できるように拡張されている。表 1-6. を参照のこと。
- **デバッグレジスタ**。デバッグレジスタは 64 ビットに拡張されている。

1.3.3.1. 拡張機能イネーブル・レジスタ（IA32_EFER）

拡張機能イネーブル・レジスタ（IA32_EFER）は、制御ビットを格納する。このレジスタはアドレス C0000080H にある。表 1-3. は、IA32_EFER のビットについてまとめたものである。各ビットの定義は、表 1-4. に記載されている。

表 1-3. 拡張機能イネーブル MSR（IA32_EFER）

63:11	10	9	8	7:1	0
予約済み	IA-32e モード・アクティブ（LMA）	予約済み	IA-32e モード・イネーブル（LME）	予約済み	SysCall イネーブル（SCE）

表 1-4. IA32_EFER ビットの説明

名前	説明	動作
LMA	IA-32e モード・ アクティブ（ビット10）	このビットは読み取り専用ステータス・ビットである。LMA をセットしようとすると、何も反応なく書き込みは無視される。このビットは、IA-32e モードがアクティブになっていることを示す。 IA-32e モードとページングの両方がイネーブルにされると、プロセッサは LMA を 1 にセットする。LMA = 1 の場合、プロセッサは、表 1-16. に示すコード・セグメント・ディスクリプタの L ビットと D ビットの値に基づいて、互換モードまたは 64 ビットモードになっている。 LMA = 0 の場合、プロセッサは、レガシーモードで動作している。レガシーモードでは、プロセッサは従来の 32 ビット IA-32 プロセッサと同じように動作する。
LME	IA-32e モード・ イネーブル（ビット 8）	このビットを 1 にセットすると、プロセッサは IA-32e モードへの切り替えが可能となる。IA-32e モードは、ソフトウェアが PAE モードのページングをイネーブルにしたときに実際にアクティブになる。 LME が 1 にセットされているときに PAE ページがイネーブルになると、プロセッサは、IA32_EFER.LMA ビットを 1 にセットする。これは、IA-32e モードがイネーブルであるだけでなくアクティブでもあることを示す。
SCE	Syscall/Sysret イネーブル（ビット 0）	このビットを 1 にセットすると、SYSCALL/SYSRET のサポートがイネーブルになる。SYSCALL/SYSRET は、64 ビットモードでのみサポートされる。64 ビット動作用に SYSCALL/SYSRET をイネーブルにするのは、OS の役割である。 IA32_EFER 内の他のビットはすべて予約済みであり、0 が書き込まれていなければならない（MBZ）。

1.3.3.2. 制御レジスタ

インテル EM64T アーキテクチャでは、制御レジスタは次のように構成されている。

- 制御レジスタ CR0 ～ CR4 は、64 ビットに拡張されている。MOV CRn 命令は、64 ビットの読み取りまたは書き込みを実行する。オペランド・サイズ・プリフィックスは無視される。
- 互換モードとレガシーモードでは、制御レジスタの書き込みは、上位 32 ビットを 0 で埋める。制御レジスタの読み取りは、下位 32 ビットのみを返す。
- 64 ビットモードでは、CR0 と CR4 の上位 32 ビットは予約済みであり、0 が書き込まれていなければならない。CR0 または CR4 の上位 32 ビットに 0 でない値を書き込むと、一般保護例外 #GP(0) が発生する。CR2 の全 64 ビットは、ソフトウェアによって書き込み可能である。CR3 のビット 51:40 は予約済みであり、0 になっていなければならない。ただし、MOV CRn 命令は、CR2 または CR3 に書き込まれたアドレスがプロセッサのリニアアドレスまたは物理アドレスの制限の範囲内であるかどうかはチェックしない。
- タスク・プライオリティ・レジスタ（TPR）として定義される、新しい制御レジスタ CR8 が追加されている。オペレーティング・システムは、TPR を使用して、外

部割り込みの優先度レベルに基づいてその割り込みがプロセッサに割り込みをかけられるかどうかを制御できる。TPR についての詳細は、1.6.10.6. 節を参照のこと。

1.3.3.3. ディスクリプタ・テーブル・レジスタ

4 個のシステム・ディスクリプタ・テーブル・レジスタ（GDTR、IDTR、LDTR、TR）は、64 ビットのベースアドレスを格納するようにハードウェア内で拡張されている。これにより、IA-32e モードで動作するオペレーティング・システムは、プロセッサがサポートしているリニアアドレス空間内の任意の位置に、システム・ディスクリプタ・テーブルを配置できる。

表 1-5. は、GDTR と IDTR を示している。表 1-6. は、LDTR と TR を示している。いかなる場合にも、ベースアドレスは正規形式でなければならない。プロセッサがサポートしているリニア・アドレス・ビット数と物理アドレスビット数を確認するには、EAX を 80000008H に設定して CPUID 命令を実行する。

CPUID についての詳細は、第 2 章を参照のこと。

表 1-5. GDTR と IDTR

クワッドワード・オフセット	ビット 63:16	ビット 15:0
1		リミット
0	ベース	

表 1-6. LDTR と TR

クワッドワード・オフセット	ビット 63:20	ビット 19:16	ビット 15:0
3			セレクタ
2			属性
1		リミット	
0	ベース		

1.3.3.4. デバッグレジスタ

64 ビットモードでは、デバッグレジスタ DR0 ～ DR7 は 64 ビット・レジスタである。MOV DRn 命令は、レジスタの全 64 ビットの読み取りまたは書き込みを実行する。オペランド・サイズ・プリフィックスは無視される。

IA-32e プラットフォーム上のすべての 16 ビットモードまたは 32 ビットモード（レガシーモードまたは互換モード）では、デバッグレジスタへの書き込みは、上位 32 ビットを 0 で埋める。デバッグレジスタからの読み取りは、下位 32 ビットのみを返す。64 ビットモードでは、DR6 と DR7 の上位 32 ビットは予約済みであり、0 が書き込まれていなければならない。DR6 または DR7 の上位 32 ビットに 1 を書き込むと、#GP(0) 例外が発生する。

DR0 ～ DR3 の全 64 ビットは、ソフトウェアによって書き込み可能である。ただし、MOV DRn 命令は、DR0 ～ DR3 に書き込まれたアドレスがプロセッサのリニアアドレス制限の範囲内に入っているかをチェックしない。アドレスのマッチングは、プロセッサによって生成される有効なアドレスについてのみサポートされている。

1.4. 命令セットの変更

1.4.1. アドレス・サイズ・プリフィックスとオペランド・サイズ・プリフィックス

64 ビットモードでは、デフォルトのアドレスサイズは 64 ビットで、デフォルトのオペランド・サイズは 32 ビットである。デフォルト値は、新しい一連の命令プリフィックス（REX）を使用してオーバーライドできる。アドレス・サイズ・プリフィックスとオペランド・サイズ・プリフィックスにより、命令ごとにアドレスサイズとオペランド・サイズを切り替え、32 ビットと 64 ビットのデータとアドレスを共存させることができる。表 1-7. は、IA-32e 動作モードでのアドレスサイズのオーバーライド用の命令プリフィックスの必要条件を示している。

なお、64 ビットモードは、16 ビットアドレスをサポートしていない。互換モードとレガシーモードでは、アドレスサイズは従来の IA-32 アーキテクチャと同じように機能する。

表 1-7. IA-32e モードでのアドレスサイズのオーバーライドの必要条件

IA-32e モードのサブモード	デフォルトのアドレスサイズ（ビット）	実効アドレスサイズ（ビット）	アドレス・サイズ・プリフィックスの必要性
64 ビットモード	64	64	なし
		32	あり
互換モード	32	32	なし
		16	あり
	16	32	あり
		16	なし

表 1-8. は、IA-32e 動作モードでオペランド・サイズのオーバーライドの指定に使用できる、66H 命令プリフィックスと REX.W プリフィックスの有効な組み合わせを示している。

64 ビットモードでは、デフォルトのオペランド・サイズは 32 ビットである。REX プリフィックスには、異なる 16 の値を指定できる 4 ビット・フィールドが含まれている。REX プリフィックス内の W ビット・フィールドは、REX.W と呼ばれる。REX.W = 1 のプリフィックスは、64 ビットのオペランド・サイズを指定する。この場合でも、ソフトウェアはオペランド・サイズ・プリフィックス 66H を使用して、16 ビットのオペ

ランド・サイズに切り替えられる。ただし、REX.W = 1プリフィックスとオペランド・サイズ・プリフィックス（66H）の両方が使用されている場合は、REX.W = 1 プリフィックスが優先する。

SSE/SSE2/SSE3 の SIMD 命令の場合は、66H、F2H、F3H プリフィックスはオペコード拡張として使用され、オペコードの一部と見なされる。この場合は、有効な REX.W プリフィックスと 66H オペコード拡張プリフィックスの間に相互作用は存在しない。

表 1-8. 64 ビット拡張技術のオペランド・サイズのオーバーライド

IA-32e サブモード	デフォルトのオペランド・サイズ（ビット）	実効オペランド・サイズ（ビット）	命令プリフィックス	
			66H	REX.W = 1
64 ビットモード	32	64	X	必要
		32	不要	不要
		16	必要	不要
互換モード	32	32	不要	使用不可
		16	必要	
	16	32	必要	
		16	不要	
x: 機能は個々の命令の実装手法によって異なる。				

1.4.2. REX プリフィックス

REX プリフィックスは、64 ビットモードで使用される一連の新しい命令プリフィックス・バイトである。このプリフィックスは、以下の機能を持つ。

- 新しい GPR と SSE レジスタを指定する。
- 64 ビットのオペランド・サイズを指定する。
- （システム・ソフトウェアが使用する）拡張された制御レジスタを指定する。

すべての命令が REX プリフィックスを必要とするわけではない。命令が、拡張されたレジスタのうち 1 つを参照するか、または 64 ビット・オペランドを使用する場合にのみ、REX プリフィックスが必要になる。REX プリフィックスが無意味な状況で REX プリフィックスを使用した場合、プリフィックスは無視される。

1 つの命令で使用できる REX プリフィックスは 1 つだけである。REX プリフィックスを使用する場合は、オペコード・バイトまたは 2 バイトのオペコード・エスケープ・プリフィックス（存在する場合）の直前に置かなければならない。他の位置に置かれた REX プリフィックスは無視される。

命令サイズの 15 バイトのリミットは、REX プリフィックスを含む命令にも適用される。図 1-1. は、命令のバイトオーダ内の REX プリフィックスの位置を示している。

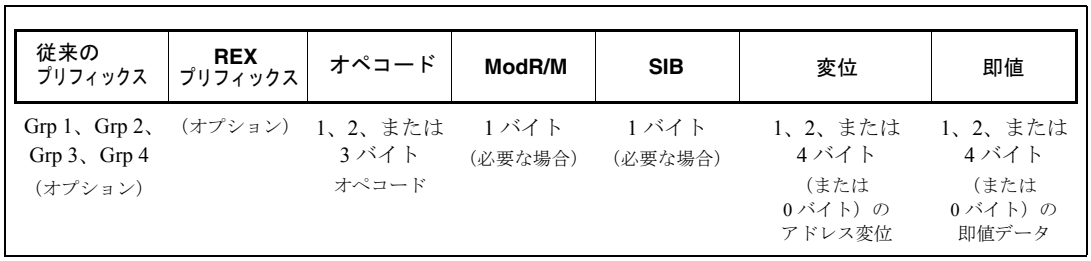

図 1-1. 64 ビットモードでのプリフィックスの順序

図 1-1. に示した従来のプリフィックスには、66H、67H、F2H、F3H が含まれる。グループ 1、グループ 2、グループ 3、グループ 4 のプリフィックスについては、『IA-32 インテル® アーキテクチャ・ソフトウェア・デベロッパーズ・マニュアル、中巻 A』の 2.2 節を参照のこと。

1.4.2.1. エンコーディング

IA-32 命令フォーマットは、以下のフォーマットに基づいて、命令エンコーディング内の 3 ビット・フィールドを使用して最大 3 個のレジスタを指定する。

- ModRM: ModRM バイトの reg フィールドと r/m フィールド
- ModRM と SIB: ModRM バイトの reg フィールドと、SIB（scale、index、base）バイトの base フィールドと index フィールド
- ModRM なしの命令 : オペコードの reg フィールド

64 ビットモードでは、これらのフィールドとフォーマットは変更されていない。64 ビット用のフィールドの拡張に必要なすべてのビットは、REX プリフィックスの追加によって提供される。

1.4.2.2. REX プリフィックスのフィールド

REX プリフィックスは、オペコード・マップの 1 行にわたってエントリ 40H ～ 4FH を占める、一連の 16 個のオペコードである。従来の IA-32 動作モードと互換モードでは、これらのオペコードは有効な命令（INC または DEC）を表す。64 ビットモードでは、これらのオペコードは命令プリフィックス REX を表し、個別の命令としては扱われない。

1 バイト・オペコードの INC/DEC 命令の機能は、64 ビットモードでは利用できなくなった。64 ビットモードでは、INC/DEC 機能は、同じ命令の ModRM 形式（オペコード FF/0 と FF/1）で利用できる。表 1-9. と図 1-2. ～図 1-5. は、REX プリフィックスの

フィールドとその使用法を示している。REXプリフィックスのフィールドの組み合わせによっては、操作は無効になる。このような場合、REXプリフィックスは無視される。

図 1-2. ～図 1-5. の 4 つの例は、REX プリフィックスの R、X、B ビットと、ModRM バイト、SIB バイト、オペコードのフィールドを組み合わせて、レジスタとメモリのアドレスを指定する方法を示している。R、X、B ビットについては、表 1-9. で説明している。

表 1-9. REX プリフィックスのフィールド

フィールド名	ビット位置	定義
-	7:4	0100
W	3	0 = デフォルトのオペランド・サイズ
		1 = 64 ビットのオペランド・サイズ
R	2	ModRM の reg フィールドの拡張
X	1	SIB の index フィールドの拡張
B	0	ModRM の r/m フィールド、SIB の base フィールド、またはオペコードの reg フィールドの拡張

以下の追加情報に注意すること。

- REX.W ビットをセットしてオペランド・サイズを指定できるが、これだけではオペランドの幅は決まらない。既存の 66H オペランド・サイズ・プリフィックスと同じように、REX による 64 ビット・オペランド・サイズのオーバーライドは、バイト固有の操作については無効である。
- 非バイト操作の場合、REX オペランド・サイズ・オーバーライドは、66H プリフィックスに優先する。66H プリフィックスと REX プリフィックス（REX.W = 1）を組み合わせて使用した場合、66H プリフィックスは無視される。66H オーバーライドと REX（REX.W = 0）を組み合わせて使用した場合、オペランド・サイズは 16 ビットになる。
- ModRM の reg フィールドが GPR、SSE レジスタ、制御レジスタ、またはデバッグレジスタをエンコードしている場合、REX.R は ModRM の reg フィールドを修正する。ModRM が他のレジスタを指定するか、または拡張されたオペコードを格納している場合は、REX.R は無視される。
- REX.X ビットは、SIB の index フィールドを修正する。
- REX.B は、ModRM の r/m フィールドまたは SIB の base フィールド内のベースを修正する。あるいは、GPR へのアクセスに使用される、オペコードの reg フィールドを修正する。

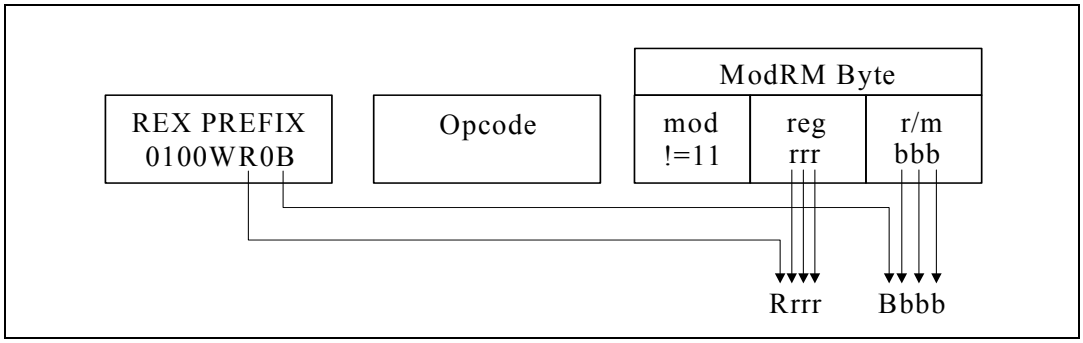

図 1-2. メモリアドレス指定
（SIB バイトなし、REX.X は使用しない、ModRM の reg フィールドを使用）

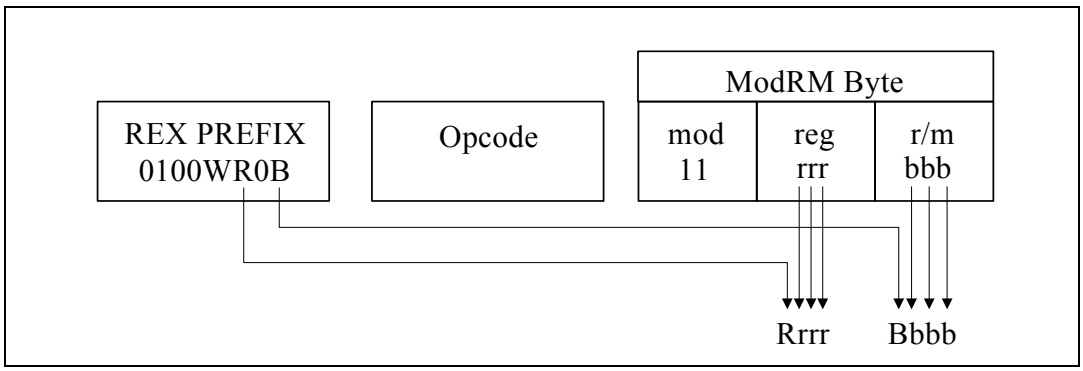

図 1-3. レジスタ・レジスタのアドレス指定（メモリ・オペランドなし）、REX.X は使用しない

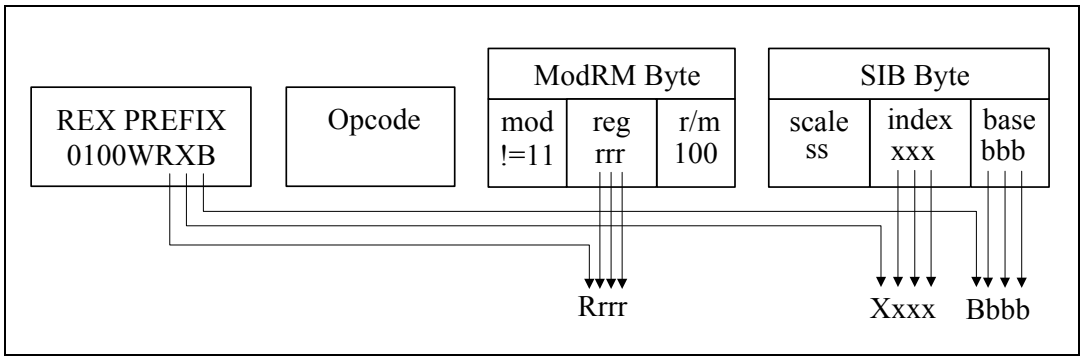

図 1-4. SIB バイトを使用したメモリアドレス指定

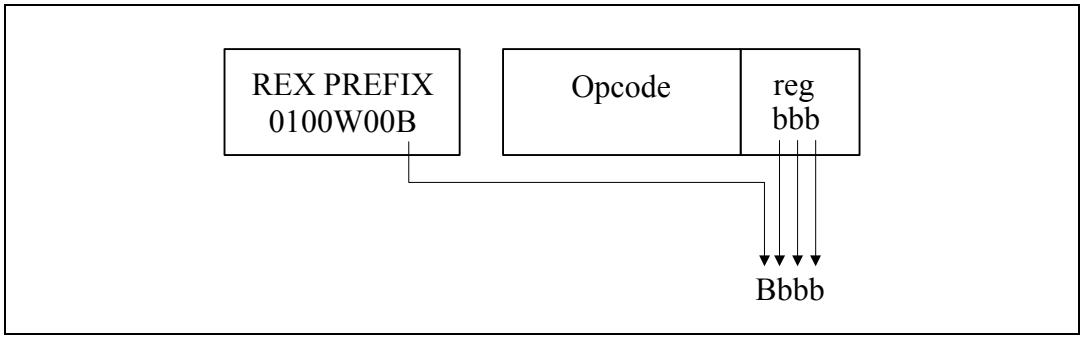

図 1-5. オペコード・バイト内にコード化されたレジスタ・オペランド
（REX.X と REX.R は使用しない）

従来の IA-32 アーキテクチャでは、バイトレジスタ（AH、AL、BH、BL、CH、CL、DH、DL）は、ModRM バイトの reg フィールド、r/m フィールド、またはオペコードの reg フィールド内で、レジスタ 0 ～ 7 としてエンコードされる。REX プリフィックスは、バイトレジスタに追加のアドレス指定機能を提供し、GPR の最下位バイトをバイト操作に利用できるようにする。

レジスタのエンコーディングでは、ModRM バイトと SIB バイトのフィールドの特定の組み合わせが特殊な意味を持つ。組み合わせによっては、REX プリフィックスによって拡張される命令フィールドがデコードされない。表 1-10. は、各条件での動作を示している。

表 1-10. REX エンコーディングの特殊な条件

ModRM または SIB	サブフィールドのエンコーディング	互換モードの動作	互換モードでの影響	その他の影響
ModRM バイト	mod != 11 r/m == b*100(ESP)	SIB バイトあり	ESPベースのアドレス指定に SIB バイトが必要。	REX プリフィックスは第 4 のビット（b）を追加するが、このビットはデコードされない。 R12 ベースのアドレス指定にも SIB バイトが必要。
ModRM バイト	mod == 0 r/m == b*101(EBP)	ベース・レジスタは使用しない	変位 0 の mod = 01 を使用して、変位なしのEBPを使用しなければならない。	REX プリフィックスは第 4 のビット（b）を追加するが、このビットはデコードされない。 変位 0 の mod = 01 を使用して、変位なしの RBP または R13 を使用しなければならない。

表 1-10. REX エンコーディングの特殊な条件（続き）

ModRM または SIB	サブフィールドのエンコーディング	互換モードの動作	互換モードでの影響	その他の影響
SIB バイト	index == 0100(ESP)	インデックス・レジスタは使用しない	ESP はインデックス・レジスタとして使用できない。	REX プリフィックスは第 4 のビット（b）を追加する。このビットはデコードされる。 その他の影響はない。拡張されたindexフィールドにより、RSP と R12 を区別できるため、R12 をインデックスとして使用できる。
SIB バイト	base == 0101(EBP)	mod = 0 の場合、ベースレジスタは使用しない	ベースレジスタは modのエンコーディングに依存する。	REX プリフィックスは第 4 のビット（b）を追加する。このビットはデコードされる。 その他の影響はない。拡張されたbaseフィールドを使用してRBPとR13を区別できるため、mod に関係なく、R13 を SIB の base として使用できる。
* b の値（REX.B）は無視される。				

1.4.2.3. 変位

64 ビットモードのアドレス指定は、既存の 32 ビットの ModRM エンコーディングと SIB エンコーディングを使用する。特に、ModRM と SIB の変位のサイズは変更されていない。変位は 8 ビットまたは 32 ビットのままであり、64 ビットに符号拡張される。

1.4.2.4. 直接メモリ・オフセット MOV

64 ビットモードでは、MOV 命令の直接メモリ・オフセット形式（表 1-11）が、64 ビット即値絶対アドレスを指定するように拡張される。このアドレスは、moffset と呼ばれる。この 64 ビット・メモリ・オフセットを指定するのに、プリフィックスは不要である。これらの MOV 命令の場合、メモリ・オフセットのサイズは、アドレスサイズのデフォルト値（64 ビットモードでは 64 ビット）に従う。

表 1-11. MOV の直接メモリ・オフセット形式

オペコード	命令
A0	MOV AL, moffset
A1	MOV EAX, moffset
A2	MOV moffset, AL
A3	MOV moffset, EAX

1.4.2.5. 即値

64 ビットモードでは、即値オペランドの標準サイズは 32 ビットのままである。オペランド・サイズが 64 ビットの場合、プロセッサは、即値を使用する前に、すべての即値を 64 ビットに符号拡張する。

64 ビット即値オペランドは、既存の移動命令（MOV reg, imm16/32）の語彙を拡張することによってサポートされる。これらの命令（オペコード B8H ～ BFH）は、（実効オペランド・サイズに基づいて）16 ビットまたは 32 ビットの即値データを GPR 内に移動する。実効オペランド・サイズが 64 ビットの場合、これらの命令を使用して、GPR に即値をロードできる。32 ビットのデフォルト・オペランド・サイズを 64 ビットのオペランド・サイズにオーバーライドするには、REX プリフィックスが必要である。

例えば、次の命令を使用する。

```
48 B8 8877665544332211 MOV RAX,1122334455667788H
```

1.4.2.6. RIP 相対アドレス指定

64 ビットモードには、新しいアドレス指定形式である RIP 相対（相対命令ポインタ）アドレス指定が実装されている。実効アドレスは、次の命令の 64 ビット RIP に変位を加算することによって得られる。

従来の IA-32 アーキテクチャでは、命令ポインタを基準とするアドレス指定は、制御移行命令でのみ利用可能である。64 ビットモードでは、ModRM アドレス指定を使用する命令は、RIP 相対アドレス指定を使用できる。RIP 相対アドレス指定がない場合、すべての ModRM 命令モードは、0 を基準としてメモリにアクセスする。

RIP 相対アドレス指定により、特定の ModRM モードは、符号付き 32 ビット変位を使用して、64 ビット RIP を基準としてメモリにアクセスできる。これにより、RIP から ±2GB のオフセット範囲が得られる。表 1-12. は、RIP 相対アドレス指定に関する ModRM と SIB のエンコーディングを示している。現在の ModRM と SIB のエンコーディングには、32 ビット変位アドレス指定の冗長形式が存在する。ModRM エンコーディングは 1 種類、SIB エンコーディングは複数存在する。RIP 相対アドレス指定は、冗長形式を使用してエンコードされる。

64 ビットモードでは、ModRM Disp32（32 ビット変位）のエンコーディングは、変位だけでなく、RIP+Disp32 になるように再定義されている。表 1-12. を参照のこと。

表 1-12. RIP 相対アドレス指定

ModRM と SIB のサブフィールドのエンコーディング		互換モードの動作	64 ビットモードの動作	64 ビットモードでのその他の影響
ModRM バイト	mod == 00	Disp32	RIP + Disp32	通常の（0 を基準とする）変位アドレス指定の SIB 形式を使用しなければならない。
	r/m == 101（なし）			
SIB バイト	base == 101（なし）	mod = 00 の場合は、Disp32	レガシーモードと同じ	なし
	index == 100（なし）			
	scale = 0、1、2、4			

RIP 相対アドレス指定に関する ModRM のエンコーディングは、REX プリフィックスの使用に依存しない。具体的には、RIP 相対アドレスの選択に使用される、r/m ビット・フィールドのエンコーディング 101B は、REX プリフィックスの影響を受けない。例えば、mod = 00B に設定して R13（REX.B = 1、r/m = 101B）を選択した場合でも、RIP 相対アドレス指定が使用される。ModRM と組み合わされる REX.B の 4 ビット r/m フィールドは、完全にはデコードされない。ソフトウェアは、変位なしで R13 をアドレス指定するには、1 バイトの変位 0 を使用して R13 + 0 としてアドレスをエンコードしなければならない。

RIP 相対アドレス指定は、64 ビットのアドレスサイズによってではなく、64 ビットモードによってイネーブルにされる。アドレス・サイズ・プリフィックスを使用しても、RIP 相対アドレス指定はディスエーブルにならない。アドレス・サイズ・プリフィックスの影響は、計算された実効アドレスが切り捨てられるか、または 0 で拡張されて、32 ビットに変換されることである。

1.4.2.7. デフォルトの 64 ビット・オペランド・サイズ

64 ビットモードでは、2 つのグループの命令が、64 ビットのデフォルト・オペランド・サイズを使用する（このオペランド・サイズを指定する REX プリフィックスは不要である）。これらの命令には、以下のものがある。

- near 分岐
- 暗黙的に RSP を参照するすべての命令（far 分岐を除く）

1.4.3. 制御レジスタとデバッグレジスタの新しいエンコーディング

64 ビットモードでは、制御レジスタとデバッグレジスタの新しいエンコーディングを利用できる。ModRM の reg フィールドが制御レジスタまたはデバッグレジスタをエン

コードする場合は、REX.R ビットを使用して ModRM の reg フィールドを修正できる（表 1-9. を参照）。これらのエンコーディングにより、プロセッサは、CR8 ～ CR15 と DR8 ～ DR15 にアクセスできる。

64 ビットモードでは、制御レジスタ（CR8）が追加定義されている。CR8 は、タスク・プライオリティ・レジスタ（TPR）になる。IA-32e テクノロジを搭載した最初のプロセッサ・モデルには、CR9 ～ CR15 と DR8 ～ DR15 は実装されていない。実装されていないレジスタにアクセスしようとすると、無効オペコード例外（#UD）が発生する。

1.4.4. 新しい命令

64 ビットの拡張技術により、64 ビットモードに以下の新しい命令が導入されている。これらの命令については、第 2 章で詳しく説明する。

- SWAPGS
- SYSCALL と SYSRET
- CDQE
- CMPSQ
- CMPXCHG16B
- LODSQ
- MOVSQ
- MOVZX（64 ビット）
- STOSQ

1.4.5. スタックポインタ

64 ビットモードでは、スタックポインタのサイズは 64 ビットである。このスタックサイズは、（互換モードやレガシーモードとは異なり）SS ディスクリプタ内のビットによって制御されることはない。また、命令プリフィックスによってオーバーライドもできない。

暗黙的なスタック参照では、アドレスサイズのオーバーライドは無視される。64 ビットモードでは、暗黙的に RSP を参照するすべての命令（far 分岐を除く）のオペランド・サイズは、デフォルトにより 64 ビットになる。この規則の影響を受ける命令には、PUSH、POP、PUSHF、POPF、ENTER、LEAVE が含まれる。64 ビットモードでは、これらの命令を使用して、スタック上に 32 ビット値をプッシュ / ポップできない。16 ビットのプッシュとポップは、66H オペランド・サイズ・プリフィックスの使用によって可能である。

デフォルトのオペランド・サイズは64ビットであるため、レジスタRAX～RSPをオペランドとして使用する場合、これらの命令の先頭のREXプリフィックスは不要である。ただし、R8～R15レジスタを使用する場合は、REXは必要である。これは、新しい拡張されたレジスタのアドレス指定に、REXプリフィックスが必要だからである。

1.4.6. 分岐

64ビットの拡張技術は、2つの分岐機構を拡張して、64ビットのリニアアドレス空間内の分岐に対応している。拡張された分岐機構は、以下の2つである。

- 64ビットモードでのnear分岐の再定義
- 64ビットモードと互換モードでのfarコール用の64ビット・コールゲート・ディスクリプタ

64ビットモードでは、すべてのnear分岐命令（CALL、RET、JCC、JCXZ、JMP、LOOP）のオペランド・サイズは、強制的に64ビットに設定される。これらの命令は、REXオペランド・サイズ・プリフィックスを必要とせずに、64ビットRIPを更新する。near分岐の以下の要素は、実効オペランド・サイズによって制御される。

- 命令ポインタのサイズの切り捨て
- CALLまたはRETを原因とする、スタックのポップまたはプッシュのサイズ
- CALLまたはRETを原因とする、スタックポインタのインクリメントまたはデクリメントのサイズ
- 間接分岐のオペランド・サイズ

64ビットモードでは、上記のすべての動作は、オペランド・サイズ・プリフィックスに関係なく、強制的に64ビットに設定される（オペランド・サイズ・プリフィックスは、何も反応なしで無視される）。ただし、64ビットモードでは、相対分岐の変位フィールドは32ビットまでに制限され、near分岐のアドレスサイズは強制的に設定されない。

アドレスサイズは、JCXZとLOOPに使用されるRCXのサイズに影響を与える。また、アドレスサイズは、メモリ間接分岐のアドレス計算にも影響を与える。メモリ間接分岐のアドレスは、デフォルトでは64ビットであるが、アドレス・サイズ・プリフィックスによって32ビットにオーバーライドできる。

ソフトウェアは、通常はfar分岐を使用して特権レベルを変更する。従来のIA-32アーキテクチャは、ソフトウェアがある特権レベルから別の特権レベルに分岐できるように、コールゲート機構を用意している（コールゲートは、特権レベルを変更しない分岐にも使用される）。コールゲートを使用する場合、直接ポインタまたは間接ポイン

タのセレクタ部分はゲート・ディスクリプタを参照する（この命令内のオフセットは無視される）。デスティネーションのコード・セグメントへのオフセットは、コールゲート・ディスクリプタから得られる。IA-32e モードでは、32 ビット・コールゲート・ディスクリプタのタイプ値が再定義されて 64 ビット・コールゲート・ディスクリプタになり、64 ビット・ディスクリプタのサイズは 64 ビット・オフセットを格納できるように拡張される。64 ビットモードのコールゲート・ディスクリプタにより、far 分岐は、サポートしているリニアアドレス空間内のすべての位置を参照できる。これらのコールゲートは、ターゲット・コード・セレクタ（CS）も格納し、ゲート移行の結果として特権レベルとデフォルトサイズの変更を可能にする。

即値は一般的に最大 32 ビットで指定されるため、64 ビットモードでフル 64 ビットの絶対 RIP を指定する方法は、間接分岐を使用する以外にない。この理由で、64 ビットモードの命令セットには、直接 far 分岐は含まれていない。

IA-32e モードでは、SYSENTER 命令と SYSEXIT 命令が、64 ビットメモリ空間内で動作するように拡張されている。この拡張についての詳細は、第 3 章の「SYSENTER - Fast System Call」と「SYSEXIT - Fast Return from Fast System Call」を参照のこと。また、IA-32e モードには、SYSCALL と SYSRET の 2 つの新しい命令が導入されている。SYSCALL 命令と SYSRET 命令は、64 ビットモードでのみ有効である。第 3 章の「SYSCALL - Fast System Call」と「SYSRET - Return From Fast System Call」を参照のこと。

1.5. メモリの構成

1.5.1. 64 ビットモードのアドレス計算

64 ビットモードでは（アドレスサイズ・オーバーライドがない場合）、実効アドレス計算のサイズは 64 ビットになる。実効アドレス計算は、64 ビット・ベース / インデックス・レジスタと、64 ビットに符号拡張された変位を使用する。

64 ビットモードでは、フラットなアドレス空間を使用するため、リニアアドレスは実効アドレスに等しくなる。ただし、0 でないベースと組み合わせて FS セグメントまたは GS セグメントを使用する場合は、この規則は適用されない。64 ビットモードでは、フル 64 ビットのセグメント・ベースを加算する前に、実効アドレス成分が加算され、実効アドレスが切り捨てられる。64 ビットモードでは、アドレス指定モードに関係なく、セグメント・ベースは切り捨てられない。

IA-32eモードでは、64ビットのコード・オフセットをサポートするために、命令ポインタが64ビットに拡張される。この64ビット命令ポインタは、RIPと呼ばれる。表1-13.は、RIP、EIP、IPの間の関係を示している。

表 1-13. 命令ポインタ

クワッドワード・オフセット	ビット 63:32	ビット 31:16	ビット 15:0
2	変更なし		IP
1	0 で拡張	EIP	
0	RIP		

一般的に、64ビットモードでは、変位と即値は64ビットに拡張されない。64ビットモードでも、変位と即値は32ビットまでに制限され、実効アドレス計算の際に符号拡張される。ただし、64ビットモードでは、MOV命令については64ビット形式の変位と即値がサポートされている。

IA-32eモードでは、すべての16ビット/32ビットアドレス計算は0で拡張され、64ビットアドレスに変換される。アドレス計算は、まず、任意のアドレス・サイズ・プリフィックスによってオーバーライドされた、現在のモード（64ビットモードまたは互換モード）の実効アドレスサイズに合わせて切り捨てられる。次に、その結果は、フル64ビットのアドレスサイズに合わせて0で拡張される。このため、互換モードで動作する16ビット・アプリケーションと32ビット・アプリケーションは、64ビットモードの実効アドレスの下位4GBにのみアクセスできる。同じように、64ビットモードで生成される32ビットアドレスも、64ビットモードの実効アドレスの下位4GBにのみアクセスできる。

1.5.2. 正規のアドレス指定

正規形式と見なされるアドレスは、アドレスビット63から、マイクロアーキテクチャが実装している最上位ビットまでのビットが、すべて1またはすべて0にセットされている。

IA-32eモードは、64ビットのリニアアドレスを定義している。ただし、プロセッサ・モデルによっては、サポートしているリニアアドレスが64ビットより小さい場合がある。インテル® EM64Tを搭載した最初のIA-32プロセッサは、48ビットのリニアアドレスをサポートする。この場合、正規のアドレスは、（ビット47が0か1かに基づいて）ビット63～48が0または1にセットされていなければならない。

プロセッサ・モデルによっては、64ビットのリニアアドレス全体を使用しないときもあるが、これらのプロセッサも、ビット63から実装されている最上位ビットまでをチェックして、アドレスが正規形式になっているかを確認する必要がある。リニアメモリ参照が正規形式になっていない場合は、プロセッサは例外を生成する必要があ

る。通常は一般保護例外（#GP）が生成されるが、明示的または暗黙的なスタック参照の場合は、スタックフォルト（#SS）が生成される。

暗黙的なスタック参照を含む命令は、デフォルトにより、SSセグメント・レジスタを使用する。これらの命令には、PUSH/POP に関連する命令と、ベースレジスタとしてRSP/RBP を使用する命令が含まれる。これらの場合、正規形式に関するフォルトは#SF になる。ベースレジスタとしてRSP/RBP を使用する命令が、SS でないセグメントを指定するセグメント・オーバライド・プリフィックスを持つ場合は、正規形式に関するフォルトによって#GP が発生する。正規のアドレス形式かどうかのチェックは、特権チェックの実行後、ページング・チェックとアライメント・チェックの前に実行される。

1.6. オペレーティング・システムに関する注意事項

1.6.1. CPUID 命令

CPUID 命令は、プロセッサの機能の有無を報告する。オペレーティング・システムは、IA-32 モード、IA32_EFER MSR、64 ビットモードで利用できる命令について、CPUIDを使用して利用可能かどうかを確認しなければならない。IA-32eモードに関連する機能については、2.2. 節のCPUIDのリファレンス・ページを参照のこと。

1.6.2. レジスタの設定値と IA-32e モード

64ビットモードと互換モードの動作は、IA32_EFER MSR とCSディスクリプタの各種の制御ビットによって制御される。表1-14. は、IA-32eモードと従来のIA-32モードの制御ビットの設定値を示している。この表は、レガシーモードとIA-32e モードのデフォルトのアドレスサイズとデータサイズも示している。SMM とレジスタの拡張についての詳細は、表1-1. を参照のこと。

表 1-14. プロセッサ・モード

モード		エンコーディング			デフォルトのアドレスサイズ	デフォルトのオペランド・サイズ
		IA32_EFER.LMA	CS.L	CS.D		
レガシーモード		0	使用不可	1	32	32
				0	16	16
IA-32e モード	64 ビットモード	1	1	0	64	32
	互換モード		0	1	32	32
				0	16	16

1.6.3. プロセッサ・モード

1.6.3.1. IA-32e モード

IA-32eモードは、コード・セグメント・ディスクリプタ内の2ビット（CS.LとCS.D、表1-14.を参照）を使用して、サブ動作モードを制御する。プロセッサが64ビットモードで動作している場合は、CS.L = 1、CS.D = 0 である。この設定では、デフォルトのオペランド・サイズは32ビットで、デフォルトのアドレスサイズは64ビットになる。命令プリフィックスを使用して、オペランド・サイズを 64 ビットまたは 16 ビットに変更できる。アドレスサイズは32ビットに変更できる。

IA-32eモードがアクティブでCS.L = 0の場合は、プロセッサは互換モードになる。互換モードでは、従来のIA-32アーキテクチャと全く同じように、CS.Dがデフォルトのオペランド・サイズとアドレスサイズを制御する。CS.D = 1にセットすると、デフォルトのオペランド・サイズとアドレスサイズは 32 ビットに設定される。CS.D を 0 にクリアすると、デフォルトのオペランド・サイズとアドレスサイズは16ビットに設定される。

CS.L = 1 と CS.D = 1 の組み合わせは、将来に備えて予約されている。

1.6.3.2. IA-32e モードのアクティブ化

オペレーティング・システムは、以下の手順に従って、IA-32eモードをアクセスにする必要がある。

1. まず、ページングがイネーブルにされた保護モードから、CR0.PG = 0 に設定してページングをディスエーブルにする。ページングのディスエーブルには、MOV CR0 命令を使用する（この命令は、ID がマッピングされたページ内に置かれていなければならない）。
2. CR4.PAE = 1 にセットして、物理アドレス拡張機構をイネーブルにする。PAE をイネーブルにしないと、IA-32e モードをイネーブルにしようとしたときに #GP フォルトが発生する。
3. レベル 4 ページ・マップ・テーブル（PML4）の物理ベースアドレスを CR3 にロードする。
4. IA32_EFER.LME = 1 にセットして、IA-32e モードをイネーブルにする。
5. CR0.PG = 1 にセットして、ページングをイネーブルにする。これで、プロセッサは、LMA ビットを 1 にセットする。ページングをイネーブルにする MOV CR0 命令とそれに続く命令は、ID がマッピングされていないページへの分岐が実行可能になるまでは、ID がマッピングされたページ内に置かれていなければならない。

IA-32eモードから従来のページング保護モードに戻るには、以下の手順に従って、IA-32eモードを非アクティブにし、ディスエーブルにする。

1. 互換サブモードでなければならない。
2. CR0.PG = 0 にクリアして、IA-32e モードを非アクティブにする。これで、プロセッサは、IA32_EFER.LMA = 0 に設定する。ページングのディスエーブルに使用される MOV CR0 命令とそれに続く命令は、ID がマッピングされたページ内に置かれていなければならない。
3. 従来のページ・テーブル・ディレクトリのベースアドレスの物理ベースアドレスを CR3 にロードする。
4. IA32_EFER.LME = 0 に設定して、IA-32e モードをディスエーブルにする。
5. CR0.PG = 1 に設定して、従来のページング保護モードをイネーブルにする。
6. ページングをイネーブルにする MOV CR0 命令の直後に、分岐命令を実行しなければならない。MOV CR0 と分岐命令のいずれも、ID がマッピングされたページ内に置かれていなければならない。

IA-32e モードをアクティブにした直後、システム・ディスクリプタ・テーブル・レジスタ（GDTR、LDTR、IDTR、TR）は、従来のディスクリプタ・テーブルを参照し続ける。これらのディスクリプタによって参照されるテーブルは、すべてリニアアドレス空間の下位 4GB 内に置かれる。64 ビット・オペレーティング・システムは、IA-32e モードをアクティブにした後、LGDT、LLDT、LIDT、LTR 命令を使用して、64 ビット・ディスクリプタ・テーブルへの参照をシステム・ディスクリプタ・テーブル・レジスタにロードする必要がある。

ソフトウェアは、IA-32e モードがアクティブになってから、割り込みディスクリプタ・テーブル・レジスタ（IDTR）の次の更新によって 64 ビット割り込みディスクリプタ・テーブル（IDT）への参照が設定されるまでの間、例外や割り込みを発生させてはならない。これは、IA-32e モードがアクティブになった直後は、IDT が従来の形式のままになっているからである。IDTR の更新より前に割り込みや例外が発生すると、従来の 32 ビット割り込みゲートが参照されて、64 ビット割り込みゲートとして解釈され、予測不可能な結果が発生する。外部割り込みは、CLI 命令によってディスエーブルにできる。マスク不可割り込み（NMI）は、外部ハードウェアを使用してディスエーブルにしなければならない。

IA-32e モードをアクティブにする前に、64 ビットモードのページング・テーブルが、物理アドレス空間の最初の 4GB 内に置かれていなければならない。これは、IA-32e モードをアクティブにする前に、ページ・ディレクトリ・ベースの初期化に使用される MOV CR3 命令をレガシーモードで実行しなければならないからである（ページングをイネーブルにするには、CR0.PG = 1 にセットする）。MOV CR3 はレガシーモードで実行されるため、CR3 レジスタの下位 32 ビットだけが書き込まれ、テーブルの位置はメモリの下位 4GB に制限される。IA-32e モードがアクティブになった後、ソフトウェアは、物理メモリ内の任意の位置にページテーブルを再配置できる。

ソフトウェアが、IA-32e モードのアクティブ化に直接関連するイネーブル・ビット（IA32_EFER.LME、CR0.PG、CR4.PAE）を変更しようとするたびに、プロセッサは 64 ビットモードの整合性チェックを実行する。整合性チェックが不合格になった場合、プロセッサは一般保護例外（#GP）を生成する。64 ビットモードの整合性チェックは、プロセッサが未定義のモードやステートに移行して予測不可能な動作をすることを防止する。

64 ビットモードの整合性チェックは、以下の場合に不適合になる。

- ページングがイネーブルになっているとき、IA-32eモードをイネーブルまたはディスエーブルにしようとした。
- IA-32e モードがイネーブルになっているとき、物理アドレス拡張機構（PAE）をイネーブルにする前にページングをイネーブルにしようとした。
- IA-32e モードがアクティブになっているとき、物理アドレス拡張機構（PAE）をディスエーブルにしようとした。
- IA-32eモードをアクティブにしようとしたとき、現在のCSのLビットがセットされていた。
- TR には 16 ビット TSS が格納されていなければならない。

表 1-15. は、64 ビットモードの整合性チェックについてまとめたものである。

表 1-15. IA-32e モードの整合性チェック

レジスタ	ビット	チェック
EFER	LME 0 → 1	if (CR0.PG ==1) then #GP(0)
	LME 1 → 0	if (CR0.PG == 1) then #GP(0)
CR0	PG 0 → 1	if ((IA32_EFER.LME == 1) & (CR4.PAE-0)) then #GP(0)
CR4	PAE 1 → 0	if (IA32_EFER.LMA == 1) then #GP(0)

1.6.3.3. 仮想 8086 モード

プロセッサが IA-32e モードで動作しているときは、仮想 8086 モードはサポートされない。IA-32e モードがイネーブルになっているとき、EFLAGS.VM ビットをセットしようとすると、何も反応なしで無視される。

1.6.3.4. 互換モード

IA-32e モード内の互換モードは、従来の 16 ビット /32 ビットの IA-32 アプリケーションとのバイナリ互換性を維持している。従来の 16 ビットまたは 32 ビット・アプリケーションのうち、仮想 8086 モードで動作するものとハードウェア・タスク管理を使用するものは、互換モードではサポートしていない。

互換モードの実行は、コード・セグメントごとに行われる。これにより、従来の16ビット/32ビット・アプリケーションと、64ビットモードで動作する64ビット・アプリケーションは、64ビット・オペレーティング・システム上で共存できる。IA-32eモードで動作するオペレーティング・システムは、コード・セグメント・ディスクリプタのCS.Lビットを0にクリアして、既存の16ビット/32ビット・アプリケーションを実行できる。

CS.L = 0の場合、CS.Dビット、アドレス・サイズ・プリフィックス、オペランド・サイズ・プリフィックスについては、従来のIA-32の意味が保たれる。また、セグメント化がイネーブルになる。IA-32eモードがアクティブであっても、アプリケーションの観点から見ると、プロセッサは（CS.Dの値に基づいて）従来の16ビットまたは32ビット・オペレーティング環境で動作している。

互換モードでは、IA-32eモードのアーキテクチャ上の定義によって、以下のシステムレベルのメカニズムが機能する。

- リニアアドレスから物理アドレスへの変換には、64ビットモードの拡張されたページ変換機構が使用される。
- 割り込みと例外の処理には、64ビットモードのメカニズムが使用される。
- システムコール（コールゲートを介した呼び出しとSYSENTER/SYSEXIT）の処理には、IA-32eモードのメカニズムが使用される。

1.6.4. セグメント化

IA-32eモードでは、セグメント化の影響は、プロセッサが互換モードで動作しているか、64ビットモードで動作しているかによって異なる。互換モードでは、セグメント化は、従来の16ビットまたは32ビットの保護モードの仕組みを使用して、従来のIA-32モードと全く同じように機能する。

64ビットモードでは、セグメント化は、全般に（ただし、完全にではなく）ディスエーブルになり、フラットな64ビットのリニアアドレス空間が構成される。厳密には、64ビットモードでは、プロセッサはCS、DS、ES、SSのセグメント・ベースを0として扱い、実効アドレスに等しいリニアアドレスを作成する。ただし、FSセグメントとGSセグメントは、この規則の例外である。これらのセグメントのセグメント・レジスタ（セグメント・ベースを格納する）は、追加のベースレジスタとしてリニアアドレス計算に使用できる。これにより、ローカルデータとオペレーティング・システムの特定のデータ構造を簡単にアドレス指定できる。

なお、セグメント化が全般にディスエーブルになっていても、セグメント・レジスタのロードにより、プロセッサにセグメント・アクセスの支援を実行できる。また、セグメント・レジスタのロードでは、64ビットモードでは値が使用不可の場合でも、値に関する従来のチェックはすべて実行される。値のチェックが必要なのは、互換モー

ドでのアプリケーションによる使用のために、64 ビットモードでセグメント・レジスタに値をロードする場合があるからである。

1.6.4.1. コード・セグメント

64 ビットモードでは、コード・セグメント（CS）ディスクリプタの一部の内容（例えば、ベース・アドレス・フィールドとリミット・フィールド）は無視され、その他のフィールドは正常に機能する（ただし、タイプ・フィールドの読み取り可能ビットは正常に機能しない可能性がある）。コード・セグメントは、64 ビットモードでも存在する。プロセッサの動作モードや実行の特権レベルを設定するには、コード・セグメントとそれに関連するディスクリプタおよびセレクタが必要である。コード・セグメントの動作モードと特権レベルは、ロング（L）、デフォルト・オペレーション・サイズ（D）、ディスクリプタ特権レベル（DPL）で指定する。

64 ビットモードのアドレス計算では、セグメント・ベースは 0 であるかのように扱われる。IA-32e モードは、CS ディスクリプタ内の従来未使用のビットを使用する。ビット 53 は、ロング（L）ビットとして定義され、IA-32e モードがアクティブ（IA32_EFER.LMA = 1）になっているときの 64 ビットモードと互換モードの選択に使用される。

表 1-16. は、従来の CS ディスクリプタに L ビットが追加された状態を示している。

表 1-16. コード・セグメント・ディスクリプタ

	ビット位置										
DW オフセット	31:24	23	22	21	20	19:16	15	14:13	12	11:8	7:0
1	ベースアドレス 31:24	G	D	L	AVL	セグメント・リミット 19:16	P	DPL	1	タイプ	ベースアドレス 23:16
0	ベースアドレス 15:0						セグメント・リミット 15:0				

CS ディスクリプタの D ビットは、デフォルトのオペランド・サイズとアドレスサイズを選択する。CS.L ビットが 1 の場合、CS.D の有効な設定値は 0 だけである。この設定値は、32 ビットのデフォルト・オペランド・サイズと 64 ビットのデフォルト・アドレス・サイズに対応する。なお、CS.L = 1 と CS.D = 1 のビット組み合わせは、将来に備えて予約されている。IA-32e モードで、これらのビットがセットされたコード・セグメントを使用しようとすると、#GP フォルトが発生する。

IA-32e モードがアクティブで CS.L = 0 の場合は、プロセッサは互換モードで動作している。この場合、CS.D は、レガシーモードと同じように、データとアドレス両方のデフォルト・サイズを選択する。CS.D = 0 の場合は、デフォルトのデータサイズとアドレスサイズは 16 ビットになる。CS.D = 1 の場合は、32 ビットのデフォルト・データ・サイズとアドレスサイズが選択される。

IA-32e モードでは、レガシーモードと同じように、CS ディスクリプタの DPL は実行特権チェックに使用される。

1.6.4.2. セグメント LOAD 命令

IA-32e モードでは、セグメント・ロード命令とセグメント・ロード・レジスタについて、以下に示す動作を予想できる。

- ES、DS、SS セグメント・レジスタは使用されない。対応するセグメント・ディスクリプタ・レジスタ内のフィールド（ベース、リミット、属性）は無視される。
- セグメント・ロード命令の一部の形式も無効になる（例えば、LDS、POP ES）。
- 64 ビットモードでは、ES、DS、または SS セグメントを参照するアドレス計算は、セグメント・ベースが 0 であるかのように扱われる。SS DPL は、CPL に常に等しくなるように変更される。このことは、DPL が SS ディスクリプタ内で変更される唯一のフィールドである場合にも該当する。
- プロセッサは、リミットチェックを実行する代わりに、すべてのリニアアドレス参照が正規形式になっているかをチェックする。モードを切り替えても、セグメント・レジスタの内容やそれに関連するディスクリプタ・レジスタの内容は変更されない。また、これらのレジスタの内容は、明示的なセグメント・ロードが実行されない限り、64 ビットモードでの実行中に変更されない。
- アプリケーション用に互換モードの設定ができるように、セグメント・ロード命令（MOV Sreg、POP Sreg）は、64 ビットモードで正常に動作する。システム・ディスクリプタ・テーブル（GDT または LDT）から 1 つのエントリが読み取られ、セグメント・ディスクリプタ・レジスタの隠れた部分にロードされる。ディスクリプタ・レジスタのベース、リミット、属性の各フィールドがすべてロードされる。ただし、データ・セグメントとスタック・セグメントのセレクタ / ディスクリプタ・レジスタの内容は無視される。
- 64 ビットモードでは、FS セグメント・レジスタと GS セグメント・レジスタは、他のセグメント・レジスタとは異なる方法で扱われる。FS セグメント・オーバライドや GS セグメント・オーバライドを使用する場合は、(FS または GS).base + index + displacement のように、リニアアドレス計算にそれぞれのベースアドレスが使用される。その後、FS.base と GS.base は、プロセッサがサポートしているフルサイズのリニアアドレスに合わせて拡張される。得られる実効アドレス計算は、正のアドレスと負のアドレスにラップアラウンドできる。また、結果のリニアアドレスは正規形式でなければならない。
- 64 ビットモードでは、FS セグメント・オーバライドや GS セグメント・オーバライドを使用するメモリアクセスについては、ランタイム・リミットのチェックや属性チェックは実行されない。FS と GS への通常のセグメント・ロード（MOV Sreg と POP Sreg）は、セグメント・ディスクリプタ・レジスタの隠れた部分に、標準の 32 ビットのベース値をロードする。64 ビットより小さいアドレス空間を使用す

るプロセッサ・モデルとの整合性を維持するために、標準の 32 ビットを超えるベース・アドレス・ビットは 0 にクリアされる。

- ディスクリプタ・レジスタ内の隠れた FS.base フィールドと GS.base フィールドは、64 ビット・プロセッサでサポートされるすべてのアドレスビットをロードできるように、MSR に物理的にマッピングされる。CPL = 0 のソフトウェア（特権ソフトウェア）は、WRMSR 命令を使用して、サポートされるすべてのリニア・アドレス・ビットを FS.base または GS.base にロードできる。
- FS.base の MSR インデックスは C0000100H であり、GS.base のインデックスは C0000101H である。64 ビットの FS.base レジスタと GS.base レジスタに書き込まれるアドレスは、正規形式でなければならない。WRMSR 命令がこれらのレジスタに非正規のアドレスを書き込もうとすると、一般保護例外（#GP）が発生する。
- 互換モードでは、FS オーバライドと GS オーバライドは、ディスクリプタ・レジスタの隠れたベース・フィールドのリニアアドレス上位 32 ビットにロードされる値に関係なく、従来の IA-32 アーキテクチャの定義に従って動作する。互換モードでは、実効アドレスを計算する際に、上位 32 ビットは無視される。
- 新しい 64 ビットモード命令の SWAPGS は、GS ベースのロードに使用できる。SWAPGS は、KernelGSbase MSR から得られるカーネルデータ構造ポインタと GS ベースレジスタを交換する。その後、カーネルは、通常のメモリ参照で、GS プリフィックスを使用してカーネルデータ構造にアクセスできる（WRMSR を使用して KernelGSBase MSR に非正規の値を書き込もうとすると、#GP フォルトが発生する）。

1.6.4.3. システム・ディスクリプタ

特定のモードでは、システム・ディスクリプタは、64 ビットのベースアドレスを扱えるように、64 ビットだけ拡張される。どのような条件でこのサイズの拡張が行われるかは、ディスクリプタの使用目的によって異なる。以下の点に注意する。

- GDTR、IDTR、LDTR、TR レジスタにロードされるディスクリプタと疑似ディスクリプタは、システムテーブルの定義に使用される。これらのディスクリプタは、64 ビットモードでは拡張されるが、互換モードでは拡張されない。
- IA-32e モードでは、システムテーブルを定義するディスクリプタのうち、アプリケーション・プラットフォームによって参照されるものが拡張される。これらのディスクリプタには、コールゲート、割り込みゲート、トラップゲートが含まれる（IA-32e モードでは、タスクゲートはサポートしていない）。
- GDTR、LDTR、IDTR、TR システム・ディスクリプタ・レジスタ（プロセッサは、これらのレジスタを使用して、GDT、LDT、IDT システム・ディスクリプタ・テーブルと現在のプロセスの TSS を見つける）は、IA-32e モードでは、拡張されたメモリアドレス空間をサポートするように変更される。

- LDT と TSS のベースアドレスは、関連するディスクリプタによって指定される。これらのディスクリプタ・レジスタは、64 ビットアドレスを格納できるように拡張されている（1.3.3.3. 節を参照）。一方、GDT と IDT は、ディスクリプタを使用しない。GDT と IDT のベースは、LGDT 命令と LIDT 命令によってロードされる。64 ビットモードでは、これらの命令のオペランド・サイズは、64 ビットのベースアドレスを指定できるように拡張される（ただし、互換モードとレガシーモードでは拡張されない）。
- IA-32e モードでは、プロセッサは、ディスクリプタ・テーブルのリミットをチェックする。4 個のディスクリプタ・テーブル・レジスタのリミット・フィールドのサイズは、どれも従来の IA-32 サイズから変更されていない。GDTR と IDTR のリミットは 16 ビットのままであり、LDTR と TR のリミットは通常の 20/32 ビットである。LDTR レジスタと TR レジスタのセグメント属性フィールドのサイズも、IA-32e モードでは変更されていない。
- 既存の LDT タイプ・フィールド（02H）と既存の 32 ビット TSS タイプ・フィールド（09H）は、64 ビットモードでは、64 ビット LDT タイプと 64 ビット TSS タイプとして使用できるように再定義される。LDT システム・ディスクリプタと TSS システム・ディスクリプタは、64 ビットのベースアドレスを格納できるように、64 ビットだけ拡張されて 16 バイトになる（表 1-17.）。

表 1-17. 64 ビットモードの LDT ディスクリプタと TSS ディスクリプタ

	ビット位置							
	31:24	23	22:20	19:16	15	14:13	12:8	7:0
バイト 15:12	予約済み						00000	予約済み
バイト 11:8	ベース 63:32							
バイト 7:4	ベース アドレス 31:24	G	予約済み	リミット 19:16	P	DPL	タイプ	ベース 23:16
バイト 3:0	ベースアドレス 15:0				リミット 15:0			

バイト 11:8 は、ベースアドレスの上位 32 ビットを正規形式で格納する。整合性チェックに使用される第 2 のタイプ・フィールドは、最上位ダブルワード（バイト 15:12）のビット 12:8 に定義されている。このフィールド全体が 0（無効なタイプ）にクリアされていなければならない。このフィールドを無効なタイプ（00H）にしておくことにより、従来の IA-32 ディスクリプタとして 64 ビットモード・ディスクリプタの上位半分にアクセスしようとした場合、一般保護例外（#GP）を生成できる。

64 ビットモードでは、既存のタイプ・フィールド（最下位ダブルワードのビット 12:8）は、既存のタイプコードの一部が再定義される。例えば、32 ビット LDT タイプ（02H）と 32 ビット TSS タイプ（09H）は、64 ビットモードでは、64 ビット LDT タイプと 64 ビット TSS タイプとして再定義される。互換モードでは、従来と同じように、02H タイプは 32 ビット LDT を参照し、09H タイプは 32 ビット TSS を参照する。タイプ・

フィールドのその他のコードの定義や再定義は行われていない。ディスクリプタ内で指定される64ビットのベースアドレスは、正規形式でなければならない。正規形式でない場合は、一般保護例外#GP（セレクタ）が発生する。

LGDT命令とLIDT命令は、GDTRレジスタまたはIDTRレジスタに疑似ディスクリプタをロードする。すべてのモード（レガシー、互換、64ビット）で、最初にロードされる2バイトは16ビットリミットである。次にロードされるバイトは、モードによって異なる。

- 16ビットモードまたは32ビットモード（レガシーモードまたは互換モード）では、次にロードされる4バイトはベースであり、合計6バイトがロードされる。
- 64ビットモードでは、次にロードされる8バイトはベースであり、合計10バイトがロードされる。

LGDT命令とLIDT命令では、オペランド・サイズ・プリフィックスは無視される。64ビットモードでは、GDTRレジスタとIDTRレジスタにロードされる64ビットのベースアドレスは、正規形式でなければならない。正規形式でない場合は、一般保護例外#GP(0)が発生する。

LLDT命令とLTR命令は、プロセッサの内部LDTRおよびTRセグメント・ディスクリプタ・レジスタ（隠れた部分）にシステム・ディスクリプタをロードする。64ビットモードでは、ディスクリプタ・フォーマットの拡張とディスクリプタ・タイプの再定義により、LLDT命令とLTR命令がロードできるディスクリプタに、以下の制限が適用される。

- 第2のタイプ・フィールド（最上位ダブルワードのビット12:8）に00H以外の値をロードしようとすると、一般保護例外#GP（セレクタ）が発生する。
- LLDTまたはLTRによってロードされる64ビットのベースアドレスは、正規形式でなければならない。正規形式でない場合は、一般保護例外 #GP（セレクタ）が発生する。
- LTR命令がビジー状態のTSSまたは16ビットTSSを参照すると、一般保護例外#GP（セレクタ）が発生する。

64ビットモードでも、LTR命令は、タスクのステートをビジーに変更する（ディスクリプタ・タイプを0BHに設定する）。IA-32eモードはタスクスイッチをサポートしていないため、タスク・ディスクリプタのビジービットは自動的にクリアされない。オペレーティング・システムがLTR命令を使用してタスク・ディスクリプタのロードを実行していた場合、オペレーティング・システムがそのタスクのビジービットをクリアする（ディスクリプタ・タイプを09Hに設定する）役割を受け持つ。LLDT命令とLTR命令によって発生するメモリアクセスの実際の順序は、プロセッサ・モデルによって異なる。

1.6.5. リニアアドレス指定とページング

IA-32eモードがイネーブルの場合、リニアアドレスから物理アドレスへの変換方法は、従来の保護モードとは異なる。新しいページ・マッピング・テーブルであるページ・マップ・レベル4（PML4）の導入により、64ビットアドレスは、以下の項目で説明する変換方法で物理アドレスに変換される。

1.6.5.1. 64ビットモードのソフトウェア・アドレス変換

64ビットモードでは、フラットなアドレス空間を使用するため、リニアアドレスは実効アドレスに等しくなる。ただし、メモリの節ですでに説明したように、0でないベースと組み合わせてFSセグメントまたはGSセグメントを使用する場合は、リニアアドレスと実効アドレスは一致しない。

64ビットモードでは、一般的に、変位と即値は64ビットに拡張されない。64ビットモードでも、変位と即値は32ビットまでに制限され、実効アドレス計算の際に符号拡張される。ただし、64ビットモードでは、MOV命令については64ビット形式の変位と即値をサポートしている。

IA-32eモードでは、すべての16ビット/32ビットアドレス計算は0で拡張され、64ビットアドレスに変換される。アドレス計算は、まず、任意のアドレス・サイズ・プリフィックスによってオーバーライドされた、現在のモード（64ビットモードまたは互換モード）の実効アドレスサイズに合わせて切り捨てられる。次に、その結果は、フル64ビットのアドレスサイズに合わせて0で拡張される。このため、互換モードで動作する16ビット・アプリケーションと32ビット・アプリケーションは、64ビットモードの実効アドレスの下位4GBにのみアクセスできる。同じように、64ビットモードで生成される32ビットアドレスも、64ビットモードの実効アドレス空間の下位4GBにのみアクセスできる。

1.6.5.2. ページング・データ構造

64ビット拡張技術アーキテクチャは、64ビットのリニアアドレスから52ビットの物理アドレスへのマッピングをサポートするように、物理アドレス拡張機構（PAE）ページング構造を拡張する。インテル® EM64Tを搭載した最初のプロセッサ・モデルでは、PAEページング構造は、48ビットのリニアアドレスから40ビットの物理アドレスへの変換をサポートするように拡張される。

IA-32eモードをアクティブにする前に、CR4.PAE = 1にセットして、PAEをイネーブルにしなければならない。PAEは、32ビットを超える物理アドレスサイズをサポートするために、個々のページ・ディレクトリ・エントリ（PDE）とページ・テーブル・エントリ（PTE）のサイズを32ビットから64ビットに拡張する。PAEをイネーブルにする前にIA-32eモードをアクティブにしようとすると、一般保護例外（#GP）が発生する。

64 ビット拡張技術アーキテクチャでは、ページ・マップ・レベル 4 テーブル（PML4）と呼ばれる新しいテーブルが、リニアアドレス変換階層に追加されている。PML4 は、ページ変換階層内でページ・ディレクトリ・ポインタ・テーブル（PDP）の上位に位置する。PML4 は、512 個の 8 バイトエントリで構成され、各エントリが PDP テーブルを指す。9 個のリニア・アドレス・ビットを使用して、PML4 内のインデックスを指定する。

PML4 テーブルは、IA-32e モードがアクティブの場合にのみ、ページ変換に使用される。IA-32e モードがディスエーブルの場合は、PAE がイネーブルかどうかに関係なく、PML4 テーブルは使用されない。既存のページ・ディレクトリ・ポインタ・テーブルは、64 ビット拡張技術により、4 個のエントリから、512 個の 8 バイトエントリに拡張されている。その結果、（2 ビットではなく）9 ビットのリニアアドレスを使用して、PDP テーブル内のインデックスを指定できる。ページ・ディレクトリ・エントリ（PDE）テーブルとページ・テーブル・エントリ（PTE）テーブルのサイズは、512 個の 8 バイトエントリのままであり、それぞれ 9 個のリニア・アドレス・ビットによってインデックスを指定される。上に定義されたページング・データ構造の集合体に対するリニア・アドレス・インデックス・ビットの合計（PML4 + PDP + PDE + PTE + ページ・オフセット）は、48 になる。リニア・アドレスの上位 16 ビットを物理アドレスに変換する手法は、現在のところ予約されている。

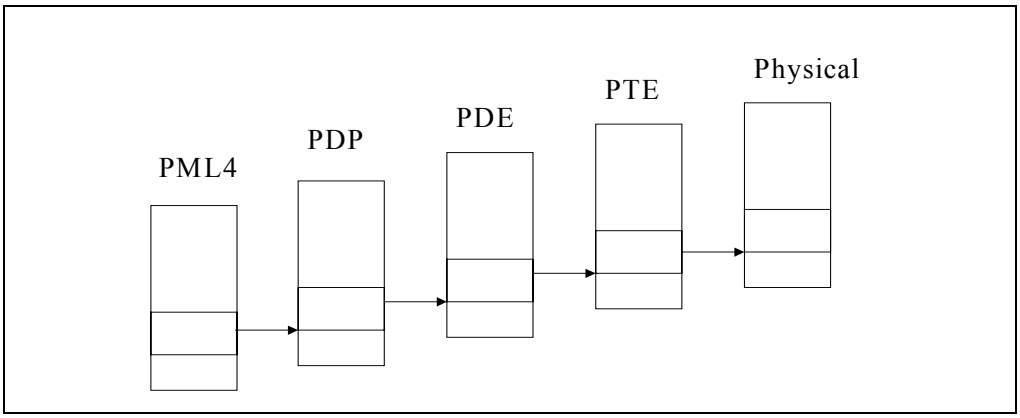

図 1-6. ページング・データ構造

ページ・ディレクトリ・エントリ内の PS フラグ（PDE.PS）は、4K バイトまたは 2M バイトのページサイズを選択する。ラージページの選択には PDE.PS を使用するため、CR4.PSE ビットは無視される。表 1-18. ～表 1-21. は、4K バイトページがイネーブルの場合の 64 ビットモードの PML4、PDP、PDE、PTE のフォーマットを示している。

表 1-18. IA-32e モードのページ・マップ・レベル 4 のエントリ（PML4、4K ページ）

63	62:52	51:40	39:12	11:9	8:6	5	4	3	2	1	0
予約済み	利用可能	予約済み	ページ・ディレクトリ・ポインタのベースアドレス	利用可能	予約済み	A	PCD	PWT	U/S	R/W	P

表 1-19. IA-32e モードのページ・ディレクトリ・ポインタ・テーブルのエントリ（PDPTE、4K ページ）

63	62:52	51:40	39:12	11:9	8:6	5	4	3	2	1	0
予約済み	利用可能	予約済み	ページ・ディレクトリのベースアドレス	利用可能	予約済み	A	PCD	PWT	U/S	R/W	P

表 1-20. IA-32e モードのページ・ディレクトリ・エントリ（PDE、4K ページ）

63	62:52	51:40	39:12	11:9	8	7	6	5	4	3	2	1	0
予約済み	利用可能	予約済み	ページテーブルのベースアドレス	利用可能	予約済み	0	予約済み	A	PCD	PWT	U/S	R/W	P

表 1-21. IA-32e モードのページ・テーブル・エントリ（PTE、4K ページ）

63	62:52	51:40	39:12	11:9	8	7	6	5	4	3	2	1	0
予約済み	利用可能	予約済み	ページのベースアドレス	利用可能	G	PAT	D	A	PCD	PWT	U/S	R/W	P

4 種類すべてのテーブル・エントリ・フォーマットの物理ベース・アドレス・フィールドは、64 ビット拡張技術によってビット 51:12 に拡張される。これにより、64 ビット・プロセッサによってサポートされる物理メモリ内の任意の位置に、ページング・テーブルを配置できる。最大物理アドレスサイズをサポートしていないプロセッサでは、サポートしていない上位ビットは予約済みとされ、0 にクリアされる必要がある。IA-32e テクノロジを搭載した最初のプロセッサ・モデルの物理ベース・アドレス・フィールドは、ビット 39:12 によって指定される。レガシーモードへのモードの変更が必要になることが予想される場合、ソフトウェアは、メモリ内の 4G を超える領域にページング・テーブルを配置しないように注意する必要がある。

ビット63は予約済みである。すべてのページ・テーブル・エントリ・フォーマットのビット62:52は、システム・ソフトウェアによって利用可能である。64ビット拡張技術アーキテクチャでは、今後のプロセッサ・モデルでも、ビット62:52はソフトウェアによって利用可能とされる。ベース・アドレス・フィールドが拡張され、ソフトウェアによって利用可能なフィールドがビット62:52に追加された以外は、PDEとPTEのその他のフィールドはすべてレガシーモードと同じである。

PDPテーブルエントリ内のフィールドは、レガシーモードのPDPテーブルエントリとほぼ同じであるが、以下の点が異なる。これらの相違点は、より上位のページング構造（PML4）がPDPテーブルを参照するようになったために必要とされる変更を示している。

- ビット0は予約済みではなくなった。IA-32eモードでは、このビットは、PDPエントリによって参照されるPDEテーブルが物理メモリ内に現在格納されているかを示す存在（P）フラグとして定義される。プロセッサが、Pフラグが0にクリアされたPDPエントリにアクセスすると、ページフォルト例外（#PF）が発生する。
- ビット1は予約済みではなくなった。IA-32eモードでは、このビットは読み取り/書き込み（R/W）フラグとして定義される。
- ビット2は予約済みではなくなった。IA-32eモードでは、このビットはユーザ/スーパーバイザ（U/S）フラグとして定義される。
- ビット5は予約済みではなくなった。IA-32eモードでは、このビットはアクセス（A）フラグとして定義される。
- 前述したベース・アドレス・フィールドの拡張。
- 前述したように、ビット62:52はソフトウェアによって利用可能である。PML4テーブルエントリのフォーマットは、64ビットモードのPDPテーブルエントリのフォーマットと同じである。

表1-22.～表1-24.は、2Mバイトページがイネーブルの場合の64ビットモードのPML4、PDP、PDEのフォーマットを示している。レガシーモードと同じように、2Mバイトページをイネーブルにするには、PDEのページ・サイズ・ビットを1にセットする（PDE.PS = 1）。2Mページサイズの選択は、CR4.PSEには依存しない。

表1-22. IA-32eモードのページ・マップ・レベル4のエントリ（PML4、2MBページ）

63	62:52	51:40	39:12	11:9	8:6	5	4	3	2	1	0
予約済み	利用可能	予約済み	ページ・ディレクトリ・ポインタのベースアドレス	利用可能	予約済み	A	PCD	PWT	U/S	R/W	P

表 1-23. IA-32e モードのページ・ディレクトリ・ポインタ・テーブル・エントリ（PDPTR、2MB ページ）

63	63:52	51:40	39:12	11:9	8:6	5	4	3	2	1	0
予約済み	利用可能	予約済み	ページ・ディレクトリのベースアドレス	利用可能	予約済み	A	PCD	PWT	U/S	R/W	P

表 1-24. IA-32e モードのページ・ディレクトリ・エントリ（PDE、2MB ページ）

63	62:52	51:40	39:21	20:13	12	11:9	8	7	6	5	4	3	2	1	0
予約済み	利用可能	予約済み	ページのベースアドレス	予約済み	PAT	利用可能	G	1	D	A	PCD	PWT	U/S	R/W	P

3 種類すべてのテーブル・エントリ・フォーマットの物理ベース・アドレス・フィールドは、64 ビット拡張技術によってビット 51:12 に拡張される。これにより、64 ビットモードのプロセッサによってサポートされる物理メモリ内の任意の位置に、ページング・テーブルを配置できる。最大物理アドレスサイズをサポートしていないプロセッサでは、サポートしていない上位ビットは予約済みとされ、0 にクリアされる必要がある。

インテル EM64T を搭載した最初のプロセッサ・モデルの物理ベース・アドレス・フィールドは、ビット 39:12 によって指定される。すべてのページ・テーブル・エントリ・フォーマットのビット 63:52 は、システム・ソフトウェアによって利用可能である。64 ビット拡張技術では、今後のプロセッサ・モデルでも、ビット 63:52 はソフトウェアによって利用可能とされる。2M バイトページが選択されている場合、PDE は PTE を指すのではなく、直接に物理ページを指す。ベース・アドレス・フィールドが拡張され、ソフトウェアによって利用可能なフィールドがビット 63:52 に追加された以外は、PDE のその他のフィールドはすべてレガシーモードと同じである。

PDP テーブルエントリ内のフィールドは、レガシーモードの PDP テーブルエントリとほぼ同じであるが、以下の点が異なる。これらの相違点は、より上位のページング構造（PML4）が PDP テーブルを参照するようになったために必要とされる変更を示している。

- ビット 0 は予約済みではなくなった。IA-32e モードでは、このビットは、PDP エントリによって参照される PDE テーブルが物理メモリ内に現在格納されているかを示す存在（P）フラグとして定義される。プロセッサが、P フラグが 0 にクリアされた PDP エントリにアクセスすると、ページフォルト例外（#PF）が発生する。
- ビット 1 は予約済みではなくなった。IA-32e モードでは、このビットは読み取り / 書き込み（R/W）フラグとして定義される。

- ビット2は予約済みではなくなった。IA-32eモードでは、このビットはユーザ/スーパーバイザ（U/S）フラグとして定義される。
- ビット 5 は予約済みではなくなった。IA-32e モードでは、このビットはアクセス（A）フラグとして定義される。
- 前述したベース・アドレス・フィールドの拡張。
- 前述したように、ビット62:52はソフトウェアによって利用可能である。

PML4テーブルエントリのフォーマットは、64ビットモードのPDPテーブルエントリのフォーマットと同じである。

1.6.5.3. 全体的なページ保護

ページングの階層が順次追加されたため、R/WフラグとU/Sフラグに関する同一のIA-32手法を4レベルのページテーブルに適用して、ページ保護方法を変更する必要が生じる。表1-25.は、IA-32eモードのページング保護を示している。

表 1-25. IA-32e モードのページレベル保護のマトリックス

特権（U/S ビット、ビット 2）				アクセスタイプ（R/W ビット、ビット 1）				組み合わせた影響	
PML4	PDP	PDE	PTE	PML4	PDP	PDE	PTE		
ユーザ	ユーザ	ユーザ	ユーザ	RO	*	*	*	ユーザ	RO
ユーザ	ユーザ	ユーザ	ユーザ	*	RO	*	*	ユーザ	RO
ユーザ	ユーザ	ユーザ	ユーザ	*	*	RO	*	ユーザ	RO
ユーザ	ユーザ	ユーザ	ユーザ	*	*	*	RO	ユーザ	RO
ユーザ	ユーザ	ユーザ	ユーザ	R/W	R/W	R/W	R/W	ユーザ	R/W
スーパー	*	*	*	*	*	*	*	スーパー	R/W
*	スーパー	*	*	*	*	*	*	スーパー	R/W
*	*	スーパー	*	*	*	*	*	スーパー	R/W
*	*	*	スーパー	*	*	*	*	スーパー	R/W

ページウォーク内の複数レベルのページエントリから得られる、特権レベルとアクセスタイプの組み合わせには、次の2つの基本規則がある。

1. ページウォーク内のページエントリのうち最高の特権モードが使用される（例えば、いずれかのページエントリ内で U/S ビットがクリアされている場合、そのページはスーパーバイザ特権レベルになる）。
2. ページウォーク内のページエントリのうち最も制限されたアクセスタイプが使用される（例えば、いずれかのページエントリ内でR/W ビットがクリアされている場合、そのページは読み取り専用アクセスタイプになる）。

1.6.5.4. 予約ビットのチェック

プロセッサは、以下のページングモード固有のビットについて、予約ビットのチェックを実行する。表 1-26. は、チェックされる予約ビットを示している。表 1-26. のレガシー・ページング・モードは、以下のとおりである。

- 非 PAE 4KB ページング : 4KB ページのみのページング（CR4.PAE = 0、CR4.PSE = 0）
- PSE-36: 4KB ページと 4MB ページ（CR4.PAE = 0、CR4.PSE = 1）
- PAE: 4KB ページと 2MB ページ（CR4.PAE = 1、CR4.PSE = x）

表 1-26. 予約ビットのチェック

モード	ページング・モード	チェックされるビット
レガシー	4KB ページング（非 PAE）	チェックされる予約ビットはない
	PSE36 - PDE、4MB ページ	ビット [21]
	PSE36 - PDE、4KB ページ	チェックされる予約ビットはない
	PSE36 - PTE	チェックされる予約ビットはない
	PAE - PDP テーブルエントリ	ビット [63:40] と [8:5] と [2:1]
	PAE - PDE、2MB ページ	ビット [63:40] と [20:13]
	PAE - PDE、4KB ページ	ビット [63:40]
	PAE - PTE	ビット [63:40]
64 ビット	PML4E	ビット [63]、ビット [51:40]
	PDPTE	ビット [63]、ビット [51:40]
	PDE、2MB ページ	ビット [63]、ビット [51:40] と [20:13]
	PDE、4KB ページ	ビット [63]、ビット [51:40]
	PTE	ビット [63]、ビット [51:40]

1.6.6. 拡張されたレガシー・モード・ページング

IA-32eモードで使用される大きな物理アドレスサイズをサポートするためのページング・データ構造のいくつかの変更は、レガシーモードのオペレーティング・システムでも利用可能である。レガシーモードのオペレーティング・システムは、物理アドレス拡張機構（PAE）のサポートとページサイズ拡張機構（PSE）のサポートの拡張（物理アドレスビットの追加）を利用できる。ただし、IA-32e モードで導入された 4 レベルのページ変換機構は、レガシーモードのソフトウェアは利用できない。

すでに説明したように、CR4.PAE = 1 にセットすると、個々の PDE と PTE のサイズは 32 ビットから 64 ビットに拡張される。これにより、32 ビットを超える物理アドレスサイズが利用可能になる。従来の IA-32 プロセッサでは、物理アドレスサイズは 36 ビットに制限されている。

インテル EM64T を搭載したプロセッサのアーキテクチャは、レガシーモードのソフトウェアが PDE と PTE に最大 52 ビットの物理アドレスをロードすることを認めている。サポートしていない物理アドレスビットは予約済みとされ、0 にクリアされていなければならない。インテル EM64T を搭載した最初のプロセッサ・モデルでは、レガシーモードのソフトウェアは PDE と PTE のエントリ内で最大 40 ビットの物理アドレスを使用できる。ソフトウェアは、ビット 62:40 を 0 にクリアしなければならない。ビット 63 は予約済みである。

レガシーモードのページサイズ拡張機構（PSE）をイネーブルにするには、CR4 ページ・サイズ・イネーブル・ビットを 1 にセットする（CR4.PSE = 1）。PSE は、従来の 4K バイトページ以外に 4M バイトページもサポートするように、元の 4 バイト PDE フォーマットを変更する。4M バイトページを選択するには、PDE ページ・サイズ・ビットを 1 にセットする（PDE.PS = 1）。このビットをクリアすると、4K バイト・ページが選択される（PDE.PS = 0）。

PDE.PS = 1 の場合、プロセッサは、PDE ビット 31:22 とリニア・アドレス・ビット 21:0 を組み合わせて、4M バイトページ内に 32 ビット物理アドレスを構成する。4M バイト・ページ変換では、従来の PTE は使用されない。従来の PTE が使用されないため、元の PSE モードの定義では、PDE ビット 21:12 は予約済みになる。PSE モードを更新すると、4 バイト PDE フォーマットが変更され、PAE によって使用される 8 バイト・フォーマットを必要とせずに、36 ビット物理アドレスがサポートされる。これを実現するために、以前予約されていた PDE ビット 16:13 を使用して、追加の上位物理アドレスビット 4 ビットを格納する。ビット 21:17 は予約済みである。

インテル EM64T をサポートする IA-32 プロセッサのアーキテクチャでは、PSE モードの 4 バイト PDE フォーマットがさらに変更され、物理アドレスサイズのサポートが 40 ビットに拡張される。これを実現するために、以前予約されていた PDE ビット 20:17 を定義して、追加の上位物理アドレスビット 4 ビットを格納する。ビット 21 は予約済みとされ、0 にクリアされていなければならない。表 1-27. は、PSE モードがイネーブルの場合の PDE のフォーマットを示している。上位物理アドレスビット 39:32 は、PDE[20:13] に置かれる。物理アドレスビット 31:22 は、PDE[31:22] に置かれる。

表 1-27. レガシーモードのページ・ディレクトリ・エントリ（4MB ページ）

31:22	21	20:13	12	11:9	8	7	6	5	4	3	2	1	0
ベースアドレス 31:22	0	ページのベースアドレス 39:32	PAT	利用可能	G	1	D	A	PCD	PWT	U/S	R/W	P

1.6.7. CR2 と CR3

CR2（ページフォルト・アドレス・レジスタ）のサイズは、64 ビットのリニアアドレスを格納できるように、64 ビット拡張技術によって 64 ビットに拡張される。また、CR3（ページ・ディレクトリ・ベース・レジスタ）のサイズも、64 ビットに拡張される。これにより、プロセッサに依存する物理アドレスサイズのリミットに従って、第 1 レベルのページング構造を物理メモリ内の任意の位置に配置できる。

表 1-28. は、CR3 の 64 ビット・モード・フォーマットを示している。ベース・アドレス・フィールドは、ページ・ディレクトリ・ベース・アドレスのビット 11 より上位のビットを指定する。ページ・ディレクトリ・ベース・フィールドは、最上位のページング構造の最上位の物理アドレスビットを格納する。CR3 のビット 51:12 は、64 ビットモードのアーキテクチャで許容される最大ベースアドレスを定義するが、プロセッサ・モデルによっては、サポートする物理アドレス空間はこれより小さくなる。ベースアドレスの下位 12 ビットは、常に 0 と見なされる。これにより、最上位のページング構造のアライメントは 4K バイトに合わされる。

表 1-28. IA-32e モードの CR3

63:52	51:40	39:32	32:12	11:5	4	3	2:0
予約済み	予約済み	ベース・アドレス	ページ・ディレクトリ・ベース	予約済み	PCD	PWT	予約済み

CR3[12] から、プロセッサがサポートしている物理アドレス空間の上限を表すビット位置（プロセッサが 40 ビットの物理アドレス空間をサポートしている場合は、CR3[39]）までのビットは、最上位のページング構造（PML4）のベースアドレスを指定する。CR3[51] から、プロセッサがサポートしている物理アドレス空間の上限を表すビット位置までのビット（例えば、プロセッサが 40 ビットの物理アドレスをサポートしている場合は、CR3[51:40]）は予約済みであり、0 にクリアされていなければならない。

CR3[63:52] は予約済みである。CR3[51:40] は、ページ・ディレクトリ・ベース・アドレスの将来の拡張に備えて予約されている。インテル EM64T を搭載した最初のプロセッサ・モデルでは、プロセッサはこれらのビットに 0 が書き込まれているかをチェックし、0 以外の値が書き込まれている場合は、一般保護例外 #GP(0) を生成する。

IA-32e モードでは、MOV to CR3 命令はオペランド・サイズの影響を受けない。64 ビットモードでは、CR3 の全 64 ビットはソースレジスタからロードされる。互換モードでは、CR3 の下位 32 ビットだけがソースレジスタからロードされ、上位 32 ビットは 0 にクリアされる。

1.6.8. アドレス変換

IA-32eモードでページングを使用する場合、プロセッサは、レガシーモードと同じように、リニアアドレスを分割し、テーブル・オフセットと物理ページ・オフセットの集合体に変換する。ただし、64ビット・アーキテクチャは、64ビットのリニア・アドレス・サイズとページング・データ構造の階層の追加をサポートするために、プロセッサがリニアアドレスを分割する方法を拡張している。

4Kバイトページをイネーブルにするには、PDEページ・サイズ・フラグをクリアする（PDE.PS = 0）。IA-32eテクノロジを搭載した最初のプロセッサは最大48ビットのリニアアドレスをサポートしているため、このページング・オプションは、2^{48}バイト（256テラバイト）のリニアアドレス空間にわたる、2^{36}の4Kバイトページをサポートする。

48ビットのリニアアドレスは、次のように、4レベルのページング構造内のインデックスを指定する5つのフィールドに分割される。

- ビット47:39は、512エントリのページ・マップ・レベル4テーブル（PML4）内のインデックスを指定する。
- ビット38:30は、512エントリのページ・ディレクトリ・ポインタ・テーブル（PDP）内のインデックスを指定する。
- ビット29:21は、512エントリのページ・ディレクトリ・テーブル（PDE）内のインデックスを指定する。
- ビット20:12は、512エントリのページ・テーブル（PDE）内のインデックスを指定する。
- ビット11:0は、物理ページ内のバイト・オフセットを指定する。

表 1-29. 4KB ページの変換

63:48	47:39	38:30	29:21	20:12	11:0
符号拡張	ページ・マップ・レベル 4 テーブル・オフセット（PML4E）	ページ・ディレクトリ・ポインタ・オフセット（PDPE）	ページ・ディレクトリ・テーブル・オフセット（PDE）	ページ・テーブル・オフセット（PTE）	ページ・オフセット

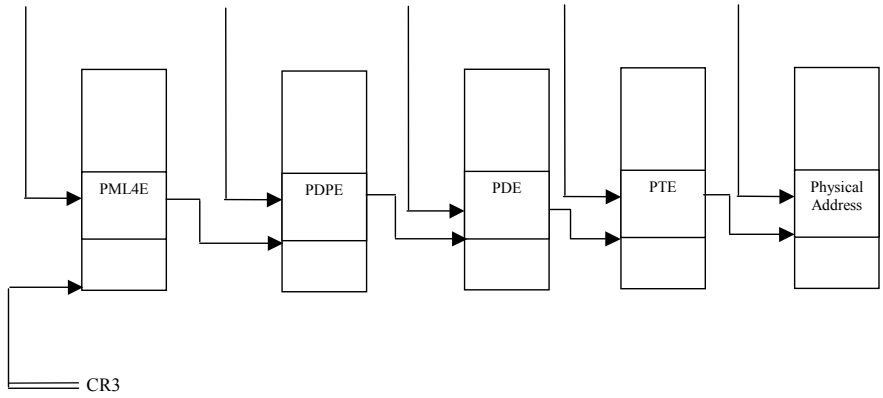

表 1-30. CR3 ページ・ディレクトリ・ポインタ・オフセット

63:40	39:12	11:0
	ページ・ディレクトリ・ポインタ・オフセット	

2M バイトページをイネーブルにするには、PDE ページ・サイズ・フラグを 1 にセットする（PDE.PS = 1）。インテル EM64T を搭載した最初のプロセッサは最大 48 ビットのリニアアドレスをサポートしているため、このページング・オプションは、2^{48} バイト（256 テラバイト）のリニアアドレス空間にわたる、2^{27} の 2M バイトページをサポートする。

48 ビットのリニアアドレスは、次のように、3 レベルのページング構造内のインデックスを指定する 4 つのフィールドに分割される。

- ビット 47:39 は、512 エントリのページ・マップ・レベル 4 テーブル内のインデックスを指定する。
- ビット 38:30 は、512 エントリのページ・ディレクトリ・ポインタ・テーブル内のインデックスを指定する。
- ビット 29:21 は、512 エントリのページ・ディレクトリ・テーブル内のインデックスを指定する。
- ビット 20:0 は、物理ページ内のバイト・オフセットを指定する。

表 1-31. 2MB ページの変換

63:48	47:39	38:30	29:21	20: 0
符号拡張	ページ・マップ・レベル4テーブル・オフセット（PML4E）	ページ・ディレクトリ・ポインタ・オフセット（PDPE）	ページ・ディレクトリ・オフセット（PDE）	ページ・オフセット

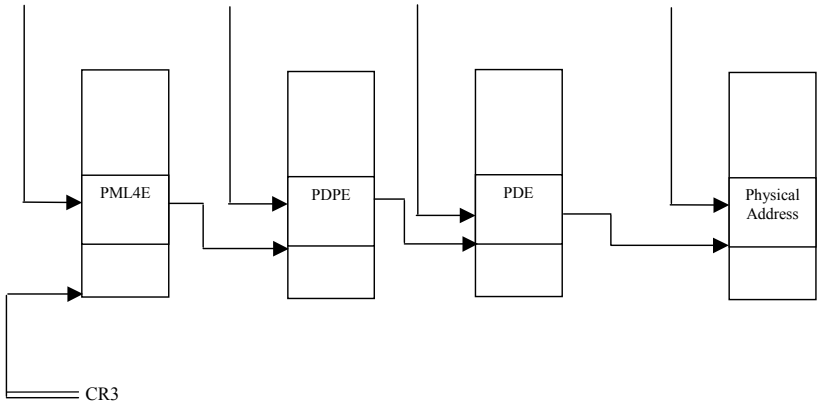

表 1-32. CR3

63:40	39:12	11:0
	ページ・ディレクトリ・ポインタ・オフセット	

1.6.9. 特権レベルの移行と far 転送

64 ビット拡張技術は、特権レベルを変更するための 3 つのメカニズムを用意している。

- コールゲートと割り込みゲート
- SYSCALL 命令と SYSRET 命令
- SYSENTER 命令と SYSEXIT 命令

1.6.9.1. コールゲート

コールゲート機構は、オペレーティング・システムへのパブリックなエントリ・ポイントを提供する。また、コールゲート機構は、オペレーティング・システムを呼び出す際に特権レベルとスタックを変更する手段も提供する。

従来のIA-32コールゲート・ディスクリプタは、命令ポインタ（EIP）のための32ビット・オフセットを提供する。64ビット拡張技術は、従来のコールゲートのサイズを2倍に拡張して、命令ポインタ（RIP）のための64ビット・オフセットを提供する。表1-33. は、IA-32eモードのコールゲート・ディスクリプタのレイアウトを示している。表1-34.は、64ビットモードのコールゲート内のフィールドの説明を示す。

表 1-33. IA-32e モードのコールゲート

	ビット位置				
DW オフセット	31:16	15	14:13	12:8	7:0
3	予約済み			00000	予約済み
2	オフセット 63:32				
1	オフセット 31:16	P	DPL	0CH（タイプ）	00000000
0	ターゲット・セグメント・セレクタ	オフセット 15:0			

64ビットモードのコールゲートの最初の8バイト（バイト7:0）は、従来の32ビット・コールゲートによく似ているが、同一ではない。パラメータ・コピー・カウント・フィールドが削除されている。バイト11:8は、ターゲット・セグメント・オフセットの上位32 ビットを正規形式で格納する。ソフトウェアが、正規形式でないターゲット・オフセットを指定してコールゲートを使用しようとすると、一般保護例外（#GP）が発生する。

コールゲートが参照するターゲット・コード・セグメントは、64ビット・コード・セグメント（CS.L = 1、CS.D = 0）でなければならない。64 ビット・コード・セグメントでない場合は、一般保護例外 #GP（セレクタ）が発生し、ターゲット CS セレクタがエラーコードとして報告される。この2倍のサイズのディスクリプタは、従来の16ビット/ 32ビット・ディスクリプタと同じディスクリプタ・テーブル内に置くことができる。整合性チェックに使用される第2のタイプ・フィールドは、最上位ダブルワードのビット12:8 で定義される。このフィールドは、0にクリアされていなければならない。このフィールドが無効なタイプ（00H）になっているため、64ビットモードのディスクリプタの上位半分に従来のディスクリプタとしてアクセスしようとすると、一般保護例外（#GP）が発生する。

IA-32eモード（64ビットモードと互換モード）で参照できるコールゲートは、64ビットモードのコールゲートに限られる。従来の32ビット・コールゲート・タイプ（0CH）は、IA-32eモードでは、64ビット・コールゲート・タイプとして再定義される。
IA-32e モードには、32 ビット・コールゲート・タイプは存在しない。far コールが 16 ビット・コールゲート・タイプ（04H）を参照すると、一般保護例外（#GP）が発生する。

表 1-34. IA-32e モードのコールゲート・フィールド

ダブルワード・オフセット	ゲート・フィールド	機能
3	+12(31:13)	未使用
	+12(12:8)	0 でなければならない
	+12(7:0)	未使用
2	+8(31:0)	正規形式のオフセット
1	+4(31:16)	オフセット 31:16
	+4(15:13)	存在（P）とディスクリプタ特権レベル（DPL）
	+4(12:8)	64 ビットモードのコールゲート・タイプ（0Ch）
	+4(7:0)	未使用
0	+0(31:16)	ターゲット・セグメント・セレクタ
	+0(15:0)	オフセット 15:0

CALL が 64 ビットモードのコールゲートを参照したときに実行される動作は、32 ビットゲートを介した従来のコールで実行される動作と同じであるが、以下の点が異なる。

- スタックのプッシュは 8 バイト単位で実行される。
- 64 ビット RIP がスタック上にプッシュされる。
- パラメータのコピー操作は実行されない。

ソフトウェアは、正常に動作するためには、適合する far リターン命令サイズを使用する必要がある（例えば、スタックを適切に処理するためには、64 ビットコールからのリターンは、64 ビット・オペランド・サイズのリターン命令で実行しなければならない）。

1.6.9.2. 特権レベルの変更とスタックの切り替え

コールゲートを使用して、より高い特権レベルのコード・セグメントへの移行が可能である。IA-32e モードでは、コールゲートの保護チェックの規則はレガシーモードから変わっていない。ただし、IA-32e モードでは、それに関連するスタックスイッチが多少異なる。

IA-32e モードでは、コールゲートのターゲットは、64 ビット・コード・セグメントでなければならない。64 ビットモードは、セグメント化を使用しない。スタックポインタは、64 ビット・スタック・ポインタ（RSP）のみで構成され、SS セグメント・レジスタは無視される。

コールゲートによる 64 ビットモードの特権レベル移行の一環としてスタックが切り替えられるときは、新しいSSディスクリプタはロードされない。IA-32eモードでは、内側レベルの RSP だけが TSS からロードされる。新しい SS は強制的に NULL に設定され、SS セレクタの RPL フィールドは強制的に新しい CPL に設定される。新しい SS が NULL に設定されるのは、ネストされた far 転送（CALLF、INTn、割り込み、例外）を処理するためである。元の SS と RSP は、新しいスタック上にセーブされる表 1-35.）。次の RETF で、元の SS はスタックからポップされ、SS レジスタにロードされる。

表 1-35. 64 ビットモードのスタック・レイアウト（CPL の変更を伴う CALLF 命令の実行後）

レガシーモード			IA-32e モード		
元の SS セレクタ	+12			+24	元の SS セレクタ
元の ESP	+8			+16	元の RSP
CS セレクタ	+4			+8	元の CS セレクタ
EIP	0	**ESP**	**RSP**	0	RIP
< 4 バイト >					< 8 バイト >

要約すると、IA-32eモードのスタックスイッチは、従来のスタックスイッチと同じように機能するが、新しい SS セレクタが TSS からロードされず、強制的に NULL に設定される点が異なる。特権レベルの変更を伴う far コールまたは far リターンの結果行われる、すべての 64 ビットモードのスタック操作は、8 バイト単位で行われ、RSP を 8 バイト単位で変更する。

IA-32eモードは、レガシーモードの自動パラメータコピー機能をサポートしていない。IA-32eモードでは、コールゲート・カウント・フィールドは無視される。ソフトウェアは、新しいプロセススタック上にセーブされている元のスタック・セグメント・レジスタとスタックポインタを参照することによって、必要に応じて元のスタックにアクセスできる。

IA-32eモードでは、RETF は、特定の条件下で NULL の SS をロードすることを許される。ターゲット・モードが 64 ビットモードで、ターゲット CPL < > 3 の場合、IRET は、SS に NULL セレクタがロードされることを許可する。

割り込みや例外は、スタックスイッチ機構の一環として、TSS から新しい SS セレクタをフェッチして GDT または LDT からそれに対応するディスクリプタをロードする代わりに、新しい SS を NULL に設定する。新しい SS セレクタを NULL に設定するのは、これ以降のネストされた far 転送からのリターンを適切に処理するためである。呼び出し先プロシージャに割り込みがかけられた場合は、NULL の SS はスタックフレーム上にプッシュされる。次の RETF の実行時に、スタック上の NULL の SS は、新しい SS ディスクリプタをロードしないようにプロセッサに指示するフラグとして機能する。

1.6.9.3. 高速システムコール

SYSCALL 命令と SYSRET 命令は、セグメント化を使用しないフラットなメモリモデルを使用するオペレーティング・システム向けに設計されている。これらの命令は（SYSENTER/SYSEXIT とともに）、IA-32e モードの操作に最適である。SYSCALL と SYSRET は、互換モードではサポートしていない。SYSCALL と SYSRET の利用可能性の検出についての詳細は、CPUID 命令の説明を参照のこと。

SYSCALL と SYSRET の語彙は、64 ビット・コード・オフセットを指定する。IA-32e モードでは、EFLAGS 内のビットのクリア操作は、固定ではなく設定可能である。SYSCALL と SYSRET は、EFLAGS レジスタのセーブとリストアを行う。

MSR の上位 32 ビットにはすでにターゲット CS セレクタと SS セレクタが格納されているため、従来のシステム・ターゲット・アドレス・レジスタ（STAR）を拡張して 64 ビットのターゲット RIP アドレスを指定することはできない。IA-32e モードは、64 ビットのターゲット RIP を格納する、ロング STAR（LSTAR）と互換 STAR（CSTAR）という 2 つの新しい STAR レジスタを用意している。LSTAR は、IA-32e モードがアクティブで呼び出し元プログラムが 64 ビットモードである場合に SYSCALL が使用する、ターゲット RIP を格納する。CSTAR は、IA-32e モードがアクティブで呼び出し元プログラムが互換モードである場合に SYSCALL が使用する、ターゲット RIP を格納する。IA-32e を搭載した最初のプロセッサ・モデルでは、互換モードでは SYSCALL と SYSRET をサポートしていないため、CSTAR は利用できない。

IA-32e モードとレガシーモードで使用される、SYSCALL と SYSRET の CS セレクタと SS セレクタは、STAR 内に格納される。LSTAR と CSTAR は、WRMSR 命令によって書き込まれる。LSTAR と CSTAR に書き込まれるアドレスは、WRMSR 命令によって最初にチェックされ、正規形式であるかを確認される。正規形式でない場合は、一般保護例外（#GP）が発生する。表 1-36. は、STAR、LSTAR、CSTAR、FMASK の各レジスタのレイアウトと MSR 番号を示している。

表 1-36. STAR、LSTAR、CSTAR モデル固有レジスタ（MSRs）

		63:48	47:32	31:0
STAR	C0000081H	SYSRET の CS と SS	予約済み	予約済み
LSTAR	C0000082H	64 ビットモードの呼び出し元プログラムのターゲット RIP		
CSTAR	C0000083H	互換モードの呼び出し元プログラムのターゲット RIP		
FMASK	C0000084H	予約済み		SYSCALL の EFLAGS マスク

SYSCALL と SYSRET の動作の詳細については、第 2 巻の第 4 章を参照のこと。

1.6.9.4. タスク・ステート・セグメント

IA-32e モードは、従来の IA-32 タスク・スイッチング・アーキテクチャをサポートしていない。IA-32e モードでは、ソフトウェアがタスクの管理と切り替えを実行する必要がある。IA-32e モードで以下の操作を行うと、プロセッサは一般保護例外（#GP）を生成する。

- JMP、CALL、INTn、または割り込みによる、TSS またはタスクゲートへの制御の移行
- EFLAGS.NT（ネストされたタスク）を 1 にセットした IRET

IA-32e モードはハードウェア・タスク・スイッチング機構をサポートしていないが、64 ビット・タスク・ステート・セグメント（TSS）は存在する必要がある。表 1-37. は、64 ビット TSS のフォーマットを示している。この 64 ビット・フィールドは、IA-32e モードの重要な情報を格納するが、タスクスイッチ機構に直接関連付けられていない。格納される情報は、以下の 3 つである。

- RSPn: 特権レベル 0 ～ 2 のスタックポインタ（RSP）のフル 64 ビット正規形式
- ISTn: 割り込みスタックテーブル（IST、1.6.10.4. 節を参照）ポインタのフル 64 ビット正規形式
- I/O マップ・ベース・アドレス : 64 ビット TSS のベースから I/O パーミッション・ビットマップへの 16 ビット・オフセット

オペレーティング・システムは、IA-32e モードをアクティブにした後、少なくとも 1 つの 64 ビット TSS を作成しなければならない。オペレーティング・システムは、（64 ビットモードで）LTR 命令を実行し、64 ビットモードのプログラムと互換モードのプログラムの両方に使用される 64 ビット TSS へのポインタを、TR レジスタにロードしなければならない。

表 1-37. IA-32e モードの TSS フォーマット

バイト・オフセット	31:16	15:0
+64H	I/O マップのベース	予約済み
+60H	予約済み	
+5CH	予約済み	
+58H	IST7（上位 32 ビット）	
+54H	IST7（下位 32 ビット）	
+50H	IST6（上位 32 ビット）	
+4CH	IST6（下位 32 ビット）	
+48H	IST5（上位 32 ビット）	
+44H	IST5（下位 32 ビット）	
+40H	IST4（上位 32 ビット）	
+3CH	IST4（下位 32 ビット）	
+38H	IST3（上位 32 ビット）	
+34H	IST3（下位 32 ビット）	
+30H	IST2（上位 32 ビット）	
+2CH	IST2（下位 32 ビット）	
+28H	IST1（上位 32 ビット）	
+24H	IST1（下位 32 ビット）	
+20H	予約済み	
+1CH	予約済み	
+18H	RSP2（上位 32 ビット）	
+14H	RSP2（下位 32 ビット）	
+10H	RSP1（上位 32 ビット）	
+0CH	RSP1（下位 32 ビット）	
+08H	RSP0（上位 32 ビット）	
+04H	RSP0（下位 32 ビット）	
00H	予約済み	

1.6.10. 割り込み

割り込みと例外は、現在実行中のプログラムから、特定の割り込みを処理するサービスルーチンに強制的に制御を渡す。割り込み処理機構と例外処理機構は、割り込みをかけられたプログラムの実行ステートをセーブして、割り込みサービスルーチンに制御を渡し、最終的には、割り込みをかけられたプログラムにリターンする。

本項全体を通して、「割り込み」の用語は、プロセッサの外部で生成される非同期イベントと、命令の実行に関連する同期イベント（例外、フォルト、トラップ）の両方を対象とする。64 ビット拡張技術は、64 ビット・オペレーティング・システムと 64 ビット・アプリケーションをサポートするように、従来の IA-32 割り込み処理機構と例外処理機構を拡張する。この変更には、以下の点が含まれる。

- IDT によって指定されるすべての例外ハンドラは、64 ビットコードである（SMI ハンドラには適用されない）。
- 割り込みスタック・プッシュのサイズは、64 ビットに固定される。プロセッサは、0 で拡張された 8 バイトストアを使用する。
- スタックポインタ（すなわち、SS:RSP）は、割り込みの発生時に無条件でプッシュされる。従来の環境では、このプッシュ操作は、現行特権レベル（CPL）の変更に基づいて条件付きで行われる。
- CPL が変更された場合、新しい SS は NULL に設定される。
- IRET の動作の変更。
- 新しい割り込みスタックスイッチ機構の追加。
- 割り込みスタックフレームのアライメントの変更。

1.6.10.1. ゲート・ディスクリプタのフォーマット

割り込みディスクリプタ・テーブル（IDT）には、各割り込みベクタ用のサービスルーチンの位置の指定に使用されるゲート・ディスクリプタが格納される。従来の割り込みゲート・ディスクリプタは、命令ポインタ（EIP）用の 32 ビット・オフセットを提供する。64 ビット拡張技術は、命令ポインタ（RIP）用の 64 ビット・オフセットを提供するために、従来の割り込みゲートのサイズを 2 倍にし、8 バイトから 16 バイトに拡張する。割り込みゲート・ディスクリプタによって参照される 64 ビット RIP により、リニアアドレス空間内の任意の位置に、割り込みサービスルーチンを配置できる。

表 1-38. は、64 ビットモードの割り込みゲート・ディスクリプタとトラップ・ゲート・ディスクリプタのレイアウトを示している。表 1-39. は、64 ビット割り込み / トラップゲート内のフィールドの説明を示す。

表 1-38. IA-32e モードの割り込み / トラップゲート

31:16	15	14	13	12:8	7:3	2:0
予約済み						
オフセット 63:32						
オフセット 31:16	P	DPL		タイプ	000000	IST
ターゲット・セグメント・セレクタ	オフセット 15:0					

レガシーモードでは、IDT インデックスは、割り込みベクタに 8 を掛けて作成される。IA-32e モードでは、IDT インデックスは、割り込みベクタに 16 を掛けて作成される。64 ビットモードの割り込みゲートの最初の 8 バイト（バイト 7:0）は、従来の 32 ビット割り込みゲートによく似ているが、同一ではない。バイト 11:8 は、ターゲット RIP の上位 32 ビット（割り込みセグメント・オフセット）を正規形式で格納する。ソフトウェアが、正規形式でないターゲット RIP を使用して割り込みゲートを参照しようとすると、一般保護例外（#GP）が発生する。

割り込みゲートが参照するターゲット・コード・セグメントは、64 ビット・コード・セグメント（CS.L = 1、CS.D = 0）でなければならない。ターゲットが 64 ビット・コード・セグメントでない場合は、一般保護例外（#GP）が発生し、IDT ベクタ番号がエラーコードとして報告される。

表 1-39. IA-32e モードの割り込み / トラップゲートのフィールド

ダブルワード・オフセット	ゲート・フィールド	機能
3	+12(31:0)	未使用
2	+8(31:0)	オフセット・ビット 63:32
1	+4(31:16)	オフセット・ビット 31:16
	+4(15:13)	存在（P）とディスクリプタ特権レベル（DPL）
	+4(12:8)	64 ビット割り込みゲートまたはトラップゲートのタイプ（0Eh）
	+4(7:3)	未使用
	+4(2:0)	IST インデックス
0	+0(31:16)	ターゲット・セグメント・レジスタ
	+0(15:0)	オフセット 15:0

IA-32e モード（64 ビットモードと互換モード）で参照できるのは、64 ビットの割り込みゲートとトラップゲートに限られる。従来の 32 ビット割り込みゲートタイプまたはトラップ・ゲート・タイプ（0EH または 0FH）は、IA-32e モードでは、64 ビット割り込みゲートタイプとトラップ・ゲート・タイプとして再定義される。IA-32e モードには、32 ビット割り込みゲートタイプやトラップ・ゲート・タイプは存在しない。16 ビットの割り込みゲートまたはトラップゲート（06H または 07H）を参照すると、一般保護例外（#GP(0)）が発生する。

1.6.10.2. スタックフレーム

レガシーモードでは、IDTエントリのサイズ（16ビットまたは32ビット）によって、割り込みスタックフレームのプッシュのサイズが決まる。SS:ESPは、CPLが変更された場合にのみプッシュされる。64ビットモードでは、64ビットモードのゲートだけが参照可能であるため、割り込みスタックフレームのプッシュのサイズは8バイトに固定される。また、64 ビットモードでは、SS:RSP は、CPL の変更にかかわらず、無条件にプッシュされる。

エラーコードを除き、SS:RSP のプッシュ操作は、オペレーティング・システムに対し、すべての割り込みにおいて矛盾のない割り込みスタック・フレーム・サイズを無条件で提供する。INTn 命令または外部 INTR# 信号によって生成された割り込みを処理する割り込みサービスルーチンのエントリ・ポイントは、一貫性を保つために、追加のエラーコード・プレースホルダをプッシュできる。

レガシーモードでは、割り込みまたは例外が原因でスタックフレームがプッシュされる場合、スタックポインタのアライメントに制限はない。したがって、そのスタックフレームのアライメントや、割り込みハンドラによって実行される後続のプッシュのアライメントにも制限はない。IA-32eモードでは、スタックフレームをプッシュする前に、RSPのアライメントは16バイトに整列される。割り込みハンドラに移行するとき、スタックフレームそれ自体のアライメントも16バイトに合わされる。プロセッサは、割り込み発生時に、新しいRSPのアライメントを自由に変更できる。これは、元の（アライメントが合っていない可能性がある）RSPが、新たにアライメントされたスタック上に無条件でセーブされるからである。この元の RSP は、次の IRET によって自動的にリストアされる。

スタックのアライメント操作により、割り込みが再びイネーブルになる前に、例外フレームと割り込みフレームのアライメントを16バイトに合わせる。これにより、16バイト XMM レジスタの最適な格納ができるように、スタックをレイアウトする。これにより、割り込みハンドラは、XMM レジスタのセーブとリストアを実行する際に、アライメントの合っていないアクセス（MOVUPS）ではなく、16バイトにアライメントの合った高速のロードとストア（MOVAPS）を使用できる。x87 以上では浮動小数点演算用に SSE が重視されるため、XMM レジスタのセーブとリストアを効率的に行うのは、さらに重要になる。LMA = 1 の場合は、RSP のアライメントはいかなる場合にも実行されるが、このことはスタックスイッチや IST が使用されないカーネルモードの場合にのみ重要である。スタックスイッチまたはISTの場合は、おそらくOSが適切にアライメントされたRSP値をTSS内に入れているはずである。

1.6.10.3. IRET

IA-32eモードでは、IRETの意味が変更されている。IRETは、8バイトのオペランド・サイズで実行しなければならない。この必要条件を強制する機能は存在しないが、IA-32eモードのスタックは、IRETを必要とする標準的な操作では、8バイトのIRETオペランド・サイズで正常に動作するようにフォーマットされている。

64 ビットモードでは、SS:RSP は無条件にポップされる。互換モードとレガシーモードでは、SS:RSP は、CPL が変更された場合にのみポップされる。IA-32e モードでは、割り込みスタックフレームのプッシュは常に 8 バイト単位で行われるため、IRET は、スタックから8バイトアイテムをポップしなければならない。これを行うために、IRETの前に64 ビット・オペランド・サイズ・プリフィックスを追加する。ポップのサイズは、命令のアドレスサイズによって決まる。SS/ESP/RSP のサイズ調整は、スタックのサイズによって決まる。

IRET は、64 ビットモードで実行された場合にのみ、割り込みスタックフレームから SS:RSP を無条件でポップする。互換モードでは、IRET は、CPL が変更された場合にのみ、スタックから SS:RSP をポップする。これにより、従来のアプリケーションは、互換モードで IRET 命令を使用するときに正常に動作する。

IRET で終了する 64 ビット割り込みサービスルーチンは、ターゲット・コード・セグメントが 64 ビットモードで動作している場合や CPL = 0 の場合でも、割り込みスタックフレームから SS RSP を無条件でポップする。これは、元の割り込みが常に SS:RSP をプッシュするからである。

IA-32eモードでは、IRETは、特定の条件下でNULLのSSをロードすることを許される。ターゲット・モードが64ビットモードで、ターゲットCPL <> 3の場合、IRETは、SSにNULLセレクタがロードされることを許可する。割り込みや例外は、スタックスイッチ機構の一環として、TSSから新しいSSセレクタをフェッチしてGDTまたはLDTからそれに対応するディスクリプタをロードする代わりに、新しいSSをNULLに設定する。新しいSSセレクタをNULLに設定するのは、これ以降のネストされたfar転送からのリターンを適切に処理するためである。呼び出し先プロシージャに割り込みがかけられた場合は、NULL SSはスタックフレーム上にプッシュされる。次のIRETの実行時に、スタック上のNULL SSは、新しいSSディスクリプタをロードしないようにプロセッサに指示するフラグとして機能する。

1.6.10.4. スタックの切り替え

従来の IA-32 アーキテクチャは、割り込みに応答して自動的にスタックフレームを切り替えるメカニズムを備えている。64 ビットの拡張技術は、従来のスタックスイッチ機構を多少変更したものと、割り込みスタックテーブル（IST）と呼ばれる代替スタックスイッチ機構を実装している。

レガシーモードでは、従来の IA-32 スタックスイッチ機構がそのまま使用される。IA-32e モードでは、従来のスタックスイッチ機構が変更されている。割り込みを原因とする 64 ビットモードの特権レベル移行の一環としてスタックが切り替えられる場合、新しい SS ディスクリプタはロードされない。IA-32e モードでは、内側レベルの RSP だけが TSS からロードされる。新しい SS セレクタは強制的に NULL に設定され、SS セレクタの RPL フィールドは新しい CPL に設定される。新しい SS が NULL に設定されるのは、ネストされた far 転送（CALLF、INT、割り込み、例外）を処理するためである。元の SS と RSP は、新しいスタック上にセーブされる（表 1-40.）。次の IRET の実行時に、元の SS はスタックからポップされ、SS レジスタにロードされる。

要約すると、IA-32e モードのスタックスイッチの動作は従来のスタックスイッチとよく似ているが、新しい SS セレクタが TSS からロードされず、新しい SS が強制的に NULL に設定される点が異なる。

表 1-40. IA-32e モードのスタック・レイアウト（CPL の変更を伴う割り込みの実行後）

レガシーモード			64 ビットモード		
内容	バイト・オフセット			バイト・オフセット	内容
元の SS	+20			+40	元の SS
元の ESP	+16			+32	元の RSP
EFLAGS	+12			+24	RF フラグ
CS	+8			+16	CS
EIP	+4			+8	RIP
エラーコード	0	ESP	RSP	0	エラーコード
< 4 バイト >					< 8 バイト >

1.6.10.5. 割り込みスタックテーブル

IA-32e モードでは、上記の変更された従来のスタックスイッチ機構の代替手段として、新しい割り込みスタックテーブル（IST）機構を利用できる。この IST 機構がイネーブルになると、スタックは無条件に切り替えられる。IST 機構は、IDT エントリ内のフィールドによって、個々の割り込みベクタごとにイネーブルにできる。したがって、一部の割り込みベクタは変更された従来の機構を使用し、他の割り込みベクタは IST 機構を使用できる。IST 機構は、IA-32e モードでのみ利用可能である。この機構は、表 1-37. に示した 64 ビットモードの TSS の一部である。IST 機構が導入された主な目的

は、NMI、ダブルフォルト、マシンチェックなどの特定の割り込みが、常に確認済みの正常なスタック上で実行されるように保証する方法を提供することである。レガシーモードでは、割り込みは、IDT 内に置かれたタスクゲートを介して割り込みサービスルーチンにアクセスすることによって、タスクスイッチ機構を使用して確認済みの正常なスタックをセットアップできる。しかし、IA-32e モードでは、従来のタスクスイッチ機構をサポートしていない。IST 機構は、表 1-37. に示した 64 ビットモードのタスク・ステート・セグメント（TSS）の一部である。この機構は、TSS 内に置かれる最大 7 個の IST ポインタを提供する。これらのポインタは、表 1-38. に示すように、割り込みディスクリプタ・テーブル（IDT）内の割り込みゲート・ディスクリプタによって参照される。このゲート・ディスクリプタには、TSS の IST セクション内のオフセットを指定する、3 ビットの IST インデックス・フィールドが含まれる。

割り込みゲートの IST インデックスが 0 でない場合は、割り込みの発生時に、そのインデックスに対応する IST ポインタが RSP にロードされる。新しい SS セレクタは強制的に NULL に設定され、SS セレクタの RPL フィールドは新しい CPL に設定される。元の SS、RSP、RFLAGS、CS、RIP は、新しいスタック上にプッシュされる。その後、割り込み処理は通常どおり続行される。IST インデックスが 0 の場合は、上記の修正された従来のスタックスイッチ機構が使用される。

1.6.10.6. タスク優先度

64 ビット拡張技術は、APIC 仕様に定義された 15 の外部割り込み優先度クラスに基づいている。優先度クラス 1 が最低、クラス 15 が最高の優先度になる。外部割り込みとこれらの優先度クラスがどのように対応付けられるかは、プラットフォームによって異なる。オペレーティング・システムは、TPR を使用して、（一般的には優先度の低い）割り込みによって優先度の高いタスクが中断されないように、割り込みを一時的にブロックできる。これを行うには、ブロックされる割り込みのうち優先度が最も高いものに対応する値を、TPR にロードする。例えば、TPR に値 8（01000B）をロードすると、優先度が 8 またはそれより小さい割り込みはすべてブロックされ、優先度が 9 またはそれより大きい割り込みはすべて認識される。TPR に 0 をロードすると、すべての外部割り込みがイネーブルになる。TPR に 15（01111B）をロードすると、すべての外部割り込みがディスエーブルになる。リセット時には、TPR は 0 にクリアされる。

ソフトウェアは、MOV CR8 命令を使用して、TPR の読み取りと書き込みを実行できる。MOV CR8 命令の実行が完了すると、新しい優先度レベルが設定される。ソフトウェアは、TPR のロード後にシリアル化を強制する必要はない。

MOV CRn 命令を使用するには、特権レベル 0 が必要である。0 より大きい特権レベルで動作しているプログラムは、TPR の読み取りや書き込みを実行できない。このようなプログラムが TPR の読み取りや書き込みを行おうとすると、一般保護例外 #GP(0) が発生する。

TPR は、割り込みコントローラ（IC）から抽象された機能である。IC は、プロセッサへの外部割り込みの伝達に関する優先度の設定と管理を行う。IC は、APIC や 8259 などの外部デバイスでもかまわない。通常は、IC と TPR の優先度制御機構は、全く同一ではないにしてもよく似ている。ただし、IC はプロセッサ・モデルによって異なり、基礎となる優先度制御機構が変更される可能性がある。それに対して、TPR は 64 ビット・アーキテクチャの一部である。ソフトウェアは、この定義が変更されないことを前提にできる。表 1-41. は、TPR を示している。下位 4 ビットだけが使用される。その他の 60 ビットは予約済みであり、0 が書き込まれていなければならない。0 でない値が書き込まれると、一般保護例外 #GP(0) が発生する。

表 1-41. タスク・プライオリティ・レジスタ - CR8

63:4	3:0
予約済み	タスク・プライオリティ・レジスタ（TPR）

1.6.10.7. CR8 と APIC の相互作用

インテル EM64T を搭載した最初のプロセッサ・モデルには、多くの IA-32 プロセッサで使用されているものによく似た、ローカル・アドバンスト・プログラマブル割り込みコントローラ（APIC）が搭載されている。ローカル APIC のいくつかの要素は、アーキテクチャ上で定義されたタスク・プライオリティ・レジスタ（CR8.TPR）の動作に影響を与えている。

CR8 と APIC の重要な相互作用には、以下のものがある。

- プロセッサは、ローカル APIC がイネーブルの状態で起動される。
- CR8 が TPR として機能するためには、APIC がイネーブルになっていなければならない。CR8 と APIC の間の相互作用は次のとおりである。CR8 への書き込みは、APIC のタスク・プライオリティ・レジスタに反映される。
- APIC.TPR.7:4 = CR8.3:0、APIC.TPR.3:0 = 0 である。CR8 を読み取ると、APIC.TPR.7:4 が返され、64 ビットに 0 で拡張される。
- APIC.TPR と CR8 の直接の更新の間には、順序づけ機構は存在しない。オペレーティング・ソフトウェアは、APIC TPR の直接の更新または CR8 方式の TPR の更新のいずれかを使用し、両方を混在させないと予想される。ソフトウェアは、シリアル化命令（例えば、CPUID）を使用して、MOV CR8 と APIC へのストアの間で更新をシリアル化できる。

1.7. 64 ビットモードの一般的規則

64 ビットモードでは、命令と命令オペランドの変更に対して、以下の一般的規則が適用される。

- レガシーモードで、命令のオペランド・サイズ（16 ビットまたは 32 ビット）が、（CS.D とオーバライド・プリフィックスに依存する）実効オペランド・サイズに依存する場合、64 ビットモードでは、オペランド・サイズの選択肢が「16 ビットまたは 32 ビット」から「16 ビット、32 ビット、または 64 ビット」に拡張されるか、あるいは 64 ビット・オペランドをサポートするサイズに固定される。このような命令は、「64 ビットに変換される」と言われる。ただし、このような命令のバイト・オペランド・オペコードは変換されない。
- 上記のように、64 ビットに変換される命令のバイト・オペランド・オペコードは、通常は変換されない。これらの命令は、64 ビットモードでもバイトのみを操作する。
- レガシーモードで命令のオペランド・サイズが固定されている（CS.D とオーバライド・プリフィックスに依存しない）場合、64 ビットモードでは、オペランド・サイズは通常は同じサイズに固定される。例えば、CPUID は、レガシーモードと 64 ビットモードで同じサイズのオペランドを操作する。ただし、この規則にはいくつかの例外がある。
- 64 ビットモードで 32 ビット・オペランドを操作する場合、64 ビット GPR デスティネーション・レジスタの上位 32 ビットは 0 で拡張される。
- 64 ビットモードで 8 ビット（または 16 ビット）オペランドを操作する場合、64 ビット GPR デスティネーション・レジスタの上位 56 ビット（または 48 ビット）は変更されない。
- オペランド・サイズが 64 ビットの場合、シフト命令とローテート命令は、追加されたビット（合計で 6 ビット）を使用してシフトカウントとローテート・カウントを指定し、64 ビットのシフト操作とローテート操作を可能にする。
- 64 ビット即値を 64 ビット GPR に移動できることを除いて、即値オペランドの最大サイズは 32 ビットのままである。オペランド・サイズが 64 ビットの場合、即値は（使用される前に）64 ビットに符号拡張される。
- 分岐アドレスの変位は 8 ビットまたは 32 ビットのままであるが、変位は（使用される前に）64 ビットに符号拡張される。
- プロセッサが 64 ビットモードから互換モードまたはレガシーモードに移行する場合、プロセッサは 64 ビット GPR の上位 32 ビットを維持しない。互換モードまたはレガシーモードでは、GPR の下位 32 ビットだけが定義されている。

1.7.1. その他のガイドライン

- インテル® EM64T を搭載した最初のプロセッサ・モデルでは、64 ビットモードでオペランド・サイズ・プリフィックス（66H）と near 分岐を組み合わせて使用すると、プリフィックスは無視される。64 ビットモードでは、near 分岐は 32 ビット変位を使用する（命令ポインタは、32 ビット変位に基づく次の逐次命令オフセットになるリニアアドレスまで進められ、64 ビットに符号拡張される）。将来のプロセッサ・モデルでは、この動作は変更される可能性があるため、ソフトウェアはこの動作を前提としてはならない。
- 64 ビットモードでは、BSR/BSF のソース・オペランドの値が 0 の場合、これらのレジスタの上位 32 ビットはクリアされる。

2 命令セット・リファレンス（A-L）

第2章「命令セット・リファレンス（A-L）」では、IA-32命令をアルファベット順（A-L）に説明する。この続きについては、『インテル®エクステンデッド・メモリ64テクノロジ・ソフトウェア・デベロッパーズ・ガイド、第2巻』の第3章で説明する。

本章では、インテル®EM64Tで導入された新しい命令と、IA-32eモードにおける既存のIA-32命令セットについて説明する。説明の対象となるのは、汎用、x87 FPU、MMX®テクノロジ、SSE、SSE2、SSE3、システム用の命令である。命令の説明は、アルファベット順に整理してある。

各命令については、特定の記述形式により各オペランドの組み合わせを示している。この記述形式は、オペコード、必要なオペランド、説明で構成される。その他に、各命令について、以下のことも併記する。

- 命令とそのオペランドに関する説明
- 操作に関する説明
- EFLAGSレジスタのフラグに対する命令の影響の説明
- 発生する可能性がある例外の要約

2.1. 命令リファレンス・ページの読み方

本節では、本章の説明フォーマットについて説明する。これには、表記法と省略形式が含まれる。以下に、本章で使用されるフォーマットの例を示す。

2.1.1. 命令サマリテーブル

各命令の説明は、以下の例のような「命令サマリテーブル」で始まっている。

CMC - Complement Carry Flag（説明の例 ...）

オペコード	命令	64ビットモード	互換/レガシーモード	説明
F5	CMC	有効	有効	キャリーフラグの補数をとる。

各欄の内容については、以下で説明する。

2.1.1.1. 命令サマリテーブルのオペコード欄

「オペコード」欄には、各命令フォーマットに対して生成されるオブジェクト・コードを示す。可能な場合、コードは、メモリ内に現れるのと同じ順序で16進バイトとして示される。16進バイト以外のエントリの定義は、以下のとおりである。

- **REX.W** - オペランドのサイズや命令の意味に影響を与えるREXプリフィックスの使用を示す。REXプリフィックスと、その他の任意指定/必須の命令プリフィックスとの順序については、第1章の図1-1.を参照のこと。レガシー命令の動作を64ビットに拡張することになるREXプリフィックスの使用は、オペコード欄には明示的に記載されていない。
- **/digit** - 0から7までの数字で、命令のModR/Mバイトがr/m（レジスタまたはメモリ）オペランドだけを使用することを示す。regフィールドには、命令のオペコードを拡張する数字が入っている。
- **/r** - 命令のModR/Mバイトに、レジスタ・オペランドとR/Mオペランドの両方があることを示す。
- **cb**、**cw**、**cd**、**cp**、**co**、**ct** - オペコードの後に続く1バイト（cb）、2バイト（cw）、4バイト（cd）、6バイト（cp）、8バイト（co）、または10バイト（ct）の値であり、コード・オフセット、コード・セグメント・レジスタの新しい値を指定できる。
- **ib**、**iw**、**id**、**io** - オペコード、ModM/Rバイト、またはスケール・インデクッス・バイトの後に続く命令への1バイト（ib）、2バイト（iw）、4バイト（id）、または8バイト（io）の即値オペランドである。オペランドが符号付きかどうかは、オペコードによって決まる。すべてのワードおよびダブルワードが下位バイトから先に示される。
- **+rb**、**+rw**、**+rd**、**+ro** - 0から7までのレジスタコードであり、+符号の左側に現れる16進バイトに加算されて、単一のオペコード・バイトを構成する。すべてのレジスタコードを表2-1.に示す。
- **+i** - オペランドの1つがFPUレジスタスタックからのST(i)であるときに浮動小数点命令で使用される数値。数値i（0から7までの数値）は+符号の左側に現れる16進バイトに加算されて、単一のオペコード・バイトを構成する。

表2-1. レジスタコード

レジスタ	REX.R	reg フィールド	レジスタ	REX.R	reg フィールド	レジスタ	REX.R	reg フィールド	レジスタ	REX.R	reg フィールド
AL	0	0	AX	0	0	EAX	0	0	RAX	0	0
CL	0	1	CX	0	1	ECX	0	1	RCX	0	1
DL	0	2	DX	0	2	EDX	0	2	RDX	0	2
BL	0	3	BX	0	3	EBX	0	3	RBX	0	3

表 2-1. レジスタコード（続き）

レジスタ	REX.R	reg フィールド	レジスタ	REX.R	reg フィールド	レジスタ	REX.R	reg フィールド	レジスタ	REX.R	reg フィールド
AH	REX プリフィックスなし	4	SP	REX プリフィックスなし	4	ESP	REX プリフィックスなし	4	該当なし	該当なし	該当なし
CH	REX プリフィックスなし	5	BP	REX プリフィックスなし	5	EBP	REX プリフィックスなし	5	該当なし	該当なし	該当なし
DH	REX プリフィックスなし	6	SI	REX プリフィックスなし	6	ESI	REX プリフィックスなし	6	該当なし	該当なし	該当なし
BH	REX プリフィックスなし	7	DI	REX プリフィックスなし	7	EDI	REX プリフィックスなし	7	該当なし	該当なし	該当なし
SPL	任意の REX プリフィックス	4	SP	0	4	ESP	0	4	RSP	0	4
BPL	任意の REX プリフィックス	5	BP	0	5	EBP	0	5	RBP	0	5
SIL	任意の REX プリフィックス	6	SI	0	6	ESI	0	6	RSI	0	6
DIL	任意の REX プリフィックス	7	DI	0	7	EDI	0	7	RDI	0	7
R8L	1	0	R8W	1	0	R8D	1	0	R8	1	0
R9L	1	1	R9W	1	1	R9D	1	1	R9	1	1
R10L	1	2	R10W	1	2	R10D	1	2	R10	1	2
R11L	1	3	R11W	1	3	R11D	1	3	R11	1	3
R12L	1	4	R12W	1	4	R12D	1	4	R12	1	4
R13L	1	5	R13W	1	5	R13D	1	5	R13	1	5
R14L	1	6	R14W	1	6	R14D	1	6	R14	1	6
R15L	1	7	R15W	1	7	R15D	1	7	R15	1	7

2.1.1.2. 命令サマリテーブルの命令欄

「命令」欄は、ASM386プログラムに現れる場合の命令文の文法を示す。以下に、命令文内のオペランドを表現するために使用される記号の一覧を示す。

- **rel8** -命令の終りの前の128バイトから命令の終りの後の127バイトまでの範囲の相対アドレス。
- **rel16** および **rel32** - アセンブル結果の命令と同じコード・セグメント内の相対アドレス。記号 rel16 は、オペランド・サイズ属性が 16 ビットである命令に適用され、rel32 はオペランド・サイズ属性が 32 ビットである命令に適用される。
- **rel64** - アセンブル結果の命令と同じコード・セグメント内の相対アドレス。記号 rel64 は、オペランド・サイズ属性が 64 ビットである命令に適用される。
- **ptr16:16**、**ptr16:32**、**ptr16:64** - 一般的に命令のコード・セグメントとは異なるコード・セグメント内の far ポインタ。*16:16* の表記法は、ポインタの値が 2 つの部分からなっていることを示す。コロンの左側の値は、16 ビットのセレクタ、すなわちコード・セグメント・レジスタに送られる値である。右側の値はデスティネーション・セグメント内のオフセットに対応する。記号 ptr16:16 は命令のオペランド・サイズ属性が 16 ビットである場合に使用され、記号 ptr16:32 はオペランド・サイズ属性が 32 ビットである場合に使用され、記号 ptr16:64 はオペランド・サイズ属性が 64 ビットである場合に使用される。
- **r8** - 汎用バイトレジスタ AL、CL、DL、BL、AH、CH、DH、BH、R8l ～ R15l、BPL、SPL、DIL、SIL の中の 1 つ。
- **r16** - 汎用ワードレジスタ AX、CX、DX、BX、SP、BP、SI、DI、R8 ～ R15 の中の 1 つ。
- **r32** - 汎用ダブルワード・レジスタ EAX、ECX、EDX、EBX、ESP、EBP、ESI、EDI、R8D ～ R15D の中の 1 つ。
- **r64** - 汎用クワッドワード・レジスタ RAX、RBX、RCX、RDX、RDI、RSI、RBP、RSP、R8 ～ R15 の中の 1 つ。
- **imm8** - 即値バイト値。記号 imm8 は -128 から +127 までの符号付き数値である。imm8 がワードまたはダブルワードのオペランドと結合される命令については、即値は符号拡張されてワードまたはダブルワードを構成する。その場合は、ワードの上位バイトは即値の最上位ビットで埋められる。
- **imm16** - オペランド・サイズ属性が 16 ビットである命令に使用される即値ワード値。これは -32,768 から +32,767 までの数値である。
- **imm32** - オペランド・サイズ属性が 32 ビットである命令に使用される即値ダブルワード値。これには -2,147,483,648 から +2,147,483,647 までの数値を使用できる。
- **imm64** - オペランド・サイズ属性が 64 ビットである命令に使用される即値クワッドワード値。これには -9,223,372,036,854,775,808 から +9,223,372,036,854,775,807 までの数値を使用できる。

- **r/m8** - 汎用バイトレジスタ（AL、CL、DL、BL、AH、CH、DH、BH、R8l～R15l、BPL、SPL、DIL、SIL）か、メモリからのバイトかの内容であるバイト・オペランド。
- **r/m16** - オペランド・サイズ属性が16ビットである命令に使用される汎用ワード・レジスタ・オペランドまたはメモリ・オペランド。汎用ワード・レジスタは、AX、CX、DX、BX、SP、BP、SI、DI、R8～R15である。メモリの内容は、実効アドレスの計算によって与えられるアドレスにある。
- **r/m32** - オペランド・サイズ属性が32ビットである命令に使用される汎用ダブルワード・レジスタ・オペランドまたはメモリ・オペランド。汎用ダブルワード・レジスタは、EAX、ECX、EDX、EBX、ESP、EBP、ESI、EDI、R8D～R15Dである。メモリの内容は、実効アドレスの計算によって与えられるアドレスにある。
- **r/m64** - オペランド・サイズ属性が64ビットである命令に使用される汎用クワッドワード・レジスタ・オペランドまたはメモリ・オペランド。汎用クワッドワード・レジスタは、RAX、RBX、RCX、RDX、RDI、RSI、RBP、RSP、R8～R15である。メモリの内容は、実効アドレスの計算によって与えられるアドレスにある。
- **m** - 16、32、または64ビットのメモリ・オペランド。
- **m8** - 通常、変数名または配列名として表されるが、DS:(E) SIまたはES:(E) DIレジスタによってアドレス指定されるバイト・メモリ・オペランド。これはストリング命令およびXLAT命令に対してのみ使用される。
- **m16** - 通常、変数名または配列名として表されるが、DS:(E) SIまたはES:(E) DIレジスタによってアドレス指定されるワード・メモリ・オペランド。これはストリング命令に対してのみ使用される。
- **m32** - 通常、変数名または配列名として表されるが、DS:(E) SIまたはES:(E) DIレジスタによってアドレス指定されるダブルワード・メモリ・オペランド。これはストリング命令に対してのみ使用される。
- **m64** - クワッドワード・メモリ・オペランド。
- **m128** - ダブル・クワッドワード・メモリ・オペランド。これはSSEおよびSSE2に対してのみ使用される。
- **m16:16**、**m16:32**、**m16:64** - 2つの数値からなるfarポインタを内容とするメモリ・オペランド。コロンの左側の数値はポインタのセグメント・セレクタに対応し、右側の数値はポインタのオフセットに対応する。
- **m16&32**、**m16&16**、**m32&32**、**m16&64** - サイズがアンパーサンド記号（&）の左側と右側に示されるデータ項目の対からなるメモリ・オペランド。すべてのメモリアドレス指定モードが可能である。m16&16およびm32&32オペランドはBOUND命令で使用され、配列インデックスの上限と下限を内容とするオペランドを提供する。m16&32 オペランドはLIDT と LGDT で使用され、対応するGDTR およびIDTRレジスタのlimitフィールドにロードするワード、ベース・フィールドにロードするダブルワードを提供する。m16&64オペランドはLIDTとLGDTで使用され、

対応するGDTRおよびIDTRレジスタのlimitフィールドにロードするワード、ベース・フィールドにロードするクワッドワードを提供する。

- **moffs8**、**moffs16**、**moffs32**、**moffs64** - MOV命令の一部のバリエーションによって使用されるバイト、ワード、またはダブルワード型のシンプルメモリ変数（メモリ・オフセット）。実際のアドレスは、セグメント・ベースからのシンプル・オフセットによって与えられる。命令にはModR/Mバイトは使用されない。moffsの次の数値はセグメントのサイズであり、これは命令のアドレスサイズ属性によって決まる。
- **Sreg** - セグメント・レジスタ。セグメント・レジスタのビット割り当ては、ES=0、CS=1、SS=2、DS=3、FS=4、GS=5。
- **m32fp**、**m64fp**、**m80fp** - それぞれ単精度、倍精度、拡張倍精度浮動小数点数のメモリ・オペランドである。これらのシンボルにより、x87 FPU浮動小数点命令のオペランドとして使用される浮動小数点の値が指定される。
- **m16int**、**m32int**、**m64int** - それぞれワード、ダブルワード、クワッドワード整数のメモリ・オペランドである。これらのシンボルにより、x87 FPU整数命令のオペランドとして使用される整数が指定される。
- **ST**または**ST(0)** - FPUレジスタスタックの一番上の要素。
- **ST(i)** - FPUレジスタスタックの一番上からi番目の要素（$i \leftarrow 0 \sim 7$）。
- **mm** - MMX® テクノロジ・レジスタ。64ビットのMMXテクノロジ・レジスタは、MM0からMM7まで。
- **mm/m32** - MMXテクノロジ・レジスタの下位32ビットまたは32ビットのメモリ・オペランド。64ビットのMMXテクノロジ・レジスタは、MM0からMM7まで。メモリの内容は、実効アドレスの計算によって与えられるアドレスにある。
- **mm/m64** - MMXテクノロジ・レジスタまたは64ビットのメモリ・オペランド。64ビットのMMXテクノロジ・レジスタは、MM0からMM7まで。メモリの内容は、実効アドレスの計算によって与えられるアドレスにある。
- **xmm** - XMMレジスタ。128ビットのXMMレジスタは、XMM0からXMM15まで。
- **xmm/m32** - XMMレジスタまたは32ビットのメモリ・オペランド。128ビットのXMMレジスタは、XMM0からXMM15まで。メモリの内容は、実効アドレスの計算によって与えられるアドレスにある。
- **xmm/m64** - XMMレジスタまたは64ビットのメモリ・オペランド。128ビットのSIMD浮動小数点レジスタは、XMM0からXMM15まで。メモリの内容は、実効アドレスの計算によって与えられるアドレスにある。
- **xmm/m128** - XMMレジスタまたは128ビットのメモリ・オペランド。128ビットのXMMレジスタは、XMM0からXMM15まで。メモリの内容は、実効アドレスの計算によって与えられるアドレスにある。

2.1.1.3. 命令サマリテーブルの 64 ビットモード欄

「64 ビットモード」欄では、オペコード・シーケンスが 64 ビットモードでサポートされているかどうかを示す。この欄には、以下の表記法が使用される。

- 「有効」- サポートされている。
- 「無効」- サポートされていない。
- 「N.E.」- 命令構文がエンコード不可能であることを示す。64 ビットモードでは適用できないが、有効な命令シーケンスの一部を表す場合がある。
- 「N.P.」- 64 ビットモードではREXプリフィックスがレガシー命令に影響を与えないことを示す。
- 「N.I.」- 64 ビットモードではオペコードが新しい命令として扱われることを示す。
- 「N.S.」- 64 ビットモードではアドレス・オーバーライド・プリフィックスが必要な、未サポートの命令構文を示す。64 ビットモードでアドレス・オーバーライド・プリフィックスを使用すると、モデル固有の実行動作となる場合がある。

2.1.1.4. 命令サマリテーブルの互換 / レガシーモード欄

「互換 / レガシーモード」欄では、互換モードまたはレガシーの IA-32 モードでのオペコード・シーケンスに関する情報を示す。この欄には、以下の表記法が使用される。

- 「有効」- サポートされている。
- 「無効」- サポートされていない。
- 「N.E.」- 命令構文がエンコード不可能であることを示す。オペコード・シーケンスは、互換モードでもレガシーの IA-32 モードでも個別の命令として適用できないが、レガシー IA-32 命令の有効なシーケンスを表す場合がある。

2.1.1.5. 命令サマリテーブルの説明欄

「説明」欄では、命令フォーマットについて簡単に説明する。

2.1.2. 説明の項

命令サマリテーブルの後には、各命令について説明する複数の項が記載されている。「説明」の項では、命令の目的と必要なオペランドについて詳しく説明する。さらに、命令のフラグに対する影響についても説明する。

2.1.3. 操作の項

「操作」の項には、命令のアルゴリズムに関する説明を記載する（疑似コードで記述）。この疑似コードは、Algol または Pascal 言語に似た表記法を使用している。アルゴリズムは、以下の要素からなっている。

- コメントは記号の対「(*」および「*)」で囲まれる。
- 複合文は、それぞれ、if 文については IF、THEN、ELSE、FI、do 文については DO と OD、または case 文については CASE ... OF と ESAC などのキーワードで囲まれる。
- レジスタ名は暗黙にレジスタの内容を意味する。ブラケットで囲まれたレジスタ名は、暗黙に、アドレスがそのレジスタにストアされているロケーションの内容を意味する。例えば、ES:[DI] は、ES セグメント相対アドレスがレジスタ DI にストアされているロケーションの内容を示す。[SI] は、レジスタ SI にストアされている、SI レジスタのデフォルト・セグメント（DS）またはオーバライドされたセグメントの相対アドレスの内容を示す。
- 例えば (E)SI や (R)SI のように汎用レジスタ名の中で「E」や「R」が括弧で囲まれていたり、RSI のような 64 ビット・レジスタの定義が存在するときは、現在のアドレスサイズ属性が 16 である場合はオフセットが SI レジスタから読み込まれ、アドレスサイズ属性が 32 である場合は ESI レジスタから読み込まれ、アドレスサイズ属性が 64 である場合は 64 ビットの RSI レジスタから読み込まれることを示す。
- ブラケットはメモリ・オペランドに対しても使用され、その場合は、メモリ・ロケーションの内容がセグメント相対オフセットであることを意味する。例えば、[SRC] はソース・オペランドの内容がセグメント相対オフセットであることを示す。
- A ← B は、B の値が A に代入されることを示す。
- 記号 =、≠、≥、≤ は 2 つの値の比較に使用される関係演算子であり、それぞれ、「等しい」、「等しくない」、「より大きいか等しい」、「より小さいか等しい」を意味する。例えば、関係式 A ← B は、A の値が B に等しい場合は TRUE（真）であり、そうでない場合は FALSE（偽）である。
- 式「<< COUNT」と「>> COUNT」は、デスティネーション・オペランドがカウント・オペランドで示すビット数だけそれぞれ左または右にシフトされることを示す。

以下の識別子が、アルゴリズムの記述に使用される。

- **OperandSize** と **AddressSize** - OperandSize 識別子は、16 ビット、32 ビット、または 64 ビットの命令のオペランド・サイズ属性を表す。AddressSize 識別子は、16 ビット、32 ビット、または 64 ビットのアドレスサイズ属性を表す。例えば、以下の疑似コードは、オペランド・サイズ属性が使用される MOV 命令のフォーマットに依存することを示している。

```
IF instruction ← MOVW
    THEN OperandSize ← 16;
    ELSE
        IF instruction ← MOVD
            THEN OperandSize ← 32;
            ELSE
            IF instruction ← MOVQ
                THEN OperandSize ← 64;
            FI;
        FI;
FI;
```

これらの属性の決定方法の一般的ガイドラインについては、『IA-32 インテル® アーキテクチャ・ソフトウェア・デベロッパーズ・マニュアル、上巻』の第 3 章の「オペランド・サイズ属性とアドレスサイズ属性」を参照のこと。

- **StackAddrSize** - 命令に関連するスタックのアドレスサイズ属性を表す。この属性の値は 16、32、または 64 ビットである（『IA-32 インテル® アーキテクチャ・ソフトウェア・デベロッパーズ・マニュアル、上巻』の第 6 章「スタックアクセスにおけるアドレスサイズ属性」を参照のこと）。
- **SRC** - ソース・オペランドを表す。
- **DEST** - デスティネーション・オペランドを表す。

以下の関数が、アルゴリズムの記述に使用される。

- **ZeroExtend(** 値 **)** - 命令のオペランド・サイズ属性に合わせたゼロ拡張値を返す。例えば、オペランド・サイズ属性が 32 である場合は、-10 のバイト値をゼロ拡張すると、そのバイトは F6H からダブルワード値 000000F6H に変換される。ZeroExtend 関数に渡された値とオペランド・サイズ属性とが同じサイズである場合は、ZeroExtend はその値を変えずにそのまま返す。
- **SignExtend(** 値 **)** - 命令のオペランド・サイズ属性に合わせた符号拡張値を返す。例えば、オペランド・サイズ属性が 32 である場合は、内容が値 -10 であるバイトを符号拡張すると、そのバイトは F6H からダブルワード値 FFFFFFF6H に変換される。SignExtend 関数に渡された値とオペランド・サイズ属性が同じサイズである場合は、SignExtend はその値を変えずにそのまま返す。
- **SaturateSignedWordToSignedByte** - 符号付き 16 ビット値を符号付き 8 ビット値に変換する。符号付き 16 ビット値が -128 より小さい場合は、その値は飽和値 -128（80H）で表される。127 より大きい場合は、飽和値 127（7FH）で表される。
- **SaturateSignedDwordToSignedWord** - 符号付き 32 ビット値を符号付き 16 ビット値に変換する。符号付き 32 ビット値が -32768 より小さい場合は、その値は飽和値 -32768（8000H）で表される。32767 より大きい場合は、飽和値 32767（7FFFH）で表される。

- **SaturateSignedWordToUnsignedByte** -符号付き16ビット値を符号なし8ビット値に変換する。符号付き 16 ビット値が 0 より小さい場合は、その値は飽和値 0（00H）で表される。255 より大きい場合は、飽和値255（FFH）で表される。
- **SaturateToSignedByte** -演算の結果を符号付き8ビット値として表す。結果が-128より小さい場合は、その値は飽和値-128（80H）で表される。127 より大きい場合は、飽和値127（7FH）で表される。
- **SaturateToSignedWord** - 演算の結果を符号付き 16 ビット値として表す。結果が-32768 より小さい場合は、その値は飽和値-32768（8000H）で表される。32767 より大きい場合は、飽和値32767（7FFFH）で表される。
- **SaturateToUnsignedByte** - 演算の結果を符号なし8ビット値として表す。結果が0より小さい場合は、その値は飽和値0（00H）で表される。255 より大きい場合は、飽和値255（FFH）で表される。
- **SaturateToUnsignedWord** -演算の結果を符号なし16ビット値として表す。結果が0より小さい場合は、その値は飽和値0（00H）で表される。65535 より大きい場合は、飽和値65535（FFFFH）で表される。
- **LowOrderWord(DEST * SRC)** - ワード・オペランドにワード・オペランドを掛け、ダブルワードの結果の下位ワードをデスティネーション・オペランドにストアする。
- **HighOrderWord(DEST * SRC)** - ワード・オペランドにワード・オペランドを掛け、ダブルワードの結果の上位ワードをデスティネーション・オペランドにストアする。
- **Push(**値**)** - 値をスタックにプッシュする。プッシュされるバイト数は命令のオペランド・サイズ属性によって決まる。プッシュ操作の詳細については、本章の「PUSH - Push Word or Doubleword Onto the Stack」の「操作」の項を参照のこと。
- **Pop()** - 一番上のスタックから値を削除し、その値を返す。文 EAX ← Pop(); は、一番上のスタックから 32 ビット値を EAX に代入する。Pop は、オペランド・サイズ属性に応じて、ワード、ダブルワード、またはクワッドワードを返す。ポップ操作の詳細については、「POP - Pop a Value from the Stack」の「操作」の項を参照のこと。
- **PopRegisterStack** - FPUのST(0)レジスタを空としてマークし、FPUレジスタ・スタック・ポインタ（TOP）を1だけインクリメントする。
- **Switch-Tasks** - タスクスイッチを行う。
- **Bit(BitBase, BitOffset)** - メモリまたはレジスタ内の一連のビットであるビット・ストリング内の特定ビットの値を返す。ビットには、レジスタ内またはメモリバイト内で下位から上位に番号が割り振られている。ベース・オペランドがレジスタである場合、存在可能なオフセットの範囲は、レジスタのサイズに応じて 0 ～ 64 である。このオフセットは、指定されたレジスタ内のビットのアドレスを指定する。例として、図2-1. に関数Bit[EAX, 21]の図解を示す。

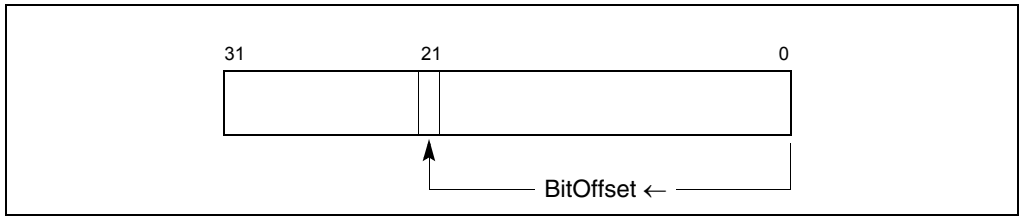

図 2-1. BIT[EAX,21] のビット・オフセット

BitBase がメモリアドレスである場合は、BitOffset の存在可能な範囲は -2G ビットから 2G ビットまでである。アドレス指定されるビットには、アドレス（BitBase + (BitOffset DIV 8)）のバイト内の番号（Offset MOD 8）が与えられる。ただし、DIV は負の無限大に対する丸めを伴う符号付き除算であり、MOD は正の数値を返す。図 2-2. に、この操作の図解を示す。

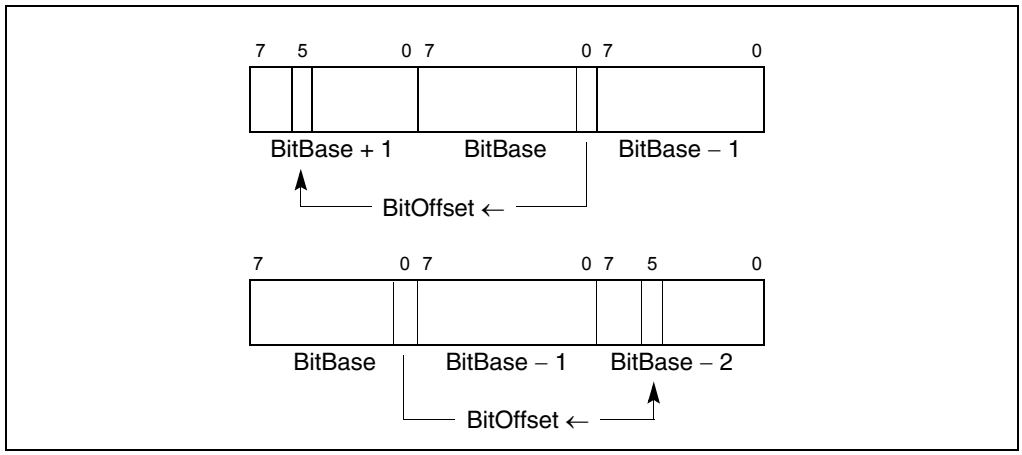

図 2-2. メモリビットのインデックス操作

レガシーモードでの命令の動作については、『IA-32 インテル® アーキテクチャ・ソフトウェア・デベロッパーズ・マニュアル、中巻 A、B : 命令セット・リファレンス』で説明されているので、本書では繰り返さない。

2.1.3.1. IA-32e モードでの操作

「IA-32e モードでの操作」の項では、レガシーの IA-32 モードとは異なる 64 ビットモードでの命令の動作について説明する。

2.1.4. 影響を受けるフラグ

「影響を受けるフラグ」の項には、命令の影響を受ける EFLAGS レジスタのフラグを示す。フラグはクリアされたときは 0 に等しく、セットされたときは 1 に等しい。算術演算および論理演算命令は、通常、一定の方法で値をステータス・フラグに代入する（詳細については『IA-32 インテル® アーキテクチャ・ソフトウェア・デベロッパーズ・マニュアル、上巻』の付録 A「EFLAGS クロス・リファレンス」を参照のこと）。「操作」の項では、非従来型の代入について説明している。未定義として示してあるフラグの値は、不確定な方法で変更される場合がある。記載されていないフラグは命令によって変更されない。

2.1.5. 影響を受ける FPU フラグ

浮動小数点命令については、「影響を受ける FPU フラグ」の項があり、そこで、各命令が FPU ステータス・ワードの 4 つの条件コードフラグにどのように影響を与えるかを説明している。

2.1.6. 保護モード例外

「保護モード例外」の項には、命令を保護モードで実行したときに発生する可能性がある例外およびそれらの例外の理由を示す。各例外は、番号記号（#）とその後に続く 2 文字からなるニーモニック、括弧で囲まれたオプションのエラーコードで表される。例えば、#GP(0) はエラーコード 0 を伴う一般保護例外を示す。表 2-2. に、各 2 文字のニーモニックと割り込みベクタ番号と例外名との対応を示す。これらの例外の詳細については、『IA-32 インテル® アーキテクチャ・ソフトウェア・デベロッパーズ・マニュアル、下巻』の第 5 章の「割り込みと例外の処理」を参照のこと。

アプリケーション・プログラマは、例外が発生した場合は、該当するオペレーティング・システムに付属のマニュアルを調べて対処すること。

表 2-2. 割り込みベクタ

ベクタ番号	名称	発生源	保護モード	実アドレスモード	仮想 8086 モード	64 ビットモード
0	#DE - 除算エラー	DIV および IDIV 命令	○	○	○	○
1	#DB - デバッグ	任意のコードまたはデータ参照	○	○	○	○
3	#BP - ブレークポイント	INT 3 命令	○	○	○	○
4	#OF - オーバーフロー	INTO 命令	○	○	○	○
5	#BR - BOUND 範囲外	BOUND 命令	○	○	○	予約
6	#UD - 無効オペコード（未定義オペコード）	UD2 命令または予約オペコード	○	○	○	○

表 2-2. 割り込みベクタ （続き）

ベクタ番号	名称	発生源	保護モード	実アドレスモード	仮想 8086 モード	64 ビットモード
7	#NM - デバイスなし（算術演算コプロセッサなし）	浮動小数点または WAIT/FWAIT 命令	○	○	○	○
8	#DF - ダブルフォルト	例外、NMI、または INTR を発生する可能性がある任意の命令	○	○	○	○
10	#TS - 無効 TSS	タスクスイッチまたは TSS アクセス	○	予約	○	○
11	#NP - セグメントなし	セグメント・レジスタのロードまたはシステム・セグメントのアクセス	○	予約	○	○
12	#SS - スタック・セグメント・フォルト	スタック操作および SS レジスタロード	○	○	○	○
13	#GP - 一般保護 *	任意のメモリ参照およびその他の保護チェック	○	○	○	○
14	#PF - ページフォルト	任意のメモリ参照	○	予約	○	○
16	#MF - 浮動小数点エラー（算術演算フォルト）	浮動小数点または WAIT/FWAIT 命令	○	○	○	○
17	#AC - アライメント・チェック	任意のメモリデータ参照	○	予約	○	○
18	#MC - マシンチェック	マシン・チェック・エラーはモデルに依存	○	○	○	○
19	#XF - SIMD 浮動小数点数値エラー	SSE および SSE2 浮動小数点命令	○	○	○	○
* 実アドレスモードにおいて、ベクタ 13 はセグメント・オーバーラン例外である。						

2.1.7. 実アドレスモード例外

「実アドレスモード例外」の項には、命令を実アドレスモードで実行したときに発生する可能性がある例外を示す。

2.1.8. 仮想 8086 モード例外

「仮想 8086 モード例外」の項には、命令を仮想 8086 モードで実行したときに発生する可能性がある例外を示す。

2.1.9. 浮動小数点例外

「浮動小数点例外」の項には、x87 FPU の浮動小数点命令を実行したときに発生する可能性がある例外を示す。これらの例外条件はすべて浮動小数点エラー例外（#MF、ベクタ番号 16）を発生させる。表 2-3. に、1 または 2 文字のニーモニックと例外名との対応を示す。これらの例外の詳細については、『IA-32 インテル® アーキテクチャ・ソフトウェア・デベロッパーズ・マニュアル、上巻』の第 8 章を参照のこと。

表 2-3. 例外名

ニーモニック	名称	発生源
#IS	無効浮動小数点演算 : スタック・オーバーフローもしくは アンダーフロー	x87FPU スタックのオーバーもしくは アンダーフロー
#IA	無効算術演算	無効な FPU 算術演算
#Z	ゼロでの浮動小数点除算	ゼロによる除算
#D	浮動小数点演算デノーマル・オペランド	デノーマル・オペランド
#O	浮動小数点数値オーバーフロー	結果がオーバーフロー
#U	浮動小数点数値アンダーフロー	結果がアンダーフロー
#P	浮動小数点不正確結果（精度）	不正確結果（精度）

2.1.10. SIMD 浮動小数点例外

「SIMD 浮動小数点例外」の項には、SSE および SSE2 浮動小数点命令を実行したときに発生する可能性がある例外を示す。これらの例外条件は、すべて SIMD 浮動小数点エラー例外（#XF、ベクタ番号 19）を発生させる。表 2-4. に、ニーモニックと例外名の対応を示す。これらの例外の詳細については、『IA-32 インテル® アーキテクチャ・ソフトウェア・デベロッパーズ・マニュアル、上巻』の第 11 章の「SSE と SSE2 の例外」を参照のこと。

表 2-4. 浮動小数点例外名

ニーモニック	名称	発生源
#I	浮動小数点無効演算	無効算術演算またはソース・オペランド
#Z	ゼロでの浮動小数点除算	ゼロによる除算
#D	浮動小数点演算デノーマル・オペランド	デノーマル・オペランド
#O	浮動小数点数値オーバーフロー	結果がオーバーフロー
#U	浮動小数点数値アンダーフロー	結果がアンダーフロー
#P	浮動小数点不正確結果	不正確結果（精度）

2.2. 命令リファレンス

本章の以降では、IA-32 の各命令の詳細について説明する。

AAA - ASCII Adjust After Addition

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
37	AAA	無効	有効	加算後に AL を ASCII 調整する。

影響を受けるフラグ

調整によって 10 進キャリーが生じた場合は、AF および CF フラグが 1 にセットされる。10 進キャリーが生じなかった場合は、両フラグが 0 にクリアされる。OF、SF、ZF、PF フラグは未定義。

IA-32e モードでの操作

64 ビットモードでは命令が無効である。

保護モード例外

なし。

実アドレスモード例外

なし。

仮想 8086 モード例外

なし。

互換モード例外

なし。

64 ビットモード例外

#UD	64 ビットモードの場合。

AAD - ASCII Adjust AX Before Division

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
D5 0A	AAD	無効	有効	除算前に AX を ASCII 調整する。
D5 *ib*	（ニーモニックなし）	無効	有効	除算前に AX を基数 *imm8* に調整する。

影響を受けるフラグ

SF、ZF、PF フラグが AL レジスタの 2 進値の結果に従ってセットされる。OF、AF、CF フラグは未定義。

IA-32e モードでの操作

64 ビットモードでは命令が無効である。

保護モード例外

なし。

実アドレスモード例外

なし。

仮想 8086 モード例外

なし。

互換モード例外

なし。

64 ビットモード例外

#UD	64 ビットモードの場合。

AAM - ASCII Adjust AX After Multiply

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
D4 0A	AAM	無効	有効	乗算後に AX を ASCII 調整する。
D4 *ib*	（ニーモニックなし）	無効	有効	乗算後に AX を基数 *imm8* に調整する。

影響を受けるフラグ

SF、ZF、PF フラグが AL レジスタの 2 進値の結果に従ってセットされる。OF、AF、CF フラグは未定義。

IA-32e モードでの操作

64 ビットモードでは命令が無効である。

保護モード例外

#DE　　即値として 0 を使用した場合。

実アドレスモード例外

#DE　　即値として 0 を使用した場合。

仮想 8086 モード例外

#DE　　即値として 0 を使用した場合。

互換モード例外

#DE　　即値として 0 を使用した場合。

64 ビットモード例外

#UD　　64 ビットモードの場合。

AAS - ASCII Adjust AL After Subtraction

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
3F	AAS	無効	有効	減算後に AL を ASCII 調整する。

影響を受けるフラグ

10進ボローが生じた場合は、AFおよびCFフラグが1にセットされ、10進ボローが生じなかった場合は両フラグが0にクリアされる。OF、SF、ZF、PFフラグは未定義。

IA-32e モードでの操作

64 ビットモードでは命令が無効である。

保護モード例外

なし。

実アドレスモード例外

なし。

仮想 8086 モード例外

なし。

互換モード例外

なし。

64 ビットモード例外

#UD	64 ビットモードの場合。

ADC - Add with Carry

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
14 *ib*	ADC AL,*imm8*	有効	有効	キャリーを加えて *imm8* を AL に加算する。
15 *iw*	ADC AX,*imm16*	有効	有効	キャリーを加えて *imm16* を AX に加算する。
15 *id*	ADC EAX,*imm32*	有効	有効	キャリーを加えて *imm32* を EAX に加算する。
REX.W + 15 *id*	ADC RAX,*imm32*	有効	N.E.	キャリーを加えて、64 ビットに符号拡張された *imm32* を RAX に加算する。
80 /2 *ib*	ADC *r/m8*,*imm8*	有効	有効	キャリーを加えて *imm8* を *r/m8* に加算する。
REX + 80 /2 *ib*	ADC *r/m8**,*imm8*	有効	N.E.	キャリーを加えて *imm8* を *r/m8* に加算する。
81 /2 *iw*	ADC *r/m16,imm16*	有効	有効	キャリーを加えて *imm16* を *r/m16* に加算する。
81 /2 *id*	ADC *r/m32,imm32*	有効	有効	CF を加えて *imm32* を *r/m32* に加算する。
REX.W + 81 /2 *id*	ADC *r/m64,imm32*	有効	N.E.	CF を加えて、64 ビットに符号拡張された *imm32* を *r/m64* に加算する。
83 /2 *ib*	ADC *r/m16,imm8*	有効	有効	CF を加えて符号拡張 *imm8* を *r/m16* に加算する。
83 /2 *ib*	ADC *r/m32,imm8*	有効	有効	CF を加えて符号拡張 *imm8* を *r/m32* に加算する。
REX.W + 83 /2 *ib*	ADC *r/m64,imm8*	有効	N.E.	CF を加えて符号拡張 *imm8* を *r/m64* に加算する。
10 /*r*	ADC *r/m8,r8*	有効	有効	キャリーを加えてバイトレジスタを *r/m8* に加算する。
REX + 10 /*r*	ADC *r/m8**,*r8**	有効	N.E.	キャリーを加えてバイトレジスタを *r/m64* に加算する。
11 /*r*	ADC *r/m16,r16*	有効	有効	キャリーを加えて *r16* を *r/m16* に加算する。
11 /*r*	ADC *r/m32,r32*	有効	有効	CF を加えて *r32* を *r/m32* に加算する。
REX.W + 11 /*r*	ADC *r/m64,r64*	有効	N.E.	CF を加えて *r64* を *r/m64* に加算する。
12 /*r*	ADC *r8,r/m8*	有効	有効	キャリーを加えて *r/m8* をバイトレジスタに加算する。
REX + 12 /*r*	ADC *r8**,*r/m8**	有効	N.E.	キャリーを加えて *r/m64* をバイトレジスタに加算する。
13 /*r*	ADC *r16,r/m16*	有効	有効	キャリーを加えて *r/m16* を *r16* に加算する。
13 /*r*	ADC *r32,r/m32*	有効	有効	CF を加えて *r/m32* を *r32* に加算する。
REX.W + 13 /*r*	ADC *r64,r/m64*	有効	N.E.	CF を加えて *r/m64* を *r64* に加算する。

* 64 ビットモードでは、REX プリフィックスを使用する場合、バイトレジスタ AH、BH、CH、DH にアクセスするように r/m8 をコード化することはできない。1.4.2.2. 節も参照のこと。

影響を受けるフラグ

OF、SF、ZF、AF、CF、PF フラグが結果に従ってセットされる。

IA-32e モードでの操作

64 ビットに拡張される。
デフォルトの操作サイズは 32 ビットである。
新しいレジスタ R8 ～ R15 へのアクセスが可能である。

保護モード例外

#GP(0)	デスティネーションが書き込み不可能なセグメントにある場合。
	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタを使用してメモリにアクセスしたが、その内容が NULL セグメント・セレクタであった場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

ADD - Add

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
04 *ib*	ADD AL,*imm8*	有効	有効	*imm8* を AL に加算する。
05 *iw*	ADD AX,*imm16*	有効	有効	*imm16* を AX に加算する。
05 *id*	ADD EAX,*imm32*	有効	有効	*imm32* を EAX に加算する。
REX.W + 05 *id*	ADD RAX,*imm32*	有効	N.E.	64 ビットに符号拡張された *imm32* を RAX に加算する。
80 /0 *ib*	ADD *r/m8,imm8*	有効	有効	*imm8* を *r/m8* に加算する。
REX + 80 /0 *ib*	ADD *r/m8*,imm8*	有効	N.E.	符号拡張 *imm8* を *r/m64* に加算する。
81 /0 *iw*	ADD *r/m16,imm16*	有効	有効	*imm16* を *r/m16* に加算する。
81 /0 *id*	ADD *r/m32,imm32*	有効	有効	*imm32* を *r/m32* に加算する。
REX.W + 81 /0 *id*	ADD *r/m64,imm32*	有効	N.E.	64 ビットに符号拡張された *imm32* を *r/m64* に加算する。
83 /0 *ib*	ADD *r/m16,imm8*	有効	有効	符号拡張 *imm8* を *r/m16* に加算する。
83 /0 *ib*	ADD *r/m32,imm8*	有効	有効	符号拡張 *imm8* を *r/m32* に加算する。
REX.W + 83 /0 *ib*	ADD *r/m64,imm8*	有効	N.E.	符号拡張 *imm8* を *r/m64* に加算する。
00 /*r*	ADD *r/m8,r8*	有効	有効	*r8* を *r/m8* に加算する。
REX + 00 /*r*	ADD *r/m8*,r8**	有効	N.E.	*r8* を *r/m8* に加算する。
01 /*r*	ADD *r/m16,r16*	有効	有効	*r16* を *r/m16* に加算する。
01 /*r*	ADD *r/m32,r32*	有効	有効	r32 を *r/m32* に加算する。
REX.W + 01 /*r*	ADD *r/m64,r64*	有効	N.E.	r64 を *r/m64* に加算する。
02 /*r*	ADD *r8,r/m8*	有効	有効	*r/m8* を *r8* に加算する。
REX + 02 /*r*	ADD *r8*,r/m8**	有効	N.E.	*r/m8* を *r8* に加算する。
03 /*r*	ADD *r16,r/m16*	有効	有効	*r/m16* を *r16* に加算する。
03 /*r*	ADD *r32,r/m32*	有効	有効	*r/m32* を *r32* に加算する。
REX.W + 03 /*r*	ADD *r64,r/m64*	有効	N.E.	*r/m64* を *r64* に加算する。

* 64 ビットモードでは、REX プリフィックスを使用する場合、バイトレジスタ AH、BH、CH、DH にアクセスするように r/m8 をコード化することはできない。1.4.2.2. 節も参照のこと。

影響を受けるフラグ

OF、SF、ZF、AF、CF、PF フラグが結果に従ってセットされる。

IA-32e モードでの操作

命令が 64 ビットに拡張される。
デフォルトの操作サイズは 32 ビットである。
新しいレジスタ R8 ～ R15 へのアクセスが可能である。

保護モード例外

#GP(0)	デスティネーションが書き込み不可能なセグメントにある場合。
	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタを使用してメモリにアクセスしたが、その内容が NULL セグメント・セレクタであった場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

ADDPD - Add Packed Double-Precision Floating-Point Values

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
66 0F 58 /r	ADDPD *xmm1, xmm2/m128*	有効	有効	*xmm2/m128* と *xmm1* のパックド倍精度浮動小数点値を加算して、結果を xmm1 に格納する。

IA-32e モードでの操作

XMM8 ～ XMM15 レジスタへのアクセスが可能である。

SIMD 浮動小数点例外

オーバーフロー、アンダーフロー、無効、精度、デノーマル。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

実アドレスモード例外

#GP(0)	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

ADDPS - Add Packed Single-Precision Floating-Point Values

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
0F 58 /r	ADDPS *xmm1*, *xmm2/m128*	有効	有効	*xmm2/m128* と *xmm1* のパックド単精度浮動小数点値を加算して、結果を xmm1 に格納する。

IA-32e モードでの操作

XMM8 ～ XMM15 レジスタへのアクセスが可能である。

SIMD 浮動小数点例外

オーバーフロー、アンダーフロー、無効、精度、デノーマル。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。

実アドレスモード例外

#GP(0)	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。

ADDSD - Add Scalar Double-Precision Floating-Point Values

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
F2 0F 58 /r	ADDSD *xmm1, xmm2/m64*	有効	有効	*xmm2/m64* と *xmm1* の下位の倍精度浮動小数点値を加算して、結果を xmm1 に格納する。

IA-32e モードでの操作

XMM8 ～ XMM15 レジスタへのアクセスが可能である。

SIMD 浮動小数点例外

オーバーフロー、アンダーフロー、無効、精度、デノーマル。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

ADDSS - Add Scalar Single-Precision Floating-Point Values

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
F3 0F 58 /r	ADDSS *xmm1*, *xmm2/m32*	有効	有効	*xmm2/m32* と *xmm1* の下位の単精度浮動小数点値を加算して、結果を xmm1 に格納する。

IA-32e モードでの操作

XMM8 ～ XMM15 レジスタへのアクセスが可能である。

SIMD 浮動小数点例外

オーバーフロー、アンダーフロー、無効、精度、デノーマル。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

ADDSUBPD - Packed Double-Precision Floating-Point Add/Subtract

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
66 0F D0 /r	ADDSUBPD *xmm1, xmm2/m128*	有効	有効	*xmm2/m128* と *xmm1* の倍精度浮動小数点値を加算 / 減算する。

IA-32e モードでの操作

XMM8 ～ XMM15 レジスタへのアクセスが可能である。

SIMD 浮動小数点例外

オーバーフロー、アンダーフロー、無効、精度、デノーマル。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
#GP(0)	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE3 が 0 の場合。

実アドレスモード例外

#GP(0)	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE3 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE3 が 0 の場合。

ADDSUBPS - Packed Single-Precision Floating-Point Add/Subtract

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
F2 0F D0 /r	ADDSUBPS *xmm1, xmm2/m128*	有効	有効	*xmm2/m128* と *xmm1* の単精度浮動小数点値を加算 / 減算する。

IA-32e モードでの操作

XMM8 ～ XMM15 レジスタへのアクセスが可能である。

SIMD 浮動小数点例外

オーバーフロー、アンダーフロー、無効、精度、デノーマル。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
#GP(0)	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE3 が 0 の場合。

実アドレスモード例外

#GP(0)	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE3 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE3 が 0 の場合。

AND - Logical AND

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
24 *ib*	AND AL,*imm8*	有効	有効	AL と *imm8* との AND をとる。
25 *iw*	AND AX,*imm16*	有効	有効	AX と *imm16* との AND をとる。
25 *id*	AND EAX,*imm32*	有効	有効	EAX と *imm32* との AND をとる。
REX.W + 25 *id*	AND RAX,*imm32*	有効	N.E.	RAX と 64 ビットに符号拡張された *imm32* との AND をとる。
80 /4 *ib*	AND *r/m8,imm8*	有効	有効	*r/m8* と *imm8* との AND をとる。
REX + 80 /4 *ib*	AND *r/m8*,imm8*	有効	N.E.	*r/m64* と *imm8*（符号拡張）との AND をとる。
81 /4 *iw*	AND *r/m16,imm16*	有効	有効	*r/m16* と *imm16* との AND をとる。
81 /4 *id*	AND *r/m32,imm32*	有効	有効	*r/m32* と *imm32* との AND をとる。
REX.W + 81 /4 *id*	AND *r/m64,imm32*	有効	N.E.	*r/m64* と 64 ビットに符号拡張された *imm32* との AND をとる。
83 /4 *ib*	AND *r/m16,imm8*	有効	有効	*r/m16* と *imm8*（符号拡張）との AND をとる。
83 /4 *ib*	AND *r/m32,imm8*	有効	有効	*r/m32* と *imm8*（符号拡張）との AND をとる。
REX.W + 83 /4 *ib*	AND *r/m64,imm8*	有効	N.E.	*r/m64* と *imm8*（符号拡張）との AND をとる。
20 */r*	AND *r/m8,r8*	有効	有効	*r/m8* と *r8* との AND をとる。
REX + 20 */r*	AND *r/m8*,r8**	有効	N.E.	*r/m64* と *r8*（符号拡張）との AND をとる。
21 */r*	AND *r/m16,r16*	有効	有効	*r/m16* と *r16* との AND をとる。
21 */r*	AND *r/m32,r32*	有効	有効	*r/m32* と *r32* との AND をとる。
REX.W + 21 */r*	AND *r/m64,r64*	有効	N.E.	*r/m64* と *r32* との AND をとる。
22 */r*	AND *r8,r/m8*	有効	有効	*r8* と *r/m8* との AND をとる。
REX + 22 */r*	AND *r8*,r/m8**	有効	N.E.	*r/m64* と *r8*（符号拡張）との AND をとる。
23 */r*	AND *r16,r/m16*	有効	有効	*r16* と *r/m16* との AND をとる。
23 */r*	AND *r32,r/m32*	有効	有効	*r32* と *r/m32* との AND をとる。
REX.W + 23 */r*	AND *r64,r/m64*	有効	N.E.	*r64* と *r/m64* との AND をとる。

* 64 ビットモードでは、REX プリフィックスを使用する場合、バイトレジスタ AH、BH、CH、DH にアクセスするように r/m8 をコード化することはできない。1.4.2.2. 節も参照のこと。

影響を受けるフラグ

OF および CF フラグがクリアされ、SF、ZF、PF フラグが結果に従ってセットされる。AF フラグの状態は未定義。

IA-32e モードでの操作

64 ビットに拡張される。
デフォルトの操作サイズは32 ビットである。
新しいレジスタ R8 ～ R15 へのアクセスが可能である。

保護モード例外

#GP(0)	デスティネーション・オペランドの指示先が書き込み不可能なセグメントの場合。
	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタの内容が NULL セグメント・セレクタの場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

64 ビットモード例外

#GP(0)	メモリアドレスが非標準形式の場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	デスティネーションが書き込み不可能なセグメントにある場合。 メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

ANDPD - Bitwise Logical AND of Packed Double-Precision Floating-Point Values

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
66 0F 54 /r	ANDPD *xmm1*, *xmm2/m128*	有効	有効	*xmm2/m128* と *xmm1* のビット単位の AND（論理積）演算を実行する。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。

SIMD 浮動小数点例外

なし。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#UD	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

実アドレスモード例外

#GP(0)	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#UD	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#UD	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

ANDPS - Bitwise Logical AND of Packed Single-Precision Floating-Point Values

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
0F 54 /r	ANDPS *xmm1*, *xmm2/m128*	有効	有効	*xmm2/m128* と *xmm1* のビット単位の AND（論理積）演算を実行する。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。

SIMD 浮動小数点例外

なし。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#UD	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。

実アドレスモード例外

#GP(0)	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#UD	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#UD	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。

ANDNPD - Bitwise Logical AND NOT of Packed Double-Precision Floating-Point Values

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
66 0F 55 /r	ANDNPD *xmm1*, *xmm2/m128*	有効	有効	*xmm2/m128* と *xmm1* のビット単位の AND NOT（否定論理積）演算を実行する。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。

SIMD 浮動小数点例外

なし。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#UD	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

実アドレスモード例外

#GP(0)	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#UD	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#UD	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

ANDNPS - Bitwise Logical AND NOT of Packed Single-Precision Floating-Point Values

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
0F 55 /r	ANDNPS *xmm1*, *xmm2/m128*	有効	有効	*xmm2/m128* と *xmm1* のビット単位の AND NOT（否定論理積）演算を実行する。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。

SIMD 浮動小数点例外

なし。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#UD	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。

実アドレスモード例外

#GP(0)	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#UD	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#UD	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。

ARPL - Adjust RPL Field of Segment Selector

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
63 /*r*	ARPL *r/m16,r16*	N. I.	有効	*r/m16* の RPL を *r16* の RPL 以上に調整する。

IA-32e モードでの操作

64 ビットモードでは命令が無効である。オペコードは MOVSXD 命令の一部として扱われる。

保護モード例外

#GP(0)	デスティネーションが書き込み不可能なセグメントにある場合。
	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタを使用してメモリにアクセスしたが、その内容が NULL セグメント・セレクタであった場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#UD	実アドレスモードでは、ARPL 命令は認識されない。

仮想 8086 モード例外

#UD	仮想 8086 モードでは、ARPL 命令は認識されない。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

なし。

BOUND - Check Array Index Against Bounds

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
62 /*r*	BOUND *r16, m16&16*	無効	有効	*r16*（配列インデックス）が *m16&16* の指定範囲内かどうかを確認する。
62 /*r*	BOUND *r32, m32&32*	無効	有効	*r32*（配列インデックス）が *m16&16* の指定範囲内かどうかを確認する。

影響を受けるフラグ

なし。

IA-32e モードでの操作

64 ビットモードでは命令が無効である。

保護モード例外

#BR	範囲テストが失敗した場合。
#UD	第 2 オペランドがメモリ・ロケーションでない場合。
#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタの内容が NULL セグメント・セレクタの場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#BR	範囲テストが失敗した場合。
#UD	第 2 オペランドがメモリ・ロケーションでない場合。
#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。

仮想 8086 モード例外

#BR	範囲テストが失敗した場合。
#UD	第 2 オペランドがメモリ・ロケーションでない場合。
#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#UD	64 ビットモードの場合。

BSF - Bit Scan Forward

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
0F BC	BSF *r16,r/m16*	有効	有効	*r/m16* を順方向にビットスキャンする。
0F BC	BSF *r32,r/m32*	有効	有効	*r/m32* を順方向にビットスキャンする。
REX.W + 0F BC	BSF *r64,r/m64*	有効	N.E.	*r/m64* を順方向にビットスキャンする。

影響を受けるフラグ

ソース・オペランドのすべてのビットが 0 の場合は、ZF フラグが 1 にセットされる。そうでない場合は、ZF フラグがクリアされる。CF、OF、SF、AF、PF フラグは未定義。

IA-32e モードでの操作

命令が 64 ビットに拡張される。
デフォルトの操作サイズは 32 ビットである。
新しいレジスタ R8 ～ R15 へのアクセスが可能である。

保護モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタの内容が NULL セグメント・セレクタの場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

BSR - Bit Scan Reverse

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
0F BD	BSR *r16,r/m16*	有効	有効	*r/m16* を逆方向にビットスキャンする。
0F BD	BSR *r32,r/m32*	有効	有効	*r/m32* を逆方向にビットスキャンする。
REX.W + 0F BD	BSR *r64,r/m64*	有効	N.E.	*r/m64* を逆方向にビットスキャンする。

影響を受けるフラグ

ソース・オペランドのすべてのビットが 0 の場合は、ZF フラグが 1 にセットされる。そうでない場合は、ZF フラグがクリアされる。CF、OF、SF、AF、PF フラグは未定義。

IA-32e モードでの操作

命令が 64 ビットに拡張される。
デフォルトの操作サイズは 32 ビットである。
新しいレジスタ R8 ～ R15 へのアクセスが可能である。

保護モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタの内容が NULL セグメント・セレクタの場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

BSWAP - Byte Swap

オペコード	命令	64 ビット モード	互換 / レガシー モード	説明
0F C8+*rd*	BSWAP *r32*	有効	有効	32 ビット・レジスタのバイト順序を逆にする。
REX.W + 0F C8+*rd*	BSWAP *r64*	有効	N.E.	64 ビット・レジスタのバイト順序を逆にする。

影響を受けるフラグ

なし。

IA-32e モードでの操作

命令が 64 ビットに拡張される。
デフォルトの操作サイズは 32 ビットである。
新しいレジスタ R8 ～ R15 へのアクセスが可能である。

保護モード例外

なし。

実アドレスモード例外

なし。

仮想 8086 モード例外

なし。

互換モード例外

なし。

64 ビットモード例外

なし。

BT - Bit Test

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
0F A3	BT *r/m16,r16*	有効	有効	選択したビットを CF フラグにストアする。
0F A3	BT *r/m32,r32*	有効	有効	選択したビットを CF フラグにストアする。
REX.W + 0F A3	BT *r/m64,r64*	有効	N.E.	選択したビットを CF フラグにストアする。
0F BA /4 *ib*	BT *r/m16,imm8*	有効	有効	選択したビットを CF フラグにストアする。
0F BA /4 *ib*	BT *r/m32,imm8*	有効	有効	選択したビットを CF フラグにストアする。
REX.W + 0F BA /4 *ib*	BT *r/m64,imm8*	有効	N.E.	選択したビットを CF フラグにストアする。

影響を受けるフラグ

CF フラグに選択されたビットの値がストアされる。OF、SF、ZF、AF、PF フラグは未定義。

IA-32e モードでの操作

命令が 64 ビットに拡張される。
デフォルトの操作サイズは 32 ビットである。
新しいレジスタ R8 ～ R15 へのアクセスが可能である。

保護モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタの内容が NULL セグメント・セレクタの場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

BTC - Bit Test and Complement

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
0F BB	BTC *r/m16,r16*	有効	有効	選択したビットを CF フラグにストアし、補数をとる。
0F BB	BTC *r/m32,r32*	有効	有効	選択したビットを CF フラグにストアし、補数をとる。
REX.W + 0F BB	BTC *r/m64,r64*	有効	N.E.	選択したビットを CF フラグにストアし、補数をとる。
0F BA /7 *ib*	BTC *r/m16,imm8*	有効	有効	選択したビットを CF フラグにストアし、補数をとる。
0F BA /7 *ib*	BTC *r/m32,imm8*	有効	有効	選択したビットを CF フラグにストアし、補数をとる。
REX.W + 0F BA /7 *ib*	BTC *r/m64,imm8*	有効	N.E.	選択したビットを CF フラグにストアし、補数をとる。

影響を受けるフラグ

CF フラグに選択されたビットの補数に変換される前の値がストアされる。OF、SF、ZF、AF、PF フラグは未定義。

IA-32e モードでの操作

命令が 64 ビットに拡張される。
デフォルトの操作サイズは 32 ビットである。
新しいレジスタ R8 ～ R15 へのアクセスが可能である。

保護モード例外

#GP(0)	デスティネーション・オペランドの指示先が書き込み不可能なセグメントの場合。
	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタの内容が NULL セグメント・セレクタの場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

BTR - Bit Test and Reset

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
0F B3	BTR *r/m16,r16*	有効	有効	選択したビットを CF フラグにストアし、クリアする。
0F B3	BTR *r/m32,r32*	有効	有効	選択したビットを CF フラグにストアし、クリアする。
REX.W + 0F B3	BTR *r/m64,r64*	有効	N.E.	選択したビットを CF フラグにストアし、クリアする。
0F BA /6 *ib*	BTR *r/m16,imm8*	有効	有効	選択したビットを CF フラグにストアし、クリアする。
0F BA /6 *ib*	BTR *r/m32,imm8*	有効	有効	選択したビットを CF フラグにストアし、クリアする。
REX.W + 0F BA /6 *ib*	BTR *r/m64,imm8*	有効	N.E.	選択したビットを CF フラグにストアし、クリアする。

影響を受けるフラグ

CF フラグに選択されたビットのクリアされる前の値がストアされる。OF、SF、ZF、AF、PF フラグは未定義。

IA-32e モードでの操作

命令が 64 ビットに拡張される。
デフォルトの操作サイズは 32 ビットである。
新しいレジスタ R8 ～ R15 へのアクセスが可能である。

保護モード例外

#GP(0)	デスティネーション・オペランドの指示先が書き込み不可能なセグメントの場合。
	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタの内容が NULL セグメント・セレクタの場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

BTS - Bit Test and Set

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
0F AB	BTS *r/m16,r16*	有効	有効	選択したビットを CF フラグにストアし、セットする。
0F AB	BTS *r/m32,r32*	有効	有効	選択したビットを CF フラグにストアし、セットする。
REX.W + 0F AB	BTS *r/m64,r64*	有効	N.E.	選択したビットを CF フラグにストアし、セットする。
0F BA /5 *ib*	BTS *r/m16,imm8*	有効	有効	選択したビットを CF フラグにストアし、セットする。
0F BA /5 *ib*	BTS *r/m32,imm8*	有効	有効	選択したビットを CF フラグにストアし、セットする。
REX.W + 0F BA /5 *ib*	BTS *r/m64,imm8*	有効	N.E.	選択したビットを CF フラグにストアし、セットする。

影響を受けるフラグ

CF フラグに選択されたビットのセットされる前の値がストアされる。OF、SF、ZF、AF、PF フラグは未定義。

IA-32e モードでの操作

命令が 64 ビットに拡張される。
デフォルトの操作サイズは 32 ビットである。
新しいレジスタ R8 ～ R15 へのアクセスが可能である。

保護モード例外

#GP(0)	デスティネーション・オペランドの指示先が書き込み不可能なセグメントの場合。
	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタの内容が NULL セグメント・セレクタの場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。

仮想 8086 モード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

CALL - Call Procedure

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
E8 *cw*	CALL *rel16*	N.S.	有効	相対 near コール、次の命令とディスプレースメント相対。64 ビットモードでは未サポート。
E8 *cd*	CALL *rel32*	有効	有効	相対 near コール、次の命令とディスプレースメント相対。64 ビットモードでは、64 ビットに符号拡張された 32 ビットのディスプレースメント。
FF /2	CALL *r/m16*	N.E.	有効	絶対間接 near コール、*r/m16* でアドレスを指定。
FF /2	CALL *r/m32*	N.E.	有効	絶対間接 near コール、*r/m32* でアドレスを指定。64 ビットモードでは、64 ビットに符号拡張された 32 ビットのディスプレースメント。
FF /2	CALL *r/m64*	有効	N.E.	絶対間接 near コール、*r/m64* でアドレスを指定。
9A *cd*	CALL *ptr16:16*	無効	有効	絶対 far コール、オペランドでアドレスを指定。
9A *cp*	CALL *ptr16:32*	無効	有効	絶対 far コール、オペランドでアドレスを指定。
FF /3	CALL *m16:16*	有効	有効	絶対間接 far コール、*m16:16* でアドレスを指定。 32 ビットモードの場合、セレクタの指示先がゲートであれば、RIP はゼロ拡張された 32 ビットのディスプレースメントであり、ゲートから取得される。指示先がゲートでなければ、RIP はゼロ拡張された 16 ビットのオフセットであり、命令で参照される far ポインタから取得される。
FF /3	CALL *m16:32*	有効	有効	64 ビットモードの場合、セレクタの指示先がゲートであれば、RIP は 64 ビットのディスプレースメントであり、ゲートから取得される。指示先がゲートでなければ、RIP はゼロ拡張された 32 ビットのオフセットであり、命令で参照される far ポインタから取得される。
FF /3	CALL *m16:64*	有効	N.E.	64 ビットモードの場合、セレクタの指示先がゲートであれば、RIP は 64 ビットのディスプレースメントであり、ゲートから取得される。指示先がゲートでなければ、RIP は 64 ビットのオフセットであり、命令で参照される far ポインタから取得される。

影響を受けるフラグ

タスクスイッチが行われた場合はすべてのフラグが影響を受け、タスクスイッチが行われなかった場合はどのフラグも影響を受けない。

64 ビットモードでの操作

命令が 64 ビットに拡張される。
デフォルトの操作サイズは 32 ビットである。
32 ビットのディスプレースメントは 64 ビットに符号拡張される。

保護モード例外

#GP(0)	デスティネーション・オペランド内のターゲット・オフセットが新しいコード・セグメントの範囲外の場合。
	デスティネーション・オペランド内のセグメント・セレクタが NULL の場合。
	ゲート内のコード・セグメント・セレクタが NULL の場合。
	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタを使用してメモリにアクセスしたが、その内容が NULL セグメント・セレクタであった場合。
#GP（セレクタ）	コード・セグメント、ゲート、または TSS のセレクタ・インデックスが記述子テーブルの範囲外の場合。
	デスティネーション・オペランドにあるセグメント・セレクタの指示先のセグメント記述子がコンフォーミング・コード・セグメント、非コンフォーミング・コード・セグメント、コールゲート、タスクゲート、またはタスク・ステート・セグメントのいずれのものでもなかった場合。
	非コンフォーミング・コード・セグメントの DPL が CPL に等しくないか、またはセグメントのセグメント・セレクタの RPL が CPL より大きい場合。
	コンフォーミング・コード・セグメントの DPL が CPL より大きい場合。
	コールゲート、タスクゲート、または TSS のセグメント記述子からの DPL が、CPL、またはコールゲート、タスクゲート、または TSS のセグメント・セレクタの RPL より小さい場合。
	コールゲートからのセグメント・セレクタのセグメント記述子が、そのコールゲートがコード・セグメントであることを示していない場合。
	コールゲートからのセグメント・セレクタが記述子テーブルの範囲外の場合。
	コールゲートから得られたコード・セグメントの DPL が CPL より大きい場合。
	TSS のセグメント・セレクタのローカル / グローバル・ビットがローカルとしてセットされている場合。
	TSS のセグメント記述子が、TSS がビジーであるか、または使用不可能であることを示している場合。
#SS(0)	スタックスイッチが行われなかったときに、リターンアドレス、パラメータ、またはスタック・セグメント・ポインタをスタックにプッシュした結果、スタック・セグメントの範囲を超えた場合。

	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#SS（セレクタ）	スタックスイッチが行われたときに、リターンアドレス、パラメータ、またはスタック・セグメント・ポインタをスタックにプッシュした結果、スタック・セグメントの範囲を超えた場合。
	スタックスイッチの一環として SS レジスタへのロードが行われようとするとき、指示先のセグメントが存在しないとマークされていた場合。
	スタックスイッチが行われたとき、リターンアドレス、パラメータ、またはスタック・セグメント・ポインタをストアするための余裕がスタック・セグメントにない場合。
#NP（セレクタ）	コード・セグメント、データ・セグメント、スタック・セグメント、コールゲート、タスクゲート、または TSS が存在しない場合。
#TS（セレクタ）	新しいスタック・セグメント・セレクタと ESP が TSS の終りを超えている場合。
	新しいスタック・セグメント・セレクタが NULL の場合。
	TSS 内の新しいスタック・セグメント・セレクタの RPL がアクセス先のコード・セグメントの DPL と等しくない場合。
	新しいスタック・セグメント用のスタック・セグメント記述子の DPL がコード・セグメント記述子の DPL と等しくない場合。
	新しいスタック・セグメントが書き込み可能なデータ・セグメントでない場合。
	スタック・セグメントのセグメント・セレクタ・インデックスが記述子テーブルの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	ターゲット・オフセットがコード・セグメントの範囲外の場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	ターゲット・オフセットがコード・セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

64 ビットモード例外と同じ。

64 ビットモード例外

#GP(0)	メモリアドレスが非標準の場合。
	デスティネーション・オペランド内のターゲット・オフセットが非標準の場合。
	デスティネーション・オペランド内のターゲット・オフセットが新しいコード・セグメントの範囲外の場合。
	デスティネーション・オペランド内のセグメント・セレクタが NULL の場合。
	64 ビットゲート内のコード・セグメント・セレクタが NULL の場合。
	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタを使用してメモリにアクセスしたが、その内容が NULL セグメント・セレクタであった場合。
#GP（セレクタ）	コード・セグメントまたは 64 ビット・コール・ゲートが記述子テーブルの範囲外の場合。
	コード・セグメントまたは 64 ビット・コール・ゲートが非標準領域に重複する場合。
	デスティネーション・オペランドにあるセグメント・セレクタの指示先のセグメント記述子がコンフォーミング・コード・セグメント、非コンフォーミング・コード・セグメント、64 ビット・コール・ゲートのいずれのものでもなかった場合。
	デスティネーション・オペランドにあるセグメント・セレクタの指示先のセグメント記述子がコード・セグメントであり、D ビットと L ビットの両方がセットされている場合。
	非コンフォーミング・コード・セグメントの DPL が CPL に等しくないか、またはセグメントのセグメント・セレクタの RPL が CPL より大きい場合。
	コンフォーミング・コード・セグメントの DPL が CPL より大きい場合。
	64 ビット・コール・ゲートからの DPL が、CPL、または 64 ビット・コール・ゲートの RPL より小さい場合。
	64 ビット・コール・ゲートの上位タイプ・フィールドが 0 でない場合。
	64 ビット・コール・ゲートからのセグメント・セレクタが記述子テーブルの範囲外の場合。
	64 ビット・コール・ゲートから得られたコード・セグメントの DPL が CPL より大きい場合。
	64 ビットゲートにあるセレクタの指示先のコード・セグメント記述子で、L ビットがセットされておらず、D ビットがクリアされていない場合。
	64 ビット・コール・ゲートからのセグメント・セレクタのセグメント記述子が、そのコールゲートがコード・セグメントであることを示していない場合。

#SS(0)	スタックスイッチが行われなかったときに、リターン・オフセットまたはCSセレクタをスタックにプッシュした結果、スタック・セグメントの範囲を超えた場合。 メモリ・オペランドの実効アドレスがSSセグメントの範囲外の場合。 スタックアドレスが非標準形式の場合。
#SS（セレクタ）	スタックスイッチが行われたときに、SSセレクタ、スタックポインタ、EFLAGS、CSセレクタ、オフセット、またはエラーコードの古い値をスタックにプッシュした結果、標準の境界に違反した場合。
#NP（セレクタ）	コード・セグメントまたは64ビット・コール・ゲートが存在しない場合。
#TS（セレクタ）	新しいRSPのロードがTSSの範囲を超えている場合。
#UD	（64ビットモードのみ）farコールがメモリ内の絶対アドレスを直接指定している場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが3のときにアライメントが合わないメモリ参照を行った場合。

CBW/CWDE/CDQE - Convert Byte to Word/Convert Word to Doubleword/Convert Doubleword to Quadword

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
98	CBW	有効	有効	AX ← AL の符号拡張
98	CWDE	有効	有効	EAX ← AX の符号拡張
REX.W + 98	CDQE	有効	N.E.	RAX ← EAX の符号拡張

影響を受けるフラグ

なし。

IA-32e モードでの操作

命令が 64 ビットに拡張される。
デフォルトの操作サイズはデスティネーション・レジスタのサイズである。

例外（すべての動作モード）

なし。

CDQ - Convert Double to Quad

「CWD/CDQ/CQQ - Convert Word to Doubleword/Convert Doubleword to Quadword/Convert Quadword to Double Quadword」を参照のこと。

CLC - Clear Carry Flag

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
F8	CLC	有効	有効	CF フラグをクリアする。

影響を受けるフラグ

CF フラグが 0 にクリアされる。OF、ZF、SF、AF、PF フラグは影響を受けない。

IA-32e モードでの操作

レガシーモードと同じ。

例外（すべての動作モード）

なし。

CLD - Clear Direction Flag

オペコード	命令	64 ビット モード	互換/ レガシー モード	説明
FC	CLD	有効	有効	DF フラグをクリアする。

影響を受けるフラグ

DF フラグが 0 にクリアされる。CF、OF、ZF、SF、AF、PF フラグは影響を受けない。

IA-32e モードでの操作

レガシーモードと同じ。

例外（すべての動作モード）

なし。

CLFLUSH - Flush Cache Line

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
0F AE /7	CLFLUSH *m8*	有効	有効	*m8* が入っているキャッシュ・ラインをフラッシュする。

IA-32e モードでの操作

レガシーモードと同じ。

同等のインテル® C/C++ コンパイラ組み込み関数

CLFLUSH　　void_mm_clflush(void const *p)

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#UD	CPUID 機能フラグ CLFSH が 0 の場合。

実アドレスモード例外

#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#UD	CPUID 機能フラグ CLFSH が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#UD	CPUID 機能フラグ CLFSH が 0 の場合。

CLI - Clear Interrupt Flag

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
FA	CLI	有効	有効	割り込みフラグをクリアする。割り込みフラグがクリアされているときは割り込みはディスエーブルにされる。

影響を受けるフラグ

CPL が IOPL に等しいか小さい場合は、IF フラグが 0 にクリアされる。そうでない場合、IF フラグは変わらない。EFLAGS レジスタ内のその他のフラグは影響を受けない。

IA-32e モードでの操作

レガシーモードと同じ。

保護モード例外

#GP(0)	CPL が現在のプログラムまたはプロシージャの IOPL より大きい（特権が小さい）場合。

実アドレスモード例外

なし。

仮想 8086 モード例外

#GP(0)	CPL が現在のプログラムまたはプロシージャの IOPL より大きい（特権が小さい）場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#GP(0)	CPL が現在のプログラムまたはプロシージャの IOPL より大きい（特権が小さい）場合。

CLTS - Clear Task-Switched Flag in CR0

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
0F 06	CLTS	有効	有効	CR0 の TS フラグをクリアする。

影響を受けるフラグ

CR0 レジスタ内の TS フラグがクリアされる。

IA-32e モードでの操作

レガシーモードと同じ。

保護モード例外

#GP(0)　　CPL が 0 より大きい場合。

実アドレスモード例外

なし。

仮想 8086 モード例外

#GP(0)　　CPL が 0 より大きい場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#GP(0)　　CPL が 0 より大きい場合。

CMC - Complement Carry Flag

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
F5	CMC	有効	有効	CF フラグの状態を反転する。

影響を受けるフラグ

CF フラグの内容がその元の値の補数になる。OF、ZF、SF、AF、PF フラグは影響を受けない。

IA-32e モードでの操作

レガシーモードと同じ。

例外（すべての動作モード）

なし。

CMOV*cc* - Conditional Move

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
0F 47 */r*	CMOVA *r16, r/m16*	有効	有効	より上（CF=0 および ZF=0）の場合転送する。
0F 47 */r*	CMOVA *r32, r/m32*	有効	有効	より上（CF=0 および ZF=0）の場合転送する。
REX.W + 0F 47 */r*	CMOVA *r64, r/m64*	有効	N.E.	より上（CF=0 および ZF=0）の場合転送する。
0F 43 */r*	CMOVAE *r16, r/m16*	有効	有効	より上か等しい（CF=0）場合転送する。
0F 43 */r*	CMOVAE *r32, r/m32*	有効	有効	より上か等しい（CF=0）場合転送する。
REX.W + 0F 43 */r*	CMOVAE *r64, r/m64*	有効	N.E.	より上か等しい（CF=0）場合転送する。
0F 42 */r*	CMOVB *r16, r/m16*	有効	有効	より下（CF=1）の場合転送する。
0F 42 */r*	CMOVB *r32, r/m32*	有効	有効	より下（CF=1）の場合転送する。
REX.W + 0F 42 */r*	CMOVB *r64, r/m64*	有効	N.E.	より下（CF=1）の場合転送する。
0F 46 */r*	CMOVBE *r16, r/m16*	有効	有効	より下か等しい（CF=1 または ZF=1）場合転送する。
0F 46 */r*	CMOVBE *r32, r/m32*	有効	有効	より下か等しい（CF=1 または ZF=1）場合転送する。
REX.W + 0F 46 */r*	CMOVBE *r64, r/m64*	有効	N.E.	より下か等しい（CF=1 または ZF=1）場合転送する。
0F 42 */r*	CMOVC *r16, r/m16*	有効	有効	キャリーがある（CF=1）場合転送する。
0F 42 */r*	CMOVC *r32, r/m32*	有効	有効	キャリーがある（CF=1）場合転送する。
REX.W + 0F 42 */r*	CMOVC *r64, r/m64*	有効	N.E.	キャリーがある（CF=1）場合転送する。
0F 44 */r*	CMOVE *r16, r/m16*	有効	有効	等しい（ZF=1）場合転送する。
0F 44 */r*	CMOVE *r32, r/m32*	有効	有効	等しい（ZF=1）場合転送する。
REX.W + 0F 44 */r*	CMOVE *r64, r/m64*	有効	N.E.	等しい（ZF=1）場合転送する。
0F 4F */r*	CMOVG *r16, r/m16*	有効	有効	より大きい（ZF=0 および SF=OF）場合転送する。
0F 4F */r*	CMOVG *r32, r/m32*	有効	有効	より大きい（ZF=0 および SF=OF）場合転送する。
REX.W + 0F 4F */r*	CMOVG *r64, r/m64*	有効	N.E.	より大きい（ZF=0 および SF=OF）場合転送する。
0F 4D */r*	CMOVGE *r16, r/m16*	有効	有効	より大きいか等しい（SF=OF）場合転送する。
0F 4D */r*	CMOVGE *r32, r/m32*	有効	有効	より大きいか等しい（SF=OF）場合転送する。
REX.W + 0F 4D */r*	CMOVGE *r64, r/m64*	有効	N.E.	より大きいか等しい（SF=OF）場合転送する。
0F 4C */r*	CMOVL *r16, r/m16*	有効	有効	より小さい（SF<>OF）場合転送する。

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
0F 4C */r*	CMOVL *r32, r/m32*	有効	有効	より小さい（SF<>OF）場合転送する。
REX.W + 0F 4C */r*	CMOVL *r64, r/m64*	有効	N.E.	より小さい（SF<>OF）場合転送する。
0F 4E */r*	CMOVLE *r16, r/m16*	有効	有効	より小さいか等しい（ZF=1 または SF<>OF）場合転送する。
0F 4E */r*	CMOVLE *r32, r/m32*	有効	有効	より小さいか等しい（ZF=1 または SF<>OF）場合転送する。
REX.W + 0F 4E */r*	CMOVLE *r64, r/m64*	有効	N.E.	より小さいか等しい（ZF=1 または SF<>OF）場合転送する。
0F 46 */r*	CMOVNA *r16, r/m16*	有効	有効	より上でない（CF=1 または ZF=1）場合転送する。
0F 46 */r*	CMOVNA *r32, r/m32*	有効	有効	より上でない（CF=1 または ZF=1）場合転送する。
REX.W + 0F 46 */r*	CMOVNA *r64, r/m64*	有効	N.E.	より上でない（CF=1 または ZF=1）場合転送する。
0F 42 */r*	CMOVNAE *r16, r/m16*	有効	有効	より上でなく等しくない（CF=1）場合転送する。
0F 42 */r*	CMOVNAE *r32, r/m32*	有効	有効	より上でなく等しくない（CF=1）場合転送する。
REX.W + 0F 42 */r*	CMOVNAE *r64, r/m64*	有効	N.E.	より上でなく等しくない（CF=1）場合転送する。
0F 43 */r*	CMOVNB *r16, r/m16*	有効	有効	より下でない（CF=0）場合転送する。
0F 43 */r*	CMOVNB *r32, r/m32*	有効	有効	より下でない（CF=0）場合転送する。
REX.W + 0F 43 */r*	CMOVNB *r64, r/m64*	有効	N.E.	より下でない（CF=0）場合転送する。
0F 47 */r*	CMOVNBE *r16, r/m16*	有効	有効	より下でなく等しくない（CF=0 および ZF=0）場合転送する。
0F 47 */r*	CMOVNBE *r32, r/m32*	有効	有効	より下でなく等しくない（CF=0 および ZF=0）場合転送する。
REX.W + 0F 47 */r*	CMOVNBE *r64, r/m64*	有効	N.E.	より下でなく等しくない（CF=0 および ZF=0）場合転送する。
0F 43 */r*	CMOVNC *r16, r/m16*	有効	有効	キャリーがない（CF=0）場合転送する。
0F 43 */r*	CMOVNC *r32, r/m32*	有効	有効	キャリーがない（CF=0）場合転送する。
REX.W + 0F 43 */r*	CMOVNC *r64, r/m64*	有効	N.E.	キャリーがない（CF=0）場合転送する。
0F 45 */r*	CMOVNE *r16, r/m16*	有効	有効	等しくない（ZF=0）場合転送する。
0F 45 */r*	CMOVNE *r32, r/m32*	有効	有効	等しくない（ZF=0）場合転送する。
REX.W + 0F 45 */r*	CMOVNE *r64, r/m64*	有効	N.E.	等しくない（ZF=0）場合転送する。
0F 4E */r*	CMOVNG *r16, r/m16*	有効	有効	より大きくない（ZF=1 または SF<>OF）場合転送する。
0F 4E */r*	CMOVNG *r32, r/m32*	有効	有効	より大きくない（ZF=1 または SF<>OF）場合転送する。

オペコード	命令	64 ビット モード	互換 / レガシー モード	説明
REX.W + 0F 4E */r*	CMOVNG *r64, r/m64*	有効	N.E.	より大きくない（ZF=1 または SF<>OF）場合転送する。
0F 4C */r*	CMOVNGE *r16, r/m16*	有効	有効	より大きくなく等しくない（SF<>OF）場合転送する。
0F 4C */r*	CMOVNGE *r32, r/m32*	有効	有効	より大きくなく等しくない（SF<>OF）場合転送する。
REX.W + 0F 4C */r*	CMOVNGE *r64, r/m64*	有効	N.E.	より大きくなく等しくない（SF<>OF）場合転送する。
0F 4D */r*	CMOVNL *r16, r/m16*	有効	有効	より小さくない（SF=OF）場合転送する。
0F 4D */r*	CMOVNL *r32, r/m32*	有効	有効	より小さくない（SF=OF）場合転送する。
REX.W + 0F 4D */r*	CMOVNL *r64, r/m64*	有効	N.E.	より小さくない（SF=OF）場合転送する。
0F 4F */r*	CMOVNLE *r16, r/m16*	有効	有効	より小さくなく等しくない（ZF=0 および SF=OF）場合転送する。
0F 4F */r*	CMOVNLE *r32, r/m32*	有効	有効	より小さくなく等しくない（ZF=0 および SF=OF）場合転送する。
REX.W + 0F 4F */r*	CMOVNLE *r64, r/m64*	有効	N.E.	より小さくなく等しくない（ZF=0 および SF=OF）場合転送する。
0F 41 */r*	CMOVNO *r16, r/m16*	有効	有効	オーバーフローがない（OF=0）場合転送する。
0F 41 */r*	CMOVNO *r32, r/m32*	有効	有効	オーバーフローがない（OF=0）場合転送する。
REX.W + 0F 41 */r*	CMOVNO *r64, r/m64*	有効	N.E.	オーバーフローがない（OF=0）場合転送する。
0F 4B */r*	CMOVNP *r16, r/m16*	有効	有効	パリティがない（PF=0）場合転送する。
0F 4B */r*	CMOVNP *r32, r/m32*	有効	有効	パリティがない（PF=0）場合転送する。
REX.W + 0F 4B */r*	CMOVNP *r64, r/m64*	有効	N.E.	パリティがない（PF=0）場合転送する。
0F 49 */r*	CMOVNS *r16, r/m16*	有効	有効	符号がない（SF=0）場合転送する。
0F 49 */r*	CMOVNS *r32, r/m32*	有効	有効	符号がない（SF=0）場合転送する。
REX.W + 0F 49 */r*	CMOVNS *r64, r/m64*	有効	N.E.	符号がない（SF=0）場合転送する。
0F 45 */r*	CMOVNZ *r16, r/m16*	有効	有効	ゼロでない（ZF=0）場合転送する。
0F 45 */r*	CMOVNZ *r32, r/m32*	有効	有効	ゼロでない（ZF=0）場合転送する。
REX.W + 0F 45 */r*	CMOVNZ *r64, r/m64*	有効	N.E.	ゼロでない（ZF=0）場合転送する。
0F 40 */r*	CMOVO *r16, r/m16*	有効	有効	オーバーフローがある（OF=0）場合転送する。
0F 40 */r*	CMOVO *r32, r/m32*	有効	有効	オーバーフローがある（OF=0）場合転送する。
REX.W + 0F 40 */r*	CMOVO *r64, r/m64*	有効	N.E.	オーバーフローがある（OF=0）場合転送する。

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
0F 4A */r*	CMOVP *r16, r/m16*	有効	有効	パリティがある（PF=1）場合転送する。
0F 4A */r*	CMOVP *r32, r/m32*	有効	有効	パリティがある（PF=1）場合転送する。
REX.W + 0F 4A */r*	CMOVP *r64, r/m64*	有効	N.E.	パリティがある（PF=1）場合転送する。
0F 4A */r*	CMOVPE *r16, r/m16*	有効	有効	パリティが偶数（PF=1）の場合転送する。
0F 4A */r*	CMOVPE *r32, r/m32*	有効	有効	パリティが偶数（PF=1）の場合転送する。
REX.W + 0F 4A */r*	CMOVPE *r64, r/m64*	有効	N.E.	パリティが偶数（PF=1）の場合転送する。
0F 4B */r*	CMOVPO *r16, r/m16*	有効	有効	パリティが奇数（PF=0）の場合転送する。
0F 4B */r*	CMOVPO *r32, r/m32*	有効	有効	パリティが奇数（PF=0）の場合転送する。
REX.W + 0F 4B */r*	CMOVPO *r64, r/m64*	有効	N.E.	パリティが奇数（PF=0）の場合転送する。
0F 48 */r*	CMOVS *r16, r/m16*	有効	有効	符号がある（SF=1）場合転送する。
0F 48 */r*	CMOVS *r32, r/m32*	有効	有効	符号がある（SF=1）場合転送する。
REX.W + 0F 48 */r*	CMOVS *r64, r/m64*	有効	N.E.	符号がある（SF=1）場合転送する。
0F 44 */r*	CMOVZ *r16, r/m16*	有効	有効	ゼロ（ZF=1）の場合転送する。
0F 44 */r*	CMOVZ *r32, r/m32*	有効	有効	ゼロ（ZF=1）の場合転送する。
REX.W + 0F 44 */r*	CMOVZ *r64, r/m64*	有効	N.E.	ゼロ（ZF=1）の場合転送する。

IA-32e モードでの操作

```
temp <-- DEST
IF condition TRUE
  THEN
    DEST <-- SRC
  ELSIF (osize = 32)
    DEST <-- temp AND 0x00000000FFFFFFFF
    FI;
  ELSE
    DEST <-- temp
    FI;
  FI;
```

影響を受けるフラグ

なし。

IA-32e モードでの操作

命令が 64 ビットに拡張される。
デフォルトの操作サイズは 32 ビットである。
新しいレジスタ R8 ～ R15 へのアクセスが可能である。

保護モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタの内容が NULL セグメント・セレクタの場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

CMP - Compare Two Operands

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
3C *ib*	CMP AL, *imm8*	有効	有効	*imm8* を AL と比較する。
3D *iw*	CMP AX, *imm16*	有効	有効	*imm16* を AX と比較する。
3D *id*	CMP EAX, *imm32*	有効	有効	*imm32* を EAX と比較する。
REX.W + 3D *id*	CMP RAX, *imm32*	有効	N.E.	64 ビットに符号拡張された *imm32* を RAX と比較する。
80 /7 *ib*	CMP *r/m8, imm8*	有効	有効	*imm8* を *r/m8* と比較する。
REX + 80 /7 *ib*	CMP *r/m8*, imm8*	有効	N.E.	*imm8* を *r/m8* と比較する。
81 /7 *iw*	CMP *r/m16, imm16*	有効	有効	*imm16* を *r/m16* と比較する。
81 /7 *id*	CMP *r/m32,imm32*	有効	有効	*imm32* を *r/m32* と比較する。
REX.W + 81 /7 *id*	CMP *r/m64,imm32*	有効	N.E.	64 ビットに符号拡張された *imm32* を *r/m64* と比較する。
83 /7 *ib*	CMP *r/m16,imm8*	有効	有効	*imm8* を *r/m16* と比較する。
83 /7 *ib*	CMP *r/m32,imm8*	有効	有効	*imm8* を *r/m32* と比較する。
REX.W + 83 /7 *ib*	CMP *r/m64,imm8*	有効	N.E.	*imm8* を *r/m64* と比較する。
38 /*r*	CMP *r/m8,r8*	有効	有効	*r8* を *r/m8* と比較する。
REX + 38 /*r*	CMP *r/m8*,r8**	有効	N.E.	*r8* を *r/m8* と比較する。
39 /*r*	CMP *r/m16,r16*	有効	有効	*r16* を *r/m16* と比較する。
39 /*r*	CMP *r/m32,r32*	有効	有効	*r32* を *r/m32* と比較する。
REX.W + 39 /*r*	CMP *r/m64,r64*	有効	N.E.	*r64* を *r/m64* と比較する。
3A /*r*	CMP *r8,r/m8*	有効	有効	*r/m8* を *r8* と比較する。
REX + 3A /*r*	CMP *r8*,r/m8**	有効	N.E.	*r/m8* を *r8* と比較する。
3B /*r*	CMP *r16,r/m16*	有効	有効	*r/m16* を *r16* と比較する。
3B /*r*	CMP *r32,r/m32*	有効	有効	*r/m32* を *r32* と比較する。
REX.W + 3B /*r*	CMP *r64,r/m64*	有効	N.E.	*r/m64* を *r64* と比較する。

* 64 ビットモードでは、REX プリフィックスを使用する場合、バイトレジスタ AH、BH、CH、DH にアクセスするように r/m8 をコード化することはできない。1.4.2.2. 節も参照のこと。

影響を受けるフラグ

CF、OF、SF、ZF、AF、PF フラグが結果に従ってセットされる。

IA-32e モードでの操作

命令が 64 ビットに拡張される。
デフォルトの操作サイズは 32 ビットである。
新しいレジスタ R8 ～ R15 へのアクセスが可能である。

保護モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタの内容が NULL セグメント・セレクタの場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

CMPPD - Compare Packed Double-Precision Floating-Point Values

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
66 0F C2 /r ib	CMPPD *xmm1*, *xmm2/m128*, *imm8*	有効	有効	imm8 を比較プレディケートとして使用して、*xmm2/m128* のパックド倍精度浮動小数点値と *xmm1* のパックド倍精度浮動小数点値を比較する。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。

SIMD 浮動小数点例外

無効（SNaN オペランドの場合）、無効（QNaN と、『IA-32 インテル® アーキテクチャ・ソフトウェア・デベロッパーズ・マニュアル、中巻 A』の表 3-6. のプレディケートの場合）、デノーマル。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

実アドレスモード例外

#GP(0)	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

CMPPS - Compare Packed Single-Precision Floating-Point Values

オペコード	命令	64 ビット モード	互換 / レガシー モード	説明
0F C2 /r ib	CMPPS *xmm1, xmm2/m128, imm8*	有効	有効	*imm8* を比較プレディケートとして使用して、*xmm2/mem* のパックド単精度浮動小数点値と *xmm1* のパックド単精度浮動小数点値を比較する。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。

SIMD 浮動小数点例外

無効（SNaN オペランドの場合）、無効（QNaN と、『IA-32 インテル® アーキテクチャ・ソフトウェア・デベロッパーズ・マニュアル、中巻 A』の表 3-6. のプレディケートの場合）、デノーマル。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。

実アドレスモード例外

#GP(0)	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。

CMPS/CMPSB/CMPSW/CMPSD/CMPSQ - Compare String Operands

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
A6	CMPS m8, m8	有効	有効	レガシーモードではアドレス ES:(E)SI のバイトを比較し、64 ビットモードではアドレス (R)SI のバイトを比較する。その結果に従ってステータス・フラグをセットする。
A7	CMPS m16, m16	有効	有効	レガシーモードではアドレス ES:(E)SI のワードを比較し、64 ビットモードではアドレス (R)SI のワードを比較する。その結果に従ってステータス・フラグをセットする。
A7	CMPS m32, m32	有効	有効	レガシーモードではアドレス ES:(E)SI のダブルワードを比較し、64 ビットモードではアドレス (R)SI のダブルワードを比較する。その結果に従ってステータス・フラグをセットする。
REX.W + A7	CMPS m64, m64	有効	N.E.	アドレス RSI のクワッドワードをアドレス RDI のクワッドワードと比較し、結果に従ってステータス・フラグをセットする。
A6	CMPSB	有効	有効	レガシーモードではアドレス ES:(E)SI のバイトを比較し、64 ビットモードではアドレス (R)SI のバイトを比較する。その結果に従ってステータス・フラグをセットする。
A7	CMPSW	有効	有効	レガシーモードではアドレス ES:(E)SI のワードを比較し、64 ビットモードではアドレス (R)SI のワードを比較する。その結果に従ってステータス・フラグをセットする。
A7	CMPSD	有効	有効	レガシーモードではアドレス ES:(E)SI のダブルワードを比較し、64 ビットモードではアドレス (R)SI のダブルワードを比較する。その結果に従ってステータス・フラグをセットする。
REX.W + A7	CMPSD	有効	N.E.	アドレス RSI のクワッドワードをアドレス RDI のクワッドワードと比較し、結果に従ってステータス・フラグをセットする。

影響を受けるフラグ

CF、OF、SF、ZF、AF、PF フラグが比較の一時的結果に従ってセットされる。

IA-32e モードでの操作

命令が 64 ビットに拡張される。
デフォルトの操作サイズは 32 ビットである。

保護モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタの内容が NULL セグメント・セレクタの場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

CMPSD - Compare Scalar Double-Precision Floating-Point Values

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
F2 0F C2 /r ib	CMPSD *xmm1*, *xmm2/m64*, *imm8*	有効	有効	*imm8* を比較プレディケートとして使用して、*xmm2/m64* の下位の倍精度浮動小数点値と *xmm1* の下位の倍精度浮動小数点値を比較する。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。

SIMD 浮動小数点例外

無効（SNaN オペランドの場合）、無効（QNaN と、『IA-32 インテル® アーキテクチャ・ソフトウェア・デベロッパーズ・マニュアル、中巻 A』の表 3-6. のプレディケートの場合）、デノーマル。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

CMPSS - Compare Scalar Single-Precision Floating-Point Values

オペコード	命令	64 ビット モード	互換 / レガシー モード	説明
F3 0F C2 /r ib	CMPSS *xmm1, xmm2/m32, imm8*	有効	有効	*imm8* を比較プレディケートとして使用して、*xmm2/m32* の最下位の単精度浮動小数点値と *xmm1* の最下位の単精度浮動小数点値を比較する。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。

SIMD 浮動小数点例外

無効（SNaN オペランドの場合）、無効（QNaN と、『IA-32 インテル® アーキテクチャ・ソフトウェア・デベロッパーズ・マニュアル、中巻 A』の表 3-6. のプレディケートの場合）、デノーマル。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

CMPXCHG - Compare and Exchange

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
0F B0/*r*	CMPXCHG *r/m8,r8*	有効	有効	AL を *r/m8* と比較し、等しい場合は ZF をセットし、*r8* を *r/m8* にロードする。等しくない場合は ZF をクリアし、*r/m8* を AL にロードする。
REX + 0F B0/*r*	CMPXCHG *r/m8*,r8**	有効	N.E.	AL を *r/m8* と比較し、等しい場合は ZF をセットし、*r8* を *r/m8* にロードする。等しくない場合は ZF をクリアし、*r/m8* を AL にロードする。
0F B1/*r*	CMPXCHG *r/m16,r16*	有効	有効	AX を *r/m16* と比較し、等しい場合は ZF をセットし、*r16* を *r/m16* にロードする。等しくない場合は ZF をクリアし、*r/m16* を AX にロードする。
0F B1/*r*	CMPXCHG *r/m32,r32*	有効	有効	EAX を *r/m32* と比較し、等しい場合は ZF をセットし、*r32* を *r/m32* にロードする。等しくない場合は ZF をクリアし、*r/m32* を EAX にロードする。
REX.W + 0F B1/*r*	CMPXCHG *r/m64,r64*	有効	N.E.	RAX を *r/m64* と比較し、等しい場合は ZF をセットし、*r64* を *r/m64* にロードする。等しくない場合は ZF をクリアし、*r/m64* を AL にロードする。

* 64 ビットモードでは、REX プリフィックスを使用する場合、バイトレジスタ AH、BH、CH、DH にアクセスするように r/m8 をコード化することはできない。1.4.2.2. 節も参照のこと。

影響を受けるフラグ

デスティネーション・オペランドと AL、AX、または EAX レジスタの値が等しい場合は ZF フラグがセットされ、等しくない場合はクリアされる。CF、PF、AF、SF、OF フラグは比較演算の結果に従ってセットされる。

IA-32e モードでの操作

64 ビットに拡張される。
デフォルトの操作サイズは 32 ビットである。

保護モード例外

#GP(0)	デスティネーションが書き込み不可能なセグメントにある場合。
	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタの内容が NULL セグメント・セレクタの場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

CMPXCHG8B/CMPXCHG16B - Compare and Exchange 8 Bytes

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
0F C7 /1 m64	CMPXCHG8B *m64*	有効	有効	EDX:EAX を *m64* と比較し、等しい場合は ZF をセットし、ECX:EBX を *m64* にロードする。等しくない場合は ZF をクリアし、*m64* を EDX:EAX にロードする。
REX.W + 0F C7 /1 m128	CMPXCHG16B *m128*	有効	N.E.	RDX:RAX を *m128* と比較し、等しい場合は ZF をセットし、RCX:RBX を *m128* にロードする。等しくない場合は ZF をクリアし、*m128* を RDX:RAX にロードする。

影響を受けるフラグ

デスティネーション・オペランドと EDX:EAX が等しい場合は ZF フラグがセットされ、等しくない場合はクリアされる。CF、PF、AF、SF、OF フラグは影響を受けない。

IA-32e モードでの操作

64 ビットに拡張される。
デフォルトの操作サイズは 64 ビットである。
CMPXCHG16B では、デスティネーション（メモリ）オペランドが 16 バイト・アライメントでなければならない。

保護モード例外

#UD	デスティネーション・オペランドがメモリ・ロケーションでない場合。
#GP(0)	デスティネーションが書き込み不可能なセグメントにある場合。
	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタの内容が NULL セグメント・セレクタの場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#UD	デスティネーション・オペランドがメモリ・ロケーションでない場合。
#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。

#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。

仮想 8086 モード例外

#UD	デスティネーション・オペランドがメモリ・ロケーションでない場合。
#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
	CMPXCHG16B のメモリ・オペランドのアライメントが 16 バイトに合っていない場合。
	CPUID 機能フラグ CMPXCHG16B が 0 の場合。
#UD	デスティネーション・オペランドがメモリ・ロケーションでない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

COMISD - Compare Scalar Ordered Double-Precision Floating-Point Values and Set EFLAGS

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
66 0F 2F /r	COMISD *xmm1*, *xmm2/m64*	有効	有効	*xmm1* と *xmm2/mem64* の下位の倍精度浮動小数点値を比較し、その結果に従って EFLAGS フラグをセットする。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。

SIMD 浮動小数点例外

無効（SNaN または QNaN オペランドの場合）、デノーマル。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

COMISS - Compare Scalar Ordered Single-Precision Floating-Point Values and Set EFLAGS

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
0F 2F /r	COMISS *xmm1*, *xmm2/m32*	有効	有効	*xmm1* と *xmm2/mem32* の最下位の単精度浮動小数点値を比較し、その結果に従って EFLAGS フラグをセットする。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。

SIMD 浮動小数点例外

無効（SNaN または QNaN オペランドの場合）、デノーマル。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

CPUID - CPU Identification

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
0F A2	CPUID	有効	有効	EAX レジスタに最初に入力された値に応じて、プロセッサの識別情報と機能情報を EAX、EBX、ECX、EDX の各レジスタに返す。

説明

EAX レジスタの入力値が 0 の場合、プロセッサは、CPUID 基本情報を返すために、CPUID 命令が認識可能な最大値を EAX レジスタに返す（表 2-5. を参照）。EBX、EDX、ECX の各レジスタには、ベンダ識別ストリングが返される。インテル® プロセッサの場合、ベンダ識別ストリングは「GenuineIntel」になり、次のように表される。

```
EBX ← 756e6547h (* "Genu", with G in the low nibble of BL *)
EDX ← 49656e69h (* "ineI", with i in the low nibble of DL *)
ECX ← 6c65746eh (* "ntel", with n in the low nibble of CL *)
```

表 2-5. CPUID 機能リーフ

EAX の初期値	プロセッサに関して提供される情報	
0H	EAX EBX ECX EDX	CPUID 基本情報の最大入力値（表 2-6. を参照） “Genu” “ntel” “inel”
1H	EAX EBX ECX EDX	バージョン情報（タイプ、ファミリ、モデル、ステッピング ID） ビット 7 ～ 0 : ブランド・インデックス ビット 15 ～ 8 : CLFLUSH のラインサイズ（返される値 * 8 = キャッシュ・ライン・サイズ） ビット 23 ～ 16 : 物理パッケージ当たりの論理プロセッサの数 ビット 31 ～ 24 : プロセッサのローカル APIC ID の初期値 機能情報（図 2-8. を参照） 機能情報（図 2-8. を参照）
2H	EAX EBX ECX EDX	キャッシュおよび TLB 情報 キャッシュおよび TLB 情報 キャッシュおよび TLB 情報 キャッシュおよび TLB 情報
3H	EAX EBX ECX EDX	予約済み 予約済み 予約済み 予約済み

表 2-5. CPUID 機能リーフ（続き）

EAX の初期値	プロセッサに関して提供される情報
4H	キャッシュ・パラメータ・リーフ EAX ビット 4 ～ 0：キャッシュ・タイプ ** ビット 7 ～ 5：キャッシュ・レベル（1 から始まる） ビット 8：キャッシュ・レベルの自己初期化（ソフトウェアによる初期化は不要） ビット 9：フル・アソシアティブ・キャッシュ ビット 13 ～ 10：予約済み ビット 25 ～ 14：このキャッシュを共有しているスレッドの数 * EBX ビット 31 ～ 26：このダイ上のプロセッサ・コアの数（マルチコア）* ビット 11 ～ 00：L = システム・コヒーレンシ・キャッシュ・サイズ * ビット 21 ～ 12：P = 物理ライン・パーティション * ECX ビット 31 ～ 22：W = アソシアティブ・ウェイ数 * EDX ビット 31 ～ 00：S = セット数 * 予約済み = 0 * この数を求めるには、レジスタファイル内の値に 1 を加算する。 例えば、プロセッサ・コアの数は、EAX[31:26]+1 になる。 ** キャッシュ・タイプ・フィールド 0 = Null - キャッシュなし 1 = データ・キャッシュ 2 = 命令キャッシュ 3 = ユニファイド・キャッシュ 4 ～ 31 = 予約済み 注：3 より大きく 80000000 より小さい CPUID リーフは、IA32_CR_MISC_ENABLES.BOOT_NT4（ビット 22）がクリアされている場合（デフォルト）にのみ参照可能である。
5H	MONITOR/MWAIT リーフ EAX ビット 15 ～ 00：最小モニタライン・サイズ（バイト単位）（デフォルトはプロセッサのモニタ・グラニュラリティ） ビット 31 ～ 16：予約済み = 0 EBX ビット 15 ～ 00：最大モニタライン・サイズ（バイト単位）（デフォルトはプロセッサのモニタ・グラニュラリティ） ECX ビット 31 ～ 16：予約済み = 0 EDX 予約済み = 0 予約済み = 0
	拡張機能 CPUID 情報
80000000H	EAX 拡張機能 CPUID 情報の最大入力値（表 2-6. を参照） EBX 予約済み ECX 予約済み EDX 予約済み
80000001H	拡張されたプロセッサ・シグネチャと拡張された機能ビット EAX 予約済み EBX 予約済み ECX 予約済み EDX ビット 10 ～ 0：予約済み ビット 11：SYSCALL/SYSRET を利用可能 ビット 19 ～ 12：予約済み ビット 28 ～ -21：予約済み ビット 29：インテル EM64T を利用可能 ビット 31 ～ -30：予約済み

表 2-5. CPUID 機能リーフ（続き）

EAX の初期値	プロセッサに関して提供される情報
80000002H	EAX プロセッサ・ブランド・ストリング EBX プロセッサ・ブランド・ストリング（続き） ECX プロセッサ・ブランド・ストリング（続き） EDX プロセッサ・ブランド・ストリング（続き）
80000003H	EAX プロセッサ・ブランド・ストリング（続き） EBX プロセッサ・ブランド・ストリング（続き） ECX プロセッサ・ブランド・ストリング（続き） EDX プロセッサ・ブランド・ストリング（続き）
80000004H	EAX プロセッサ・ブランド・ストリング（続き） EBX プロセッサ・ブランド・ストリング（続き） ECX プロセッサ・ブランド・ストリング（続き） EDX プロセッサ・ブランド・ストリング（続き）
80000005H	EAX 予約済み = 0 EBX 予約済み = 0 ECX 予約済み = 0 EDX 予約済み = 0
80000006H	EAX 予約済み = 0 EBX 予約済み = 0 ECX ビット 7 ～ 0 : キャッシュ・ライン・サイズ ビット 15 ～ 12 : L2 アソシアティビティ ビット 31 ～ 16 : キャッシュ・サイズ（1K 単位） 初期値は ECX = 0x04008040 EDX 予約済み = 0
80000007H	EAX 予約済み = 0 EBX 予約済み = 0 ECX 予約済み = 0 EDX 予約済み = 0
80000008H	EAX 仮想 / 物理アドレスサイズ ビット 7 ～ 0 : 物理アドレスビット数 ビット 15 ～ 8 : 仮想アドレスビット数 ビット 31 ～ 16 : 予約済み 初期値は EAX = 0x3028 EBX 予約済み = 0 ECX 予約済み = 0 EDX 予約済み = 0

表 2-6. インテル® EM64T 対応プロセッサに対する CPUID 命令のソース・オペランドの最大値

IA-32 プロセッサ	EAX レジスタの最大値	
	基本情報	拡張機能情報
	5H	80000008H

入力値が 1 の場合、プロセッサは EAX レジスタにバージョン情報を返す（図 2-3. を参照）。バージョン情報は、IA-32 プロセッサ・ファミリ ID、モデル ID、ステッピング ID、プロセッサ・タイプで構成される。インテル® EM64T に対応した最初のプロセッサのモデル、ファミリ、プロセッサ・タイプは、次のようになる。

- モデル - 0011B
- ファミリ - 1111B

- プロセッサ・タイプ - 00B

使用可能なプロセッサ・タイプについては、表2-7.を参照のこと。ステッピングIDについては、必要に応じて情報を提供する。

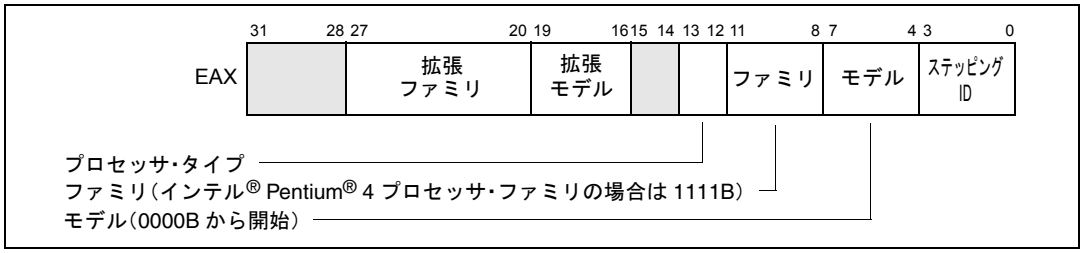

図 2-3. EAX レジスタのバージョン情報

表 2-7. プロセッサ・タイプ・フィールド

タイプ	コード化
オリジナル OEM プロセッサ	00B
インテル® OverDrive® プロセッサ	01B
デュアルプロセッサ *	10B
インテル用に予約済み	11B

* Intel486™ プロセッサには適用不可

ファミリIDが0FHまたはそれ以上の場合にのみ、拡張ファミリIDと拡張モデルIDを確認する必要がある。プロセッサ情報は、常にファミリ、モデル、ステッピングを組み合わせて表示すること。

各IDフィールドを次のように計算して、ファミリを表示する。

```
Displayed family = ((Extended Family ID(4-bits) << 4)) (8-bits)
+ Family ID (4-bits zero extended to 8-bits)
```

表示されるモデルは、モデルIDと拡張モデルIDから次のように計算する。

```
Displayed Model = ((Extended Model ID (4-bits) << 4))(8-bits)
+ Model (4-bits zero extended to 8-bits)
```

初期のIA-32プロセッサの識別については、AP-485『インテル® プロセッサの識別とCPUID命令（資料番号241618J）』および『IA-32インテル® アーキテクチャ・ソフトウェア・デベロッパーズ・マニュアル、上巻』の第13章を参照のこと。

EAXレジスタの入力値が1の場合、EBXレジスタには、関連のない4つの情報が返される。

- ブランド・インデックス EBX[7:0] - IA-32 プロセッサのブランド・ストリングで構成されるブランド・ストリング・テーブルのエントリには、この数字が入っている。ブランド・インデックスの用途の詳細については、この命令の「ブランド識別」の項を参照のこと。このフィールドは、インテル® Pentium® III Xeon™ プロセッサで導入されたものである。
- CLFLUSH 命令キャッシュ・ライン・サイズ EBX[15:8] - この数字は、CLFLUSH 命令によって 8 バイト・インクリメントでフラッシュされるキャッシュ・ラインのサイズを示す。このフィールドは、インテル® Pentium® 4 プロセッサで導入されたものである。
- EBX[23:16] には、パッケージに含まれるスレッドの数が示される。
- APIC ID の初期値 EBX[31:24] - この数字は、電源入力時にプロセッサのローカル APIC に割り当てられる 8 ビットの物理 ID を示す。このフィールドは、インテル Pentium 4 プロセッサで導入されたものである。

EAX レジスタの入力値が 1 の場合、機能情報は EDX レジスタに返される（図 2-8. を参照）。オペレーティング・システムやアプリケーションのコードは、この機能ビットを利用して、プロセッサで利用可能な IA-32 アーキテクチャ機能を識別できる。表 2-8. に、EDX レジスタの機能フラグのコード化を示す。現時点で EDX レジスタに返されるすべての機能フラグについて、1 は、対応する機能をサポートしていることを示す。ソフトウェアは、機能フラグを正しく解釈するのに当たってインテルをベンダとして識別する必要がある。ソフトウェアは、将来の機能フラグについて、機能の存在を示す 1 に依存してはならない。

表 2-8. CPUID 機能情報

レジスタビット番号	ニーモニック	説明
EDX.0	FPU	**オンチップ浮動小数点ユニット**。プロセッサは、x87 FPU を搭載している。
EDX.1	VME	**仮想 8086 モード強化**。仮想 8086 モード強化機能には次のものが含まれる。機能制御用の CR4.VME、保護モード仮想割り込み用の CR4.PVI、ソフトウェア割り込みインダイレクション、ソフトウェア・インダイレクション・ビットマップによる TSS の拡張、EFLAGS.VIF フラグ、EFLAGS.VIP フラグ。
EDX.2	DE	**デバッグ拡張**。機能制御用の CR4.DE、オプションの DR4 および DR5 へのアクセストラップを含めて、I/O ブレークポイントをサポート。
EDX.3	PSE	**ページ・サイズ拡張**。機能制御用の CR4.PSE、ページ・ディレクトリ・エントリ（PDE）内の定義済みのダーティビット、CR3、PDE、PTE 内のオプションの予約ビット・トラッピングを含めて、4M バイトのラージ・ページ・サイズをサポート。
EDX.4	TSC	**タイム・スタンプ・カウンタ**。特権制御用の CR4.TSD を含めて、RDTSC 命令をサポート。
EDX.5	MSR	**モデル固有レジスタの RDMSR 命令および WRMSR 命令**。RDMSR 命令と WRMSR 命令をサポート。MSR によっては、プロセッサに依存しないものもある。

表 2-8. CPUID 機能情報（続き）

レジスタビット番号	ニーモニック	説明
EDX.6	PAE	**物理アドレス拡張**。32 ビットを超える物理アドレスをサポート。拡張ページ・テーブル・エントリ・フォーマット、ページ変換テーブル内の特別レベルが定義され、PAE ビットが 1 の場合は、4M バイトページの代わりに 2M バイトページをサポート。32 ビットを超える場合の実際のアドレスビット数は定義されておらず、プロセッサ固有である。
EDX.7	MCE	**マシンチェック例外**。機能制御用の CR4.MCE を含めて、マシンチェック用に例外 18 が定義される。この機能は、モデル固有インプリメンテーションにおけるマシン・チェック・エラーのロギング、レポート、プロセッサ・シャットダウンについては定義しない。マシンチェック例外ハンドラは、プロセッサのバージョンに従ってモデル固有の例外処理を実行するか、またはマシンチェック機能の有無を確認する必要が生じる場合がある。
EDX.8	CX8	**CMPXCHG8B 命令**。8 バイト（64 ビット）比較交換命令をサポート（暗黙的にロックされ、アトミックに行われる）。
EDX.9	APIC	**オンチップ APIC**。プロセッサは、アドバンスト・プログラム可能割り込みコントローラ（APIC）を内蔵し、物理アドレス範囲 FFFE0000H ～ FFFE0FFFH でメモリ・マップ・コマンドに応答する（デフォルトでは、プロセッサによっては APIC の再配置を許可するものもある）。
EDX.10	Reserved	予約済み。
EDX.11	SEP	**SYSENTER 命令および SYSEXIT 命令**。SYSENTER、SYSEXIT、関連する MSR をサポート。
EDX.12	MTRR	**メモリタイプ範囲レジスタ**。MTRR をサポート。MTRRcap MSR には機能ビットが含まれていて、サポートされているメモリタイプ、サポートされている可変 MTRR の個数、固定 MTRR のサポートの有無について示される。
EDX.13	PGE	**PTE グローバル・ビット**。ページ・ディレクトリ・エントリ（PDE）およびページ・テーブル・エントリ（PTE）内のグローバル・ビットをサポート。これらは、各種の処理に共通で、フラッシュする必要のない TLB エントリであることを示す。この機能は、CR4.PGE ビットによって制御される。
EDX.14	MCA	**マシン・チェック・アーキテクチャ**。インテル® Pentium® 4 プロセッサ、P6 ファミリ・プロセッサ、将来のプロセッサのエラーレポート用の互換メカニズムを提供するマシン・チェック・アーキテクチャをサポート。MCG_CAP MSR には、サポートされているエラーレポート用 MSR バンクの個数を示す機能ビットが含まれる。
EDX.15	CMOV	**条件付き転送命令**。条件付き転送命令（CMOV）をサポート。また、x87 FPU が搭載されている場合は（CPUID.FPU 機能ビットによって示される）、FCOMI 命令および FCMOV 命令をサポート。
EDX.16	PAT	**ページ属性テーブル**。ページ属性テーブルをサポート。この機能は、メモリタイプ範囲レジスタ（MTRR）の数を増やす。これによって、オペレーティング・システムは、リニアアドレスを使用して 4K バイト単位でメモリの属性を指定できる。

表 2-8. CPUID 機能情報（続き）

レジスタビット番号	ニーモニック	説明
EDX.17	PSE-36	**36 ビット・ページ・サイズ拡張**。拡張 4M バイトページをサポート。これにより、4G バイトを超える物理メモリのアドレス指定が可能になる。この機能は、4M バイトページの物理アドレスの上位 4 ビットが、ページ・ディレクトリ・エントリのビット 13 ～ 16 によってエンコーディングされることを示す。
EDX.18	PSN	**プロセッサ・シリアル番号**。96 ビットのプロセッサ識別番号機能をサポートしており、この機能がイネーブルになっている（インテル® Pentium® III プロセッサのみがこの機能をサポート）。
EDX.19	CLFSH	**CLFLUSH 命令**。CLFLUSH 命令をサポート。
EDX.20	Reserved	予約済み。
EDX.21	DS	**デバッグストア**。メモリ常駐バッファにデバッグ情報を書き込む機能をサポート。この機能は、分岐トレースストア（BTS）およびプリサイス・イベント・ベース・サンプリング（PEBS）機能によって使用される（詳しくは、『IA-32 インテル® アーキテクチャ・ソフトウェア・デベロッパーズ・マニュアル、下巻』の第 15 章「デバッグと性能モニタリング」を参照のこと）。
EDX.22	ACPI	**温度モニタおよびソフトウェア制御クロック機能**。プロセッサは MSR を内部に実装している。MSR を使用すると、ソフトウェアによる制御の下でプロセッサ温度を監視できるとともに、事前に定義したデューティ・サイクルでプロセッサのパフォーマンスを調整できる。
EDX.23	MMX	**インテル® MMX® テクノロジ**。インテル® MMX® テクノロジをサポート。
EDX.24	FXSR	**FXSAVE 命令および FXRSTOR 命令**。浮動小数点コンテキストの高速な保存 / 復元用の FXSAVE 命令および FXRSTOR 命令をサポート。また、このビットがセットされている場合、オペレーティング・システムは、CR4.OSFXSR を利用して、FXSAVE 命令および FXRSTOR 命令のサポートを示すことができる。
EDX.25	SSE	**SSE**。SSE 拡張命令をサポート。
EDX.26	SSE2	**SSE2**。SSE2 拡張命令をサポート。
EDX.27	SS	**セルフスヌープ**。競合するメモリタイプの管理機能をサポートしており、プロセッサのキャッシュ構造のスヌープを実行して、バスに対して発行されたトランザクションを検出できる。
EDX.28	HT	**ハイパー・スレッディング・テクノロジ**。プロセッサはハイパー・スレッディング・テクノロジをサポートする。
EDX.29	TM	**温度モニタ**。プロセッサは、温度モニタ自動温度制御回路（TCC）を実装している。
EDX.30	Reserved	予約済み。
EDX.31	FERR	FERR# 信号の変更。
ECX.0	SSE3	SSE3 を利用可能。
ECX.1	Reserved	予約済み。
ECX.2	Reserved	予約済み。

表 2-8. CPUID 機能情報（続き）

レジスタビット番号	ニーモニック	説明
ECX.3	Monitor	Monitor/Mwait 命令。
ECX.4	DS_CPL	CPL の条件を満たすデバッグストアを利用可能。
ECX.5	Reserved	予約済み。
ECX.6	Reserved	予約済み。
ECX.7	EST	拡張版 Intel SpeedStep® テクノロジ。
ECX.8	TM2	温度モニタ 2 を利用可能。
ECX.9	Reserved	予約済み。
ECX.10	CNXT-ID	L1 コンテキスト ID を利用可能。
ECX.11	Reserved	予約済み。
ECX.12	Reserved	予約済み。
ECX.13	CMPXCHG16B	CMPXCHG16B を利用可能。
ECX.31:13	Reserved	予約済み。

入力値が 2 の場合、プロセッサはプロセッサの内部キャッシュおよびTLB に関する情報を EAX、EBX、ECX、EDX レジスタに返す。これらのレジスタのコード化は、以下のとおりである。

- EAX レジスタ（AL レジスタ）の最下位バイトは、入力値を 2 として、プロセッサのキャッシュと TLB の完全な記述を得るために CPUID 命令が実行されなければならない回数を示す。インテル Pentium 4 プロセッサ・ファミリの最初のメンバは 1 を返す。
- 各レジスタの最上位ビット（ビット 31）は、レジスタに有効な情報がある（0 にクリア）か、予約されている（1 にセット）かを示す。
- レジスタに有効な情報がある場合は、その情報は 1 バイトの記述子に収められる。表 2-9. に、それらの記述子のコード化を示す。EAX、EBX、ECX、EDX の各レジスタ内の記述子の順序は定義されていない。すなわち、特定のバイトが特定のキャッシュ・タイプまたは TLB タイプの記述子が入るようには指定されていない。記述子は、あらゆる順序で示されることがある。

表 2-9. キャッシュ・パラメータ・テーブル

記述子の値	説明
22h	512K バイト第 3 レベルのキャッシュ、4 ウェイ、64 バイト・ライン・サイズ、128 バイト・セクタ・サイズ
23h	1M バイト第 3 レベルのキャッシュ、8 ウェイ、64 バイト・ライン・サイズ、128 バイト・セクタ・サイズ
25h	2M バイト第 3 レベルのキャッシュ、8 ウェイ、64 バイト・ライン・サイズ、128 バイト・セクタ・サイズ
29h	4M バイト第 3 レベルのキャッシュ、8 ウェイ、64 バイト・ライン・サイズ、128 バイト・セクタ・サイズ
40h	第 3 レベルのキャッシュなし

表 2-9. キャッシュ・パラメータ・テーブル（続き）

記述子の値	説明
50h	4K バイトページおよび 2M/4M バイトページ用の 64 エントリ（ITLB）
51h	4K バイトページおよび 2M/4M バイトページ用の 128 エントリ（ITLB）
52h	4K バイトページおよび 2M/4M バイトページ用の 256 エントリ（ITLB）
5Bh	4K バイトページおよび 4M バイトページ用の 64 エントリ（DTLB）
5Ch	4K バイトページおよび 4M バイトページ用の 128 エントリ（DTLB）
5Dh	4K バイトページおよび 4M バイトページ用の 256 エントリ（DTLB）
60h	16K バイト第 1 レベルのデータ・キャッシュ、8 ウェイ・セット・アソシアティブ、64 バイト・ライン・サイズ
66h	8K バイト第 1 レベルのデータ・キャッシュ、4 ウェイ・セット・アソシアティブ、64 バイト・ライン・サイズ
67h	16K バイト第 1 レベルのデータ・キャッシュ、4 ウェイ・セット・アソシアティブ、64 バイト・ライン・サイズ
68h	32K バイト第 1 レベルのデータ・キャッシュ、4 ウェイ・セット・アソシアティブ、64 バイト・ライン・サイズ
70h	12K uops、トレース・キャッシュ、8 ウェイ・セット・アソシアティブ
71h	16K uops、トレース・キャッシュ、8 ウェイ・セット・アソシアティブ
72h	32K uops、トレース・キャッシュ、8 ウェイ・セット・アソシアティブ
78H	1M バイト第 2 レベルのキャッシュ、8 ウェイ・セット・アソシアティブ、64 バイト・ライン・サイズ、128 バイト・セクタ・サイズ
79h	128K バイト第 2 レベルのキャッシュ、8 ウェイ・セット・アソシアティブ、64 バイト・ライン・サイズ、128 バイト・セクタ・サイズ
7Ah	256K バイト第 2 レベルのキャッシュ、8 ウェイ・セット・アソシアティブ、64 バイト・ライン・サイズ、128 バイト・セクタ・サイズ
7Bh	512K バイト第 2 レベルのキャッシュ、8 ウェイ・セット・アソシアティブ、64 バイト・ライン・サイズ、128 バイト・セクタ・サイズ
7Ch	1M バイト第 2 レベルのキャッシュ、8 ウェイ・セット・アソシアティブ、64 バイト・ライン・サイズ、128 バイト・セクタ・サイズ

インテル Pentium 4 プロセッサ・ファミリの最初のメンバは、CPUID 命令が入力値 2 で実行されると、キャッシュと TLB に関する以下の情報を返す。

EAX	66 5B 50 01H
EBX	0H
ECX	0H
EDX	00 7A 70 40H

これらの値は、以下のように解釈される。

- EAX レジスタの最下位バイト（バイト 0）が 01H に設定されており、プロセッサのキャッシュと TLB に関する完全な情報を取り出すのに、CPUID 命令は入力値を 2 として 1 回実行すればよいことを示している。
- 4 つのレジスタすべて（EAX、EBX、ECX、EDX）の最上位ビットが 0 にセットされており、各レジスタが有効な 1 バイトの記述子をもつことを示している。
- EAX レジスタのバイト 1、2、3 は、プロセッサが以下のものを備えていることを示す。
  - 50H - 4 K バイト、2M バイト、4M バイト・ページ・マップ用の 64 エントリ命令 TLB
  - 5BH - 4K バイトおよび 4M バイト・ページ・マップ用の 64 エントリデータ TLB
  - 66H - 8K バイト第 1 レベルのデータ・キャッシュ、4 ウェイ・セット・アソシアティブ、64 バイト・キャッシュ・ライン・サイズ
- EBX および ECX レジスタ内の記述子は有効であるが、内容は NULL 記述子である。
- EDX レジスタのバイト 0、1、2、3 は、プロセッサが以下のものを備えていることを示す。
  - 00H - NULL 記述子
  - 70H - 12K バイト第 1 レベルのコード・キャッシュ、4 ウェイ・セット・アソシアティブ、64 バイト・キャッシュ・ライン・サイズ
  - 7AH - 256K バイト第 2 レベルのキャッシュ、8 ウェイ・セット・アソシアティブ、128 バイト・キャッシュ・ライン・サイズ
  - 00H - NULL 記述子

ブランド識別

CPUID 命令による IA-32 プロセッサのブランド識別を容易にするため、ブランド・インデックスとブランド・ストリングという 2 つの機能が用意されている。

インテル Pentium III Xeon プロセッサで CPUID 命令に追加されたブランド・インデックスは、インテル Pentium 4 プロセッサをはじめ、将来のすべての IA-32 プロセッサに搭載される。ブランド・インデックスは、ブランド識別テーブルへのエントリポイントとなる。ブランド識別テーブルはシステム・ソフトウェアによってメモリ内に保持され、システムレベルおよびユーザレベルのコードからアクセスが可能である。このブランド識別テーブルでは、各ブランド・インデックスが、公認のインテル® プロセッサ・ファミリであることを示す ASCII ブランド識別ストリングとプロセッサのモデル番号とに関連付けられている（例えば、「インテル Pentium III プロセッサ」など）。

EAX レジスタに値 1 を設定して CPUID 命令を実行すると、EBX レジスタの下位バイトにブランド・インデックスが返される。ソフトウェアでこのインデックスを使用することで、ブランド識別テーブル内にあるプロセッサのブランド識別ストリングを特定することができる。ブランド識別テーブルの最初のエントリ（ブランド・インデックス 0）は予約されており、ブランド識別機能をサポートしていないプロセッサとの下方互換性が保たれている。表 2-10. は、現時点でプロセッサ・ブランド識別ストリングが関連付けられているブランド・インデックスを示している。

次の 2 点を行うことを推奨する。(1) ブランド識別テーブル内にあるすべての予約済みエントリを、インデックスが将来のインテル・プロセッサ用に予約されていることを示すブランド・ストリングに関連付ける、(2) 予約済みのブランド・インデックスを適切に処理できるように、ソフトウェアを対応させる。

表 2-10. ブランド・ストリングのオフセット

ブランド・インデックス	ブランド・ストリング
0	このプロセッサはブランド ID 機能をサポートしていない。
1	インテル® Celeron® プロセッサ *
2	インテル® Pentium® III プロセッサ *
3	インテル® Pentium® III Xeon™ プロセッサ
4	インテル® Pentium® III プロセッサ
6	モバイル インテル® Pentium® III プロセッサ - M
7	モバイル インテル® Celeron® プロセッサ
8	インテル® Pentium® 4 プロセッサ プロセッサ・シグニチャ ≥ 00000F13H の場合、インテル® プロセッサ
9	インテル® Pentium® 4 プロセッサ
10	インテル® Celeron® プロセッサ
11	インテル® Xeon™ プロセッサ プロセッサ・シグニチャ < 00000F13H の場合、インテル® Xeon™ プロセッサ MP
12	インテル® Xeon™ プロセッサ MP
14	モバイル インテル® Pentium® 4 プロセッサ - M プロセッサ・シグニチャ < 00000F13H の場合、インテル® Xeon™ プロセッサ
15	モバイル インテル® Celeron® プロセッサ
16 ～ 255	将来のプロセッサ用に予約済み

* インテル® Pentium® III Xeon™ プロセッサ以降に発表されたプロセッサのバージョンを指す。

ブランド・ストリング機能は、インテル Pentium 4 プロセッサで導入された CPUID 命令の拡張機能である。この機能を利用すると、CPUID 命令は、ASCII ブランド識別ストリングとプロセッサの最大動作周波数を、EAX、EBX、ECX、EDX の各レジスタに返す。ただし、返される周波数は、プロセッサの現行の動作周波数ではなく、プロセッサに対して定義されている最大動作周波数である。

ブランド・ストリング機能を使用するには、CPUID 命令を 3 回実行する必要がある（EAX レジスタの入力値を 1 回目は 80000002H、2 回目は 80000003H、3 回目は 80000004H に設定）。

ブランド・ストリングの長さはアーキテクチャ上、48 バイトに定義されている。最初の 47 バイトには ASCII 文字が格納され、48 番目のバイトは NULL に定義されている。実装を容易にするため、このストリングは右揃え（先頭に空白を挿入）になっている。各入力値（EAX が 80000002H、80000003H、または 80000004H）に対して、CPUID 命令は、16 バイトのブランド・ストリングを EAX、EBX、ECX、EDX レジスタに返す。プロセッサから返される ASCII 文字が 47 文字未満のこともある。この場合でも、ストリングは NULL で終わり、各 CPUID 入力値（80000002H、80000003H、80000004H）に対して有効なデータを返す。

表 2-11. は、インテル Pentium 4 プロセッサ・ファミリの最初のプロセッサが返すブランド・ストリングを示している。

注記

ブランド・ストリング内に周波数が示されている場合、この周波数は、そのプロセッサに対して定義されている最大周波数を表すもので、実際の動作周波数を表しているのではない。

以下のプロシージャを使用して、ブランド・ストリングを検出することができる。

1. EAX の入力値を 80000000H に設定して、CPUID 命令を実行する。
2. ((EAX_Return_Value) AND (80000000H) ≠ 0) の場合、プロセッサは拡張機能 CPUID サポートしており、EAX には、拡張機能でサポートされている最大入力値が格納される。
3. EAX_Return_Value ≥ 80000004H の場合、CPUID 命令はブランド・ストリング機能をサポートしている。

表 2-11. 返されるプロセッサ・ブランド・ストリング

EAX 入力値	戻り値	対応する ASCII ストリング
80000002H	EAX = 20202020H EBX = 20202020H ECX = 20202020H EDX = 6E492020H;	“” “” “” “nI ”
80000003H	EAX = 286C6574H EBX = 50202952H ECX = 69746E65H EDX = 52286D75H	“(let” “P)R” “itne” “R(mu”
80000004H	EAX = 20342029H; EBX = 20555043H; ECX = 30303531H EDX = 007A484DH	“ 4)” “ UPC” “0051” “\0zHM”

将来のプロセッサは、MHz ではなく GHz で周波数を返す場合がある。CPUID 命令を使って IA-32 プロセッサを識別する際、ブランド識別ソフトウェアは、以下の優先度順でブランド識別方法を使用する必要がある。

- プロセッサ・ブランド・ストリング
- プロセッサ・ブランド・インデックスと、ソフトウェアが提供するブランド・ストリング・テーブル
- CPUID 命令によって返される情報（タイプ、ファミリ、モデル、ステッピング、キャッシュ）を利用したテーブルベースのメカニズム

IA-32 アーキテクチャにおける互換性

CPUID 命令は、初期モデルの Intel486™ プロセッサまたは Intel486 プロセッサよりも以前のすべての IA-32 プロセッサではサポートされていない。

影響を受けるフラグ

なし。

例外（すべての動作モード）

なし。

注記

CPUID 命令をサポートしていない初期の IA-32 プロセッサでは、CPUID 命令を実行すると、無効オペコード（#UD）例外が発生する。

CVTDQ2PD - Convert Packed Doubleword Integers to Packed Double-Precision Floating-Point Values

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
F3 0F E6	CVTDQ2PD *xmm1, xmm2/m64*	有効	有効	*xmm2/m128* の 2 つのパックド符号付きダブルワード整数を *xmm1* の 2 つのパックド倍精度浮動小数点値に変換する。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。

SIMD 浮動小数点例外

なし。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#UD	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#UD	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#UD	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

CVTDQ2PS - Convert Packed Doubleword Integers to Packed Single-Precision Floating-Point Values

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
0F 5B /r	CVTDQ2PS *xmm1*, *xmm2/m128*	有効	有効	*xmm2/m128* の 4 つのパックド符号付きダブルワード整数を *xmm1* の 4 つのパックド単精度浮動小数点値に変換する。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。

SIMD 浮動小数点例外

精度。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

実アドレスモード例外

#GP(0)	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

CVTPD2DQ - Convert Packed Double-Precision Floating-Point Values to Packed Doubleword Integers

オペコード	命令	64 ビット モード	互換 / レガシー モード	説明
F2 0F E6	CVTPD2DQ *xmm1*, *xmm2/m128*	有効	有効	*xmm2/m128* の 2 つのパックド倍精度浮動小数点値を *xmm1* の 2 つのパックド符号付きダブルワード整数に変換する。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。

SIMD 浮動小数点例外

無効、精度。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

実アドレスモード例外

#GP(0)	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

CVTPD2PI - Convert Packed Double-Precision Floating-Point Values to Packed Doubleword Integers

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
66 0F 2D /r	CVTPD2PI *mm*, *xmm/m128*	有効	有効	*xmm/m128* の 2 つのパックド倍精度浮動小数点値を *mm* の 2 つのパックド符号付きダブルワード整数に変換する。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。

SIMD 浮動小数点例外

無効、精度。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#MF	未処理の x87 FPU 例外がある場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

実アドレスモード例外

#GP(0)	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#MF	未処理の x87 FPU 例外がある場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

CVTPD2PS - Covert Packed Double-Precision Floating-Point Values to Packed Single-Precision Floating-Point Values

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
66 0F 5A /r	CVTPD2PS *xmm1, xmm2/m128*	有効	有効	*xmm2/m128* の 2 つのパックド倍精度浮動小数点値を *xmm1* の 2 つのパックド単精度浮動小数点値に変換する。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。

SIMD 浮動小数点例外

オーバーフロー、アンダーフロー、無効、精度、デノーマル。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

実アドレスモード例外

#GP(0)	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

CVTPI2PD - Convert Packed Doubleword Integers to Packed Double-Precision Floating-Point Values

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
66 0F 2A /r	CVTPI2PD *xmm*, *mm/m64*	有効	有効	*mm/mem64* の 2 つのパックド符号付きダブルワード整数を *xmm* の 2 つのパックド倍精度浮動小数点値に変換する。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。

SIMD 浮動小数点例外

なし。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#UD	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#UD	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#UD	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

CVTPI2PS - Convert Packed Doubleword Integers to Packed Single-Precision Floating-Point Values

オペコード	命令	64 ビット モード	互換 / レガシー モード	説明
0F 2A /r	CVTPI2PS *xmm*, *mm*/*m64*	有効	有効	*mm*/*m64* の 2 つの符号付きダブルワード整数を *xmm* の 2 つの単精度浮動小数点値に変換する。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。

SIMD 浮動小数点例外

精度。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

CVTPS2DQ - Convert Packed Single-Precision Floating-Point Values to Packed Doubleword Integers

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
66 0F 5B /r	CVTPS2DQ *xmm1*, *xmm2/m128*	有効	有効	*xmm2/m128* の 4 つのパックド単精度浮動小数点値を *xmm1* の 4 つのパックド符号付きダブルワード整数に変換する。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。

SIMD 浮動小数点例外

無効、精度。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

実アドレスモード例外

#GP(0)	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

CVTPS2PD - Covert Packed Single-Precision Floating-Point Values to Packed Double-Precision Floating-Point Values

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
0F 5A /r	CVTPS2PD *xmm1*, *xmm2/m64*	有効	有効	*xmm2/m64* の 2 つのパックド単精度浮動小数点値を *xmm1* の 2 つのパックド倍精度浮動小数点値に変換する。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。

SIMD 浮動小数点例外

無効。デノーマル。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

CVTPS2PI - Convert Packed Single-Precision Floating-Point Values to Packed Doubleword Integers

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
0F 2D /r	CVTPS2PI *mm, xmm/m64*	有効	有効	*xmm/m64* の 2 つのパックド単精度浮動小数点値を *mm* の 2 つのパックド符号付きダブルワード整数に変換する。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。

SIMD 浮動小数点例外

無効、精度。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#MF	未処理の x87 FPU 例外がある場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

CVTSD2SI - Convert Scalar Double-Precision Floating-Point Value to Doubleword Integer

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
F2 0F 2D /r	CVTSD2SI *r32*, *xmm/m64*	有効	有効	*xmm/m64* の 1 つの倍精度浮動小数点値を *r32* の 1 つの符号付きダブルワード整数に変換する。
REX.W + F2 0F 2D /r	CVTSD2SI *r64*, *xmm/m64*	有効	N.E.	*xmm/m64* の 1 つの倍精度浮動小数点値を、*r64* に符号拡張された 1 つの符号付きクワッドワード整数に変換する。

IA-32e モードでの操作

64 ビットに拡張される。
XMM8 ～ XMM15 へのアクセスが可能である。
新しいレジスタ R8 ～ R15 へのアクセスが可能である。

SIMD 浮動小数点例外

無効、精度。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

CVTSD2SS - Convert Scalar Double-Precision Floating-Point Value to Scalar Single-Precision Floating-Point Value

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
F2 0F 5A /r	CVTSD2SS *xmm1, xmm2/m64*	有効	有効	*xmm2/m64* の 1 つの倍精度浮動小数点値を *xmm1* の 1 つの単精度浮動小数点値に変換する。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。

SIMD 浮動小数点例外

オーバーフロー、アンダーフロー、無効、精度、デノーマル。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

CVTSI2SD - Convert Doubleword Integer to Scalar Double-Precision Floating-Point Value

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
F2 0F 2A /r	CVTSI2SD *xmm*, r/m32	有効	有効	*r/m32* の 1 つの符号付きダブルワード整数を *xmm* の 1 つの倍精度浮動小数点値に変換する。
REX.W + F2 0F 2A /r	CVTSI2SD *xmm*, r/m64	有効	N.E.	*r/m64* の 1 つの符号付きクワッドワード整数を *xmm* の 1 つの倍精度浮動小数点値に変換する。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。
64 ビットに拡張される。
新しいレジスタ R8 ～ R15 へのアクセスが可能である。

SIMD 浮動小数点例外

なし。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

CVTSI2SS - Convert Doubleword Integer to Scalar Single-Precision Floating-Point Value

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
F3 0F 2A /r	CVTSI2SS *xmm*, *r/m32*	有効	有効	*r/m32* の 1 つの符号付きダブルワード整数を *xmm* の 1 つの単精度浮動小数点値に変換する。
REX.W + F3 0F 2A /r	CVTSI2SS *xmm*, *r/m64*	有効	N.E.	*r/m64* の 1 つの符号付きクワッドワード整数を *xmm* の 1 つの単精度浮動小数点値に変換する。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。
XMMn[31:0] = CVT(reg/mem64)、XMMn[127:32] = 変更なし。
64 ビットに拡張される。
新しいレジスタ R8 ～ R15 へのアクセスが可能である。

SIMD 浮動小数点例外

精度。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

CVTSS2SD - Convert Scalar Single-Precision Floating-Point Value to Scalar Double-Precision Floating-Point Value

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
F3 0F 5A /r	CVTSS2SD *xmm1, xmm2/m32*	有効	有効	*xmm2/m32* の 1 つの単精度浮動小数点値を *xmm1* の 1 つの倍精度浮動小数点値に変換する。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。
XMMn[63:0] = CVT(reg/mem32)、XMMn[127:64] = 変更なし。

SIMD 浮動小数点例外

無効。デノーマル。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

CVTSS2SI - Convert Scalar Single-Precision Floating-Point Value to Doubleword Integer

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
F3 0F 2D /r	CVTSS2SI *r32*, *xmm/m32*	有効	有効	*xmm/m32* の 1 つの単精度浮動小数点値を *r32* の 1 つの符号付きダブルワード整数に変換する。
REX.W + F3 0F 2D /r	CVTSS2SI *r64*, *xmm/m32*	有効	N.E.	*xmm/m32* の 1 つの単精度浮動小数点値を *r64* の 1 つの符号付きクワッドワード整数に変換する。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。
64 ビットに拡張される。
新しいレジスタ R8 ～ R15 へのアクセスが可能である。

SIMD 浮動小数点例外

無効、精度。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

CVTTPD2PI - Convert with Truncation Packed Double-Precision Floating-Point Values to Packed Doubleword Integers

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
66 0F 2C /r	CVTTPD2PI *mm*, *xmm/m128*	有効	有効	切り捨てを使用して、*xmm/m128* の 2 つのパックド倍精度浮動小数点値を *mm* の 2 つのパックド符号付きダブルワード整数に変換する。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。

SIMD 浮動小数点例外

無効、精度。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#MF	未処理の x87 FPU 例外がある場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

実アドレスモード例外

#GP(0)	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

CVTTPD2DQ - Convert with Truncation Packed Double-Precision Floating-Point Values to Packed Doubleword Integers

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
66 0F E6	CVTTPD2DQ *xmm1, xmm2/m128*	有効	有効	切り捨てを使用して、*xmm2/m128* の 2 つのパックド倍精度浮動小数点値を *xmm1* の 2 つのパックド符号付きダブルワード整数に変換する。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。

SIMD 浮動小数点例外

無効、精度。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

実アドレスモード例外

#GP(0)	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

CVTTPS2DQ - Convert with Truncation Packed Single-Precision Floating-Point Values to Packed Doubleword Integers

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
F3 0F 5B /r	CVTTPS2DQ *xmm1*, *xmm2/m128*	有効	有効	切り捨てを使用して、*xmm2/m128* の 4 つの単精度浮動小数点値を *xmm1* の 4 つの符号付きダブルワード整数に変換する。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。

SIMD 浮動小数点例外

無効、精度。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

実アドレスモード例外

#GP(0)	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

CVTTPS2PI - Convert with Truncation Packed Single-Precision Floating-Point Values to Packed Doubleword Integers

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
0F 2C /r	CVTTPS2PI *mm*, *xmm/m64*	有効	有効	切り捨てを使用して、*xmm/m64* の 2 つの単精度浮動小数点値を *mm* の 2 つの符号付きダブルワード整数に変換する。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。

SIMD 浮動小数点例外

無効、精度。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#MF	未処理の x87 FPU 例外がある場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

CVTTSD2SI - Convert with Truncation Scalar Double-Precision Floating-Point Value to Signed Doubleword Integer

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
F2 0F 2C /r	CVTTSD2SI *r32*, *xmm/m64*	有効	有効	切り捨てを使用して、*xmm/m64* の 1 つの倍精度浮動小数点値を *r32* の 1 つの符号付きダブルワード整数に変換する。
REX.W + F2 0F 2C /r	CVTTSD2SI *r64*, *xmm/m64*	有効	N.E.	切り捨てを使用して、xmm/m64 の 1 つの倍精度浮動小数点値を r64 の 1 つの符号付きクワッドワード整数に変換する。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。
64 ビットに拡張される。
新しいレジスタ R8 ～ R15 へのアクセスが可能である。

SIMD 浮動小数点例外

無効、精度。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

CVTTSS2SI - Convert with Truncation Scalar Single-Precision Floating-Point Value to Doubleword Integer

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
F3 0F 2C /r	CVTTSS2SI *r32*, *xmm*/*m32*	有効	有効	切り捨てを使用して、*xmm/m32* の 1 つの単精度浮動小数点値を *r32* の 1 つの符号付きダブルワード整数に変換する。
REX.W + F3 0F 2C /r	CVTTSS2SI *r64*, *xmm*/*m32*	有効	N.E.	切り捨てを使用して、*xmm/m32* の 1 つの単精度浮動小数点値を *r64* の 1 つの符号付きクワッドワード整数に変換する。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。
64 ビットに拡張される。
新しいレジスタ R8 ～ R15 へのアクセスが可能である。

SIMD 浮動小数点例外

無効、精度。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

CWD/CDQ/CQQ - Convert Word to Doubleword/Convert Doubleword to Quadword/Convert Quadword to Double Quadword

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
99	CWD	有効	有効	DX:AX ← AX の符号拡張
99	CDQ	有効	有効	EDX:EAX ← EAX の符号拡張
REX.W + 99	CQO	有効	N.E.	RDX:RAX← RAX の符号拡張

影響を受けるフラグ

なし。

IA-32e モードでの操作

命令が 64 ビットに拡張される。
デフォルトの操作サイズはデスティネーション・レジスタのサイズである。

例外（すべての動作モード）

なし。

「CBW/CWDE/CDQE - Convert Byte to Word/Convert Word to Doubleword/Convert Doubleword to Quadword」を参照のこと。

DAA - Decimal Adjust AL after Addition

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
27	DAA	無効	有効	加算後に AL を 10 進調整する。

影響を受けるフラグ

値の調整によって結果のどちらかの桁に10進キャリーが生じた場合は、CFおよびAFフラグがセットされる。SF、ZF、PFが結果に従ってセットされる。OFフラグは未定義。

IA-32e モードでの操作

64 ビットモードでは無効である。

保護モード例外

なし。

実アドレスモード例外

なし。

仮想 8086 モード例外

なし。

互換モード例外

なし。

64 ビットモード例外

#UD	64 ビットモードの場合。

DAS - Decimal Adjust AL after Subtraction

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
2F	DAS	無効	有効	減算後に AL を 10 進調整する。

影響を受けるフラグ

値の調整によって結果のどちらかの桁に10進ボローが生じた場合は、CFおよびAFフラグがセットされる。SF、ZF、PFが結果に従ってセットされる。OFフラグは未定義。

IA-32e モードでの操作

64 ビットモードでは無効である。

保護モード例外

なし。

実アドレスモード例外

なし。

仮想 8086 モード例外

なし。

互換モード例外

なし。

64 ビットモード例外

#UD	64 ビットモードの場合。

DEC - Decrement by 1

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
FE /1	DEC *r/m8*	有効	有効	*r/m8* を 1 デクリメントする。
REX + FE /1	DEC *r/m8**	有効	N.E.	*r/m8* を 1 デクリメントする。
FF /1	DEC *r/m16*	有効	有効	*r/m16* を 1 デクリメントする。
FF /1	DEC *r/m32*	有効	有効	*r/m32* を 1 デクリメントする。
REX.W + FF /1	DEC *r/m64*	有効	N.E.	*r/m64* を 1 デクリメントする。
48+rw	DEC *r16*	N.E.	有効	*r16* を 1 デクリメントする。
48+rd	DEC *r32*	N.E.	有効	*r32* を 1 デクリメントする。

* 64 ビットモードでは、REX プリフィックスを使用する場合、バイトレジスタ AH、BH、CH、DH にアクセスするように r/m8 をコード化することはできない。1.4.2.2. 節も参照のこと。

影響を受けるフラグ

CF フラグは影響を受けない。OF、SF、ZF、AF、PF フラグが結果に従ってセットされる。

IA-32e モードでの操作

命令が 64 ビットに拡張される。
デフォルトの操作サイズは 32 ビットである。
64 ビットモードでは、オペコード 48H ～ 4FH が REX プリフィックスである。
新しいレジスタ R8 ～ R15 へのアクセスが可能である。

保護モード例外

#GP(0)	デスティネーション・オペランドが書き込み不可能なセグメントにある場合。
	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタの内容が NULL セグメント・セレクタの場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

DIV - Unsigned Divide

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
F6 /6	DIV *r/m8*	有効	有効	AX を *r/m8* で符号なし除算する。結果は次のようにストアされる。AL ← 商、AH ← 剰余
REX + F6 /6	DIV *r/m8**	有効	N.E.	AX を *r/m8* で符号なし除算する。結果は次のようにストアされる。AL ← 商、AH ← 剰余
F7 /6	DIV *r/m16*	有効	有効	DX:AX を *r/m16* で符号なし除算する。結果は次のようにストアされる。AX ← 商、DX ← 剰余
F7 /6	DIV *r/m32*	有効	有効	EDX:EAX を *r/m32* で符号なし除算する。結果は次のようにストアされる。EAX ← 商、EDX ← 剰余
REX.W + F7 /6	DIV *r/m64*	有効	N.E.	RDX:RAX を *r/m64* で符号なし除算する。結果は次のようにストアされる。RAX ← 商、RDX ← 剰余

* 64 ビットモードでは、REX プリフィックスを使用する場合、バイトレジスタ AH、BH、CH、DH にアクセスするように r/m8 をコード化することはできない。1.4.2.2. 節も参照のこと。

影響を受けるフラグ

CF、OF、SF、ZF、AF、PF フラグは未定義。

IA-32e モードでの操作

命令が 64 ビットに拡張される。
デフォルトの操作サイズは 32 ビットである。
64 ビット操作では、RAX に 64 ビットの商、RDX に 64 ビットの剰余がストアされる。
新しいレジスタ R8 ～ R15 へのアクセスが可能である。

保護モード例外

#DE	ソース・オペランド（除数）が 0 の場合。
	商が大きすぎて、指定されたレジスタにストアできない場合。
#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタの内容が NULL セグメント・セレクタの場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#DE	ソース・オペランド（除数）が 0 の場合。
	商が大きすぎて、指定されたレジスタにストアできない場合。
#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタの内容が NULL セグメント・セレクタの場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。

仮想 8086 モード例外

#DE	ソース・オペランド（除数）が 0 の場合。
	商が大きすぎて、指定されたレジスタにストアできない場合。
#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#DE	ソース・オペランド（除数）が 0 の場合。
	商が大きすぎて、指定されたレジスタにストアできない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

DIVPD - Divide Packed Double-Precision Floating-Point Values

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
66 0F 5E /r	DIVPD *xmm1*, *xmm2/m128*	有効	有効	*xmm1* のパックド倍精度浮動小数点値を *xmm2/m128* のパックド倍精度浮動小数点値で割る。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。

SIMD 浮動小数点例外

オーバーフロー、アンダーフロー、無効、ゼロ除算、精度、デノーマル。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

実アドレスモード例外

#GP(0)	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

DIVPS - Divide Packed Single-Precision Floating-Point Values

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
0F 5E /r	DIVPS *xmm1*, *xmm2/m128*	有効	有効	*xmm1* のパックド単精度浮動小数点値を *xmm2/m128* のパックド単精度浮動小数点値で割る。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。

SIMD 浮動小数点例外

オーバーフロー、アンダーフロー、無効、ゼロ除算、精度、デノーマル。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。

実アドレスモード例外

#GP(0)	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。

DIVSD - Divide Scalar Double-Precision Floating-Point Values

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
F2 0F 5E /r	DIVSD *xmm1*, *xmm2/m64*	有効	有効	*xmm1* の下位の倍精度浮動小数点値を *xmm2/mem64* の下位の倍精度浮動小数点値で割る。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。

SIMD 浮動小数点例外

オーバーフロー、アンダーフロー、無効、ゼロ除算、精度、デノーマル。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE2 が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

DIVSS - Divide Scalar Single-Precision Floating-Point Values

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
F3 0F 5E /r	DIVSS *xmm1*, *xmm2/m32*	有効	有効	*xmm1* の下位の単精度浮動小数点値を *xmm2/m32* の下位の単精度浮動小数点値で割る。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。

SIMD 浮動小数点例外

オーバーフロー、アンダーフロー、無効、ゼロ除算、精度、デノーマル。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

EMMS - Empty MMX State

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
0F 77	EMMS	有効	有効	x87 FPU タグワードを空に設定する。

影響を受けるフラグ

なし。

IA-32e モードでの操作

レガシーモードと同じ。

保護モード例外

#UD	CR0 の EM がセットされた場合。
#NM	CR0 の TS がセットされた場合。
#MF	未処理の FPU 例外がある場合。

実アドレスモード例外

保護モード例外と同じ。

仮想 8086 モード例外

保護モード例外と同じ。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

ENTER - Make Stack Frame for Procedure Parameters

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
C8 *iw* 00	ENTER *imm16*,0	有効	有効	プロシージャのスタックフレームを作成する。
C8 *iw* 01	ENTER *imm16*,1	有効	有効	プロシージャのネストされたスタックフレームを作成する。
C8 *iw* ib	ENTER *imm16,imm8*	有効	有効	プロシージャのネストされたスタックフレームを作成する。

影響を受けるフラグ

なし。

IA-32e モードでの操作

デフォルトの操作サイズは64 ビットである。
64 ビットモードでは、32 ビットの操作サイズをコード化することはできない。

保護モード例外

#SS(0)	SP または ESP レジスタの新しい値がスタック・セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。

実アドレスモード例外

#SS(0)	SP または ESP レジスタの新しい値がスタック・セグメントの範囲外の場合。

仮想 8086 モード例外

#SS(0)	SP または ESP レジスタの新しい値がスタック・セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。

F2XM1 - Compute 2^x-1

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
D9 F0	F2XM1	有効	有効	ST(0) を $(2^{ST(0)} - 1)$ で置き換える。

影響を受ける FPU フラグ

C1	スタック・アンダーフローが発生した場合は 0 にセットされる。 不正確結果例外（#P）が発生した場合は、丸めの方向を示す。0 ← 切り上げなし、1 ← 切り上げ。
C0、C2、C3	未定義。

IA-32e モードでの操作

レガシーモードと同じ。

浮動小数点例外

#IS	スタック・アンダーフローが発生した場合。
#IA	ソース・オペランドが SNaN 値であるか、またはそのフォーマットがサポートされていない場合。
#D	結果がデノーマル値である場合。
#U	結果が小さすぎて、デスティネーション・フォーマットで表現できない場合。
#P	値がデスティネーション・フォーマットでは正確に表現できない場合。

保護モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

実アドレスモード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

FABS - Absolute Value

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
D9 E1	FABS	有効	有効	ST をその絶対値で置き換える。

影響を受ける FPU フラグ

C1　スタック・アンダーフローが発生した場合は 0 にセットされ、発生しなかった場合は 0 にクリアされる。

C0、C2、C3　未定義。

IA-32e モードでの操作

レガシーモードと同じ。

浮動小数点例外

#IS　スタック・アンダーフローが発生した場合。

保護モード例外

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

実アドレスモード例外

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

FADD/FADDP/FIADD - Add

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
D8 /0	FADD *m32fp*	有効	有効	*m32fp* を ST(0) に加え、結果を ST(0) にストアする。
DC /0	FADD *m64fp*	有効	有効	*m64fp* を ST(0) に加え、結果を ST(0) にストアする。
D8 C0+i	FADD ST(0), ST(i)	有効	有効	ST(0) を ST(i) に加え、結果を ST(0) にストアする。
DC C0+i	FADD ST(i), ST(0)	有効	有効	ST(i) を ST(0) に加え、結果を ST(i) にストアする。
DE C0+i	FADDP ST(i), ST(0)	有効	有効	ST(0) を ST(i) に加え、結果を ST(i) にストアし、レジスタスタックをポップする。
DE C1	FADDP	有効	有効	ST(0) を ST(1) に加え、結果を ST(1) にストアし、レジスタスタックをポップする。
DA /0	FIADD *m32int*	有効	有効	*m32int* を ST(0) に加え、結果を ST(0) にストアする。
DE /0	FIADD *m16int*	有効	有効	*m16int* を ST(0) に加え、結果を ST(0) にストアする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1	スタック・アンダーフローが発生した場合は 0 にセットされる。 不正確結果例外（#P）が発生した場合は、丸めの方向を示す。0 ← 切り上げなし、1 ← 切り上げ。
C0、C2、C3	未定義。

浮動小数点例外

#IS	スタック・アンダーフローが発生した場合。
#IA	オペランドが SNaN 値であるか、またはそのフォーマットがサポートされていない場合。 両方のオペランドの符号が反対で、絶対値が無限大の場合。
#D	ソース・オペランドがデノーマル値である場合。
#U	結果が小さすぎて、デスティネーション・フォーマットで表現できない場合。
#O	結果が大きすぎて、デスティネーション・フォーマットで表現できない場合。
#P	値がデスティネーション・フォーマットでは正確に表現できない場合。

保護モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタの内容が NULL セグメント・セレクタの場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

FBLD - Load Binary Coded Decimal

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
DF /4	FBLD *m80 dec*	有効	有効	BCD 値を浮動小数点に変換し、FPU スタックにプッシュする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1	スタック・オーバーフローが発生した場合は 1 にセットされ、発生しなかった場合は 0 にクリアされる。
C0、C2、C3	未定義。

浮動小数点例外

#IS	スタック・オーバーフローが発生した場合。

保護モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。 DS、ES、FS、または GS レジスタの内容が NULL セグメント・セレクタの場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

FBSTP - Store BCD Integer and Pop

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
DF /6	FBSTP m80bcd	有効	有効	ST(0) を m80bcd にストアし、ST(0) をポップする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1	スタック・アンダーフローが発生した場合は 0 にセットされる。 不正確結果例外（#P）が発生した場合は、丸めの方向を示す。0 = 切り上げなし、1 ← 切り上げ。
C0、C2、C3	未定義。

浮動小数点例外

#IS	スタック・アンダーフローが発生した場合。
#IA	ソース・オペランドが空の場合、NaN、±∞、またはサポートしていないフォーマットを含む場合、18 桁の BCD 数よりも長い値を含む場合。
#P	値がデスティネーション・フォーマットでは正確に表現できない場合。

保護モード例外

#GP(0)	セグメント・レジスタに書き込み不可能なセグメントを指示先とするセグメント・セレクタがロードされようとした場合。 メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。 DS、ES、FS、または GS レジスタの内容が NULL セグメント・セレクタの場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

FCHS - Change Sign

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
D9 E0	FCHS	有効	有効	ST(0) の符号を反転する。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1	スタック・アンダーフローが発生した場合は 0 にセットされ、発生しなかった場合は 0 にクリアされる。
C0、C2、C3	未定義。

浮動小数点例外

#IS	スタック・アンダーフローが発生した場合。

保護モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

実アドレスモード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

FCLEX/FNCLEX - Clear Exceptions

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
9B DB E2	FCLEX	有効	有効	未処理のマスクされていない浮動小数点例外の有無をチェックした後、浮動小数点例外フラグをクリアする。
DB E2	FNCLEX*	有効	有効	未処理のマスクされていない浮動小数点例外の有無をチェックしないで、浮動小数点例外フラグをクリアする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

FPU ステータス・ワード内の PE、UE、OE、ZE、DE、IE、ES、SF、B フラグがクリアされる。C0、C1、C2、C3 フラグは未定義。

浮動小数点例外

なし。

保護モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

実アドレスモード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

FCMOV*cc* - Floating-Point Conditional Move

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
DA C0+i	FCMOVB ST(0), ST(i)	有効	有効	より下（CF=1）の場合転送する。
DA C8+i	FCMOVE ST(0), ST(i)	有効	有効	等しい（ZF=1）場合転送する。
DA D0+i	FCMOVBE ST(0), ST(i)	有効	有効	より下か等しい（CF=1 または ZF=1）場合転送する。
DA D8+i	FCMOVU ST(0), ST(i)	有効	有効	順序付けなし（PF=1）の場合転送する。
DB C0+i	FCMOVNB ST(0), ST(i)	有効	有効	より下でない（CF=0）場合転送する。
DB C8+i	FCMOVNE ST(0), ST(i)	有効	有効	等しくない（ZF=0）場合転送する。
DB D0+i	FCMOVNBE ST(0), ST(i)	有効	有効	より下でなく等しくない（CF=0 および ZF=0）場合転送する。
DB D8+i	FCMOVNU ST(0), ST(i)	有効	有効	順序付け（PF=0）の場合転送する。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1　スタック・アンダーフローが発生した場合は 0 にセットされる。

C0、C2、C3　未定義。

浮動小数点例外

#IS　スタック・アンダーフローが発生した場合。

影響を受ける整数フラグ

なし。

保護モード例外

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

実アドレスモード例外

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

FCOM/FCOMP/FCOMPP - Compare Floating Point Values

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
D8 /2	FCOM *m32fp*	有効	有効	ST(0) を *m32fp* と比較する。
DC /2	FCOM *m64fp*	有効	有効	ST(0) を *m64fp* と比較する。
D8 D0+i	FCOM ST(i)	有効	有効	ST(0) を ST(i) と比較する。
D8 D1	FCOM	有効	有効	ST(0) を ST(1) と比較する。
D8 /3	FCOMP *m32fp*	有効	有効	ST(0) を *m32fp* と比較し、レジスタスタックをポップする。
DC /3	FCOMP *m64fp*	有効	有効	ST(0) を m64fp と比較し、レジスタスタックをポップする。
D8 D8+i	FCOMP ST(i)	有効	有効	ST(0) を ST(i) と比較し、レジスタスタックをポップする。
D8 D9	FCOMP	有効	有効	ST(0) を ST(1) と比較し、レジスタスタックをポップする。
DE D9	FCOMPP	有効	有効	ST(0) を ST(1) と比較し、レジスタスタックを 2 回ポップする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1　　0 にクリアされる。

C0、C2、C3　　以下の表を参照。

条件	C3	C2	C0
ST(0) > SRC	0	0	0
ST(0) < SRC	0	0	1
ST(0) = SRC	1	0	0
順序付けなし *	1	1	1

浮動小数点例外

#IS　　スタック・アンダーフローが発生した場合。

#IA　　一方または両方のオペランドが NaN 値であるか、またはそれらのフォーマットがサポートされていない場合。

レジスタが空にマークされる場合。

#D　　一方または両方のオペランドがデノーマル値である場合。

保護モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタの内容が NULL セグメント・セレクタの場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

FCOMI/FCOMIP/FUCOMI/FUCOMIP - Compare Floating Point Values and Set EFLAGS

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
DB F0+i	FCOMI ST, ST(i)	有効	有効	ST(0) を ST(i) と比較し、結果に従ってステータス・フラグをセットする。
DF F0+i	FCOMIP ST, ST(i)	有効	有効	ST(0) を ST(i) と比較し、結果に従ってステータス・フラグをセットし、レジスタスタックをポップする。
DB E8+i	FUCOMI ST, ST(i)	有効	有効	ST(0) を ST(i) と比較し、順序付け値の有無をチェックし、結果に従ってステータス・フラグをセットする。
DF E8+i	FUCOMIP ST, ST(i)	有効	有効	ST(0) を ST(i) と比較し、順序付け値の有無をチェックし、結果に従ってステータス・フラグをセットし、レジスタスタックをポップする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1	0 にクリアされる。
C0、C2、C3	影響を受けない。

浮動小数点例外

#IS	スタック・アンダーフローが発生した場合。
#IA	（FCOMI または FCOMIP 命令）一方または両方のオペランドが NaN 値であるか、またはそれらのフォーマットがサポートされていない場合。 （FUCOMI または FUCOMIP 命令）一方または両方のオペランドが SNaN 値である（ただし、QNaN ではない）か、またはそれらのフォーマットが定義されていない場合。QNaN 値が検出されても、無効オペランド例外は発生しない。

保護モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

実アドレスモード例外

#MF	未処理の x87 FPU 例外がある場合。
#NM	CR0 の EM または TS がセットされた場合。

仮想 8086 モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

FCOS - Cosine

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
D9 FF	FCOS	有効	有効	ST(0) をその余弦で置き換える。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1	スタック・アンダーフローが発生した場合は 0 にセットされる。 不正確結果例外（#P）が発生した場合は、丸めの方向を示す。0 ← 切り上げなし、1 ← 切り上げ。 C2 が 1 の場合は未定義。
C2	範囲（-2^{63} ＜ソース・オペランド＜ $+2^{63}$）外の場合は 1 にセットされる。範囲内の場合は 0 にクリアされる。
C0、C3	未定義。

浮動小数点例外

#IS	スタック・アンダーフローが発生した場合。
#IA	ソース・オペランドが SNaN 値または ∞ であるか、そのフォーマットがサポートされていない場合。
#D	結果がデノーマル値である場合。
#U	結果が小さすぎて、デスティネーション・フォーマットで表現できない場合。
#P	値がデスティネーション・フォーマットでは正確に表現できない場合。

保護モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

実アドレスモード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

FDECSTP - Decrement Stack-Top Pointer

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
D9 F6	FDECSTP	有効	有効	FPU ステータス・ワードの TOP フィールドをデクリメントする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1 フラグは 0 にセットされる。C0、C2、C3 フラグは未定義。

浮動小数点例外

なし。

保護モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

実アドレスモード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

FDIV/FDIVP/FIDIV - Divide

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
D8 /6	FDIV *m32fp*	有効	有効	ST(0) を *m32fp* で割り、結果を ST(0) にストアする。
DC /6	FDIV *m64fp*	有効	有効	ST(0) を *m64fp* で割り、結果を ST(0) にストアする。
D8 F0+i	FDIV ST(0), ST(i)	有効	有効	ST(0) を ST(i) で割り、結果を ST(0) にストアする。
DC F8+i	FDIV ST(i), ST(0)	有効	有効	ST(i) を ST(0) で割り、結果を ST(i) にストアする。
DE F8+i	FDIVP ST(i), ST(0)	有効	有効	ST(i) を ST(0) で割り、結果を ST(i) にストアし、レジスタスタックをポップする。
DE F9	FDIVP	有効	有効	ST(1) を ST(0) で割り、結果を ST(1) にストアし、レジスタスタックをポップする。
DA /6	FIDIV *m32int*	有効	有効	ST(0) を *m32int* で割り、結果を ST(0) にストアする。
DE /6	FIDIV *m16int*	有効	有効	ST(0) を *m64int* で割り、結果を ST(0) にストアする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1　スタック・アンダーフローが発生した場合は 0 にセットされる。

不正確結果例外（#P）が発生した場合は、丸めの方向を示す。0 ← 切り上げなし、1 ← 切り上げ。

C0、C2、C3　未定義。

浮動小数点例外

#IS　スタック・アンダーフローが発生した場合。

#IA　オペランドが SNaN 値であるか、またはそのフォーマットがサポートされていない場合。

±∞/±∞、±0/±0 の場合。

#D　結果がデノーマル値である場合。

#Z　DEST/±0 の場合。ただし、DEST は ±0 に等しくない。

#U　結果が小さすぎて、デスティネーション・フォーマットで表現できない場合。

#O　結果が大きすぎて、デスティネーション・フォーマットで表現できない場合。

#P	値がデスティネーション・フォーマットでは正確に表現できない場合。

保護モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタの内容が NULL セグメント・セレクタの場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#MF	未処理の x87 FPU 例外がある場合。
#NM	CR0 の EM または TS がセットされた場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

FDIVR/FDIVRP/FIDIVR - Reverse Divide

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
D8 /7	FDIVR *m32fp*	有効	有効	*m32fp* を ST(0) で割り、結果を ST(0) にストアする。
DC /7	FDIVR *m64fp*	有効	有効	*m64fp* を ST(0) で割り、結果を ST(0) にストアする。
D8 F8+i	FDIVR ST(0), ST(i)	有効	有効	ST(i) を ST(0) で割り、結果を ST(0) にストアする。
DC F0+i	FDIVR ST(i), ST(0)	有効	有効	ST(0) を ST(i) で割り、結果を ST(i) にストアする。
DE F0+i	FDIVRP ST(i), ST(0)	有効	有効	ST(0) を ST(i) で割り、結果を ST(i) にストアし、レジスタスタックをポップする。
DE F1	FDIVRP	有効	有効	ST(0) を ST(1) で割り、結果を ST(1) にストアし、レジスタスタックをポップする。
DA /7	FIDIVR *m32int*	有効	有効	*m32int* を ST(0) で割り、結果を ST(0) にストアする。
DE /7	FIDIVR *m16int*	有効	有効	*m16int* を ST(0) で割り、結果を ST(0) にストアする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1	スタック・アンダーフローが発生した場合は 0 にセットされる。
	不正確結果例外（#P）が発生した場合は、丸めの方向を示す。0 ← 切り上げなし、1 ← 切り上げ。
C0、C2、C3	未定義。

浮動小数点例外

#IS	スタック・アンダーフローが発生した場合。
#IA	オペランドが SNaN 値であるか、またはそのフォーマットがサポートされていない場合。
	±∞/±∞、±0/±0 の場合。
#D	結果がデノーマル値である場合。
#Z	SCR/±0 の場合。ただし、SCR は ±0 に等しくない。
#U	結果が小さすぎて、デスティネーション・フォーマットで表現できない場合。
#O	結果が大きすぎて、デスティネーション・フォーマットで表現できない場合。

#P	値がデスティネーション・フォーマットでは正確に表現できない場合。

保護モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタの内容が NULL セグメント・セレクタの場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

FFREE - Free Floating-Point Register

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
DD C0+i	FFREE ST(i)	有効	有効	ST(i) のタグを空に設定する。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C0、C1、C2、C3 は未定義。

浮動小数点例外

なし。

保護モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

実アドレスモード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

FICOM/FICOMP - Compare Integer

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
DE /2	FICOM *m16int*	有効	有効	ST(0) を *m16int* と比較する。
DA /2	FICOM *m32int*	有効	有効	ST(0) を *m32int* と比較する。
DE /3	FICOMP *m16int*	有効	有効	ST(0) を *m16int* と比較し、スタックレジスタをポップする。
DA /3	FICOMP *m32int*	有効	有効	ST(0) を *m32int* と比較し、スタックレジスタをポップする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1	0 にセットされる。
C0、C2、C3	FCOM 命令の表を参照。

浮動小数点例外

#IS	スタック・アンダーフローが発生した場合。
#IA	一方または両方のオペランドが NaN 値であるか、またはそれらのフォーマットがサポートされていない場合。
#D	一方または両方のオペランドがデノーマル値である場合。

保護モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。 DS、ES、FS、または GS レジスタの内容が NULL セグメント・セレクタの場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

FILD - Load Integer

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
DF /0	FILD *m16int*	有効	有効	*m16int* を FPU レジスタスタックにプッシュする。
DB /0	FILD *m32int*	有効	有効	*m32int* を FPU レジスタスタックにプッシュする。
DF /5	FILD *m64int*	有効	有効	*m64int* を FPU レジスタスタックにプッシュする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1　スタック・オーバーフローが発生した場合は 1 にセットされ、発生しなかった場合は 0 にクリアされる。

C0、C2、C3　未定義。

浮動小数点例外

#IS　スタック・オーバーフローが発生した場合。

保護モード例外

#GP(0)　メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。

DS、ES、FS、または GS レジスタの内容が NULL セグメント・セレクタの場合。

#SS(0)　メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

#PF（フォルトコード）　ページフォルトが発生した場合。

#AC(0)　アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP　メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。

#SS　メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

FINCSTP - Increment Stack-Top Pointer

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
D9 F7	FINCSTP	有効	有効	FPU ステータス・レジスタの TOP フィールドをインクリメントする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1 フラグは 0 にセットされる。C0、C2、C3 フラグは未定義。

浮動小数点例外

なし。

保護モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

実アドレスモード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

FINIT/FNINIT - Initialize Floating-Point Unit

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
9B DB E3	FINIT	有効	有効	未処理のマスクされていない浮動小数点例外の有無をチェックした後、FPU を初期化する。
DB E3	FNINIT*	有効	有効	未処理のマスクされていない浮動小数点例外の有無をチェックしないで、FPU を初期化する。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C0、C1、C2、C3 が 0 にクリアされる。

浮動小数点例外

なし。

保護モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

実アドレスモード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

FIST/FISTP - Store Integer

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
DF /2	FIST *m16int*	有効	有効	ST(0) を *m16int* にストアする。
DB /2	FIST *m32int*	有効	有効	ST(0) を *m32int* にストアする。
DF /3	FISTP *m16int*	有効	有効	ST(0) を *m16int* にストアし、レジスタスタックをポップする。
DB /3	FISTP *m32int*	有効	有効	ST(0) を *m32int* にストアし、レジスタスタックをポップする。
DF /7	FISTP *m64int*	有効	有効	ST(0) を *m64int* にストアし、レジスタスタックをポップする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1	スタック・アンダーフローが発生した場合は 0 にセットされる。 不正確結果例外（#P）が発生した場合は、丸めの方向を示す。0 ← 切り上げなし、1 ← 切り上げ。 発生しなかった場合は 0 にクリアされる。
C0、C2、C3	未定義。

浮動小数点例外

#IS	スタック・アンダーフローが発生した場合。
#IA	ソース・オペランドが大きすぎて、デスティネーション・フォーマットで表現できない場合。 ソース・オペランドが NaN 値であるか、またはそのフォーマットがサポートされていない場合。
#P	値がデスティネーション・フォーマットでは正確に表現できない場合。

保護モード例外

#GP(0)	デスティネーションが書き込み不可能なセグメントにある場合。 メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。 DS、ES、FS、または GS レジスタを使用してメモリにアクセスしたが、その内容が NULL セグメント・セレクタであった場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

FISTTP - Store Integer with Truncation

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
DF /1	FISTTP *m16int*	有効	有効	切り捨てを使用して、ST(0) を *m16int* にストアする。
DB /1	FISTTP *m32int*	有効	有効	切り捨てを使用して、ST(0) を *m32int* にストアする。
DD /1	FISTTP *m64int*	有効	有効	切り捨てを使用して、ST(0) を *m64int* にストアする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1	0 にクリアされる。
C0、C2、C3	未定義。

浮動小数点例外

#IS	スタック・アンダーフローが発生した場合。
#IA	ソース・オペランドが大きすぎて、デスティネーション・フォーマットで表現できない場合。
	ソース・オペランドが NaN 値であるか、またはそのフォーマットがサポートされていない場合。
#P	値がデスティネーション・フォーマットでは正確に表現できない場合。

保護モード例外

#GP(0)	デスティネーションが書き込み不可能なセグメントにある場合。
	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。
#UD	CPUID 機能フラグ SSE3 が 0 の場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

FLD - Load Floating Point Value

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
D9 /0	FLD *m32fp*	有効	有効	*m32fp* を FPU レジスタスタックにプッシュする。
DD /0	FLD *m64fp*	有効	有効	*m64fp* を FPU レジスタスタックにプッシュする。
DB /5	FLD *m80fp*	有効	有効	*m80fp* を FPU レジスタスタックにプッシュする。
D9 C0+i	FLD ST(i)	有効	有効	ST(i) を FPU レジスタスタックにプッシュする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1　スタック・オーバーフローが発生した場合は 1 にセットされ、発生しなかった場合は 0 にクリアされる。

C0、C2、C3　未定義。

浮動小数点例外

#IS　スタック・オーバーフローが発生した場合。

#IA　ソース・オペランドが SNaN 値であるか、またはそのフォーマットがサポートされていない場合。ソース・オペランドが拡張倍精度浮動小数点形式の場合は発生しない。

#D　ソース・オペランドがデノーマル値である場合。ソース・オペランドが拡張倍精度浮動小数点形式の場合は発生しない。

保護モード例外

#GP(0)　デスティネーションが書き込み不可能なセグメントにある場合。

メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。

DS、ES、FS、または GS レジスタを使用してメモリにアクセスしたが、その内容が NULL セグメント・セレクタであった場合。

#SS(0)　メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

#PF（フォルトコード）　ページフォルトが発生した場合。

#AC(0)　アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

FLD1/FLDL2T/FLDL2E/FLDPI/FLDLG2/FLDLN2/FLDZ - Load Constant

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
D9 E8	FLD1	有効	有効	+1.0 を FPU レジスタスタックにプッシュする。
D9 E9	FLDL2T	有効	有効	$\log_2 10$ を FPU レジスタスタックにプッシュする。
D9 EA	FLDL2E	有効	有効	$\log_2 e$ を FPU レジスタスタックにプッシュする。
D9 EB	FLDPI	有効	有効	π を FPU レジスタスタックにプッシュする。
D9 EC	FLDLG2	有効	有効	$\log_{10} 2$ を FPU レジスタスタックにプッシュする。
D9 ED	FLDLN2	有効	有効	$\log_e 2$ を FPU レジスタスタックにプッシュする。
D9 EE	FLDZ	有効	有効	+0.0 を FPU レジスタスタックにプッシュする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1　スタック・オーバーフローが発生した場合は 1 にセットされ、発生しなかった場合は 0 にクリアされる。

C0、C2、C3　未定義。

浮動小数点例外

#IS　スタック・オーバーフローが発生した場合。

保護モード例外

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

実アドレスモード例外

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

IA-32 アーキテクチャにおける互換性

RC フィールドが「最も近い整数に丸める」（round-to-nearest）に設定されているときは、FPU は、インテル® 8087 およびインテル® 287 数値演算プロセッサが生じるものと同じ定数を生じる。

FLDCW - Load x87 FPU Control Word

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
D9 /5	FLDCW m2byte	有効	有効	*m2byte* から FPU 制御ワードをロードする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C0、C1、C2、C3 は未定義。

浮動小数点例外

なし。ただし、この操作で FPU ステータス・ワード内の未処理例外のマスクを解除されることがある。その場合は、次の「同期型」（waiting）浮動小数点命令が実行されたときにその例外が発生する。

保護モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタを使用してメモリにアクセスしたが、その内容が NULL セグメント・セレクタであった場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

FLDENV - Load x87 FPU Environment

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
D9 /4	FLDENV *m14/28byte*	有効	有効	*m14byte* または *m28byte* から FPU 環境をロードする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C0、C1、C2、C3 フラグがロードされる。

浮動小数点例外

なし。ただし、この操作でマスクされていない例外がステータス・ワードにロードされた場合は、次の「同期型」(waiting) 浮動小数点命令が実行されたときにその例外が発生する。

保護モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタを使用してメモリにアクセスしたが、その内容が NULL セグメント・セレクタであった場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

FMUL/FMULP/FIMUL - Multiply

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
D8 /1	FMUL *m32fp*	有効	有効	ST(0) に *m32fp* を掛け、結果を ST(0) にストアする。
DC /1	FMUL *m64fp*	有効	有効	ST(0) に *m64fp* を掛け、結果を ST(0) にストアする。
D8 C8+i	FMUL ST(0), ST(i)	有効	有効	ST(0) に ST(i) を掛け、結果を ST(0) にストアする。
DC C8+i	FMUL ST(i), ST(0)	有効	有効	ST(i) に ST(0) を掛け、結果を ST(i) にストアする。
DE C8+i	FMULP ST(i), ST(0)	有効	有効	ST(i) に ST(0) を掛け、結果を ST(i) にストアし、レジスタスタックをポップする。
DE C9	FMULP	有効	有効	ST(1) に ST(0) を掛け、結果を ST(1) にストアし、レジスタスタックをポップする。
DA /1	FIMUL *m32int*	有効	有効	ST(0) に *m32int* を掛け、結果を ST(0) にストアする。
DE /1	FIMUL *m16int*	有効	有効	ST(0) に *m16int* を掛け、結果を ST(0) にストアする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1	スタック・アンダーフローが発生した場合は 0 にセットされる。
	不正確結果例外（#P）フォルトが発生した場合は、丸めの方向を示す。0 ← 切り上げなし、1 ← 切り上げ。
C0、C2、C3	未定義。

浮動小数点例外

#IS	スタック・アンダーフローが発生した場合。
#IA	オペランドが SNaN 値であるか、またはそのフォーマットがサポートされていない場合。
	一方のオペランドが ±0 であり、他方のオペランドが ±∞ の場合。
#D	ソース・オペランドがデノーマル値である場合。
#U	結果が小さすぎて、デスティネーション・フォーマットで表現できない場合。
#O	結果が大きすぎて、デスティネーション・フォーマットで表現できない場合。
#P	値がデスティネーション・フォーマットでは正確に表現できない場合。

保護モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタを使用してメモリにアクセスしたが、その内容が NULL セグメント・セレクタであった場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

FNOP - No Operation

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
D9 D0	FNOP	有効	有効	何の操作も実行されない。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C0、C1、C2、C3 は未定義。

浮動小数点例外

なし。

保護モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

実アドレスモード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

FPATAN - Partial Arctangent

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
D9 F3	FPATAN	有効	有効	ST(1) を arctan(ST(1)/ST(0)) で置き換え、レジスタスタックをポップする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1　スタック・アンダーフローが発生した場合は 0 にセットされる。

不正確結果例外（#P）が発生した場合は、丸めの方向を示す。0 ← 切り上げなし、1 ← 切り上げ。

C0、C2、C3　未定義。

浮動小数点例外

#IS　スタック・アンダーフローが発生した場合。

#IA　ソース・オペランドが SNaN 値であるか、またはそのフォーマットがサポートされていない場合。

#D　ソース・オペランドがデノーマル値である場合。

#U　結果が小さすぎて、デスティネーション・フォーマットで表現できない場合。

#P　値がデスティネーション・フォーマットでは正確に表現できない場合。

保護モード例外

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

実アドレスモード例外

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

FPREM - Partial Remainder

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
D9 F8	FPREM	有効	有効	ST(0) を ST(1) で割って得られる剰余で ST(0) を置き換える。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C0	商のビット 2（Q2）にセットされる。
C1	スタック・アンダーフローが発生した場合は 0 にセットされ、発生しなかった場合は商の最下位ビット（Q0）にセットされる。
C2	縮小が完了した場合は 0 にセットされ、完了していない場合は 1 にセットされる。
C3	商のビット 1（Q1）にセットされる。

浮動小数点例外

#IS	スタック・アンダーフローが発生した場合。
#IA	ソース・オペランドが SNaN 値であるか、法（除数）が 0 であるか、被除数が ∞ であるか、またはそれらのいずれかのフォーマットがサポートされていない場合。
#D	ソース・オペランドがデノーマル値である場合。
#U	結果が小さすぎて、デスティネーション・フォーマットで表現できない場合。

保護モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

実アドレスモード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

FPREM1 - Partial Remainder

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
D9 F5	FPREM1	有効	有効	ST(0) を ST(1) で割って得られる IEEE 剰余で ST(0) を置き換える。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C0	商のビット 2（Q2）にセットされる。
C1	スタック・アンダーフローが発生した場合は 0 にセットされ、発生しなかった場合は商の最下位ビット（Q0）にセットされる。
C2	縮小が完了した場合は 0 にセットされ、完了していない場合は 1 にセットされる。
C3	商のビット 1（Q1）にセットされる。

浮動小数点例外

#IS	スタック・アンダーフローが発生した場合。
#IA	ソース・オペランドが SNaN 値であるか、法（除数）が 0 であるか、被除数が ∞ であるか、またはそれらのいずれかのフォーマットがサポートされていない場合。
#D	ソース・オペランドがデノーマル値である場合。
#U	結果が小さすぎて、デスティネーション・フォーマットで表現できない場合。

保護モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

実アドレスモード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

FPTAN - Partial Tangent

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
D9 F2	FPTAN	有効	有効	ST(0) をその正接で置き換え、FPU スタックに 1 をプッシュする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1　スタック・アンダーフローが発生した場合は 0 にセットされ、スタック・オーバーフローが発生した場合は 1 にセットされる。

不正確結果例外（#P）が発生した場合は、丸めの方向を示す。0 ← 切り上げなし、1 ← 切り上げ。

C2　範囲（-2^{63} ＜ソース・オペランド＜ $+2^{63}$）外の場合は 1 にセットされる。範囲内の場合は 0 にクリアされる。

C0、C3　未定義。

浮動小数点例外

#IS　スタック・アンダーフローまたはスタック・オーバーフローが発生した場合。

#IA　ソース・オペランドが SNaN 値または ∞ であるか、そのフォーマットがサポートされていない場合。

#D　ソース・オペランドがデノーマル値である場合。

#U　結果が小さすぎて、デスティネーション・フォーマットで表現できない場合。

#P　値がデスティネーション・フォーマットでは正確に表現できない場合。

保護モード例外

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

実アドレスモード例外

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

FRNDINT - Round to Integer

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
D9 FC	FRNDINT	有効	有効	ST(0) を整数に丸める。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1	スタック・アンダーフローが発生した場合は 0 にセットされる。 不正確結果例外（#P）が発生した場合は、丸めの方向を示す。0 ← 切り上げなし、1 ← 切り上げ。
C0、C2、C3	未定義。

浮動小数点例外

#IS	スタック・アンダーフローが発生した場合。
#IA	ソース・オペランドが SNaN 値であるか、またはそのフォーマットがサポートされていない場合。
#D	ソース・オペランドがデノーマル値である場合。
#P	ソース・オペランドが整数値でない場合。

保護モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

実アドレスモード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

FRSTOR - Restore x87 FPU State

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
DD /4	FRSTOR *m94/108byte*	有効	有効	*m94byte* または *m108byte* から FPU 状態をロードする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C0、C1、C2、C3 フラグがロードされる。

浮動小数点例外

なし。

保護モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタを使用してメモリにアクセスしたが、その内容が NULL セグメント・セレクタであった場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#NM	CR0 の EM または TS がセットされた場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

FSAVE/FNSAVE - Store x87 FPU State

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
9B DD /6	FSAVE *m94/108byte*	有効	有効	未処理のマスクされていない浮動小数点例外の有無をチェックした後、FPU 状態を *m94byte* または *m108byte* にストアし、次に FPU を初期化する。
DD /6	FNSAVE* *m94/108byte*	有効	有効	未処理のマスクされていない浮動小数点例外の有無をチェックしないで、FPU 状態を *m94byte* または *m108byte* にストアし、次に FPU を初期化する。

影響を受ける FPU フラグ

C0、C1、C2、C3 フラグがセーブされ、次にクリアされる。

IA-32e モードでの操作

レガシーモードと同じ。

浮動小数点例外

なし。

保護モード例外

#GP(0)	デスティネーションが書き込み不可能なセグメントにある場合。
	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタを使用してメモリにアクセスしたが、その内容が NULL セグメント・セレクタであった場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

FSCALE - Scale

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
D9 FD	FSCALE	有効	有効	ST(0) を ST(1) でスケーリングする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1	スタック・アンダーフローが発生した場合は 0 にセットされる。 不正確結果例外（#P）が発生した場合は、丸めの方向を示す。0 ← 切り上げなし、1 ← 切り上げ。
C0、C2、C3	未定義。

浮動小数点例外

#IS	スタック・アンダーフローが発生した場合。
#IA	ソース・オペランドが SNaN 値であるか、またはそのフォーマットがサポートされていない場合。
#D	ソース・オペランドがデノーマル値である場合。
#U	結果が小さすぎて、デスティネーション・フォーマットで表現できない場合。
#O	結果が大きすぎて、デスティネーション・フォーマットで表現できない場合。
#P	値がデスティネーション・フォーマットでは正確に表現できない場合。

保護モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

実アドレスモード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

FSIN - Sine

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
D9 FE	FSIN	有効	有効	ST(0) をその正弦で置き換える。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1	スタック・アンダーフローが発生した場合は 0 にセットされる。 不正確結果例外（#P）が発生した場合は、丸めの方向を示す。0 ← 切り上げなし、1 ← 切り上げ。
C2	範囲（-2^{63} < ソース・オペランド< $+2^{63}$）外の場合は 1 にセットされる。範囲内の場合は 0 にクリアされる。
C0、C3	未定義。

浮動小数点例外

#IS	スタック・アンダーフローが発生した場合。
#IA	ソース・オペランドが SNaN 値または ∞ であるか、そのフォーマットがサポートされていない場合。
#D	ソース・オペランドがデノーマル値である場合。
#P	値がデスティネーション・フォーマットでは正確に表現できない場合。
#U	結果が小さすぎて、デスティネーション・フォーマットで表現できない場合。

保護モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

実アドレスモード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

FSINCOS - Sine and Cosine

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
D9 FB	FSINCOS	有効	有効	ST(0) の正弦と余弦を計算し、ST(0) を正弦で置き換え、余弦をレジスタスタックにプッシュする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1　スタック・アンダーフローが発生した場合は 0 にセットされ、スタック・オーバーフローが発生した場合は 1 にセットされる。

不正確結果例外（#P）が発生した場合は、丸めの方向を示す。0 ← 切り上げなし、1 ← 切り上げ。

C2　範囲（-2^{63} ＜ソース・オペランド＜ $+2^{63}$）外の場合は 1 にセットされる。範囲内の場合は 0 にクリアされる。

C0、C3　未定義。

浮動小数点例外

#IS　スタック・アンダーフローまたはスタック・オーバーフローが発生した場合。

#IA　ソース・オペランドが SNaN 値または ∞ であるか、そのフォーマットがサポートされていない場合。

#D　ソース・オペランドがデノーマル値である場合。

#U　結果が小さすぎて、デスティネーション・フォーマットで表現できない場合。

#P　値がデスティネーション・フォーマットでは正確に表現できない場合。

保護モード例外

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

実アドレスモード例外

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

FSQRT - Square Root

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
D9 FA	FSQRT	有効	有効	ST(0) の平方根を計算し、結果を ST(0) にストアする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1	スタック・アンダーフローが発生した場合は 0 にセットされる。
	不正確結果例外（#P）が発生した場合は、丸めの方向を示す。0 ← 切り上げなし、1 ← 切り上げ。
C0、C2、C3	未定義。

浮動小数点例外

#IS	スタック・アンダーフローが発生した場合。
#IA	ソース・オペランドが SNaN 値であるか、またはそのフォーマットがサポートされていない場合。
	ソース・オペランドが負の値（ただし、–0 は除く）である場合。
#D	ソース・オペランドがデノーマル値である場合。
#P	値がデスティネーション・フォーマットでは正確に表現できない場合。

保護モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

実アドレスモード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

FST/FSTP - Store Floating Point Value

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
D9 /2	FST *m32fp*	有効	有効	ST(0) を *m32fp* にコピーする。
DD /2	FST *m64fp*	有効	有効	ST(0) を *m64fp* にコピーする。
DD D0+i	FST ST(i)	有効	有効	ST(0) を ST(i) にコピーする。
D9 /3	FSTP *m32fp*	有効	有効	ST(0) を *m32fp* にコピーし、レジスタスタックをポップする。
DD /3	FSTP *m64fp*	有効	有効	ST(0) を *m64fp* にコピーし、レジスタスタックをポップする。
DB /7	FSTP *m80fp*	有効	有効	ST(0) を *m80fp* にコピーし、レジスタスタックをポップする。
DD D8+i	FSTP ST(i)	有効	有効	ST(0) を ST(i) にコピーし、レジスタスタックをポップする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1　スタック・アンダーフローが発生した場合は 0 にセットされる。

浮動小数点不正確結果例外（#P）が発生した場合は、丸めの方向を示す。0 ← 切り上げなし、1 ← 切り上げ。

C0、C2、C3　未定義。

浮動小数点例外

#IS　スタック・アンダーフローが発生した場合。

#IA　ソース・オペランドが SNaN 値であるか、またはそのフォーマットがサポートされていない場合。デスティネーション・オペランドが拡張倍精度浮動小数点形式の場合は発生しない。

#U　結果が小さすぎて、デスティネーション・フォーマットで表現できない場合。デスティネーション・オペランドが拡張倍精度浮動小数点形式の場合は発生しない。

#O　結果が大きすぎて、デスティネーション・フォーマットで表現できない場合。デスティネーション・オペランドが拡張倍精度浮動小数点形式の場合は発生しない。

#P　値がデスティネーション・フォーマットでは正確に表現できない場合。デスティネーション・オペランドが拡張倍精度浮動小数点形式の場合は発生しない。

保護モード例外

#GP(0)	デスティネーションが書き込み不可能なセグメントにある場合。
	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタを使用してメモリにアクセスしたが、その内容が NULL セグメント・セレクタであった場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

FSTCW/FNSTCW - Store x87 FPU Control Word

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
9B D9 /7	FSTCW *m2byte*	有効	有効	未処理のマスクされていない浮動小数点定例外の有無をチェックした後、FPU 制御ワードを *m2byte* にストアする。
D9 /7	FNSTCW* *m2byte*	有効	有効	未処理のマスクされていない浮動小数点定例外の有無をチェックしないで、FPU 制御ワードを *m2byte* にストアする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C0、C1、C2、C3 フラグは未定義。

浮動小数点例外

なし。

保護モード例外

#GP(0)	デスティネーションが書き込み不可能なセグメントにある場合。
	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタを使用してメモリにアクセスしたが、その内容が NULL セグメント・セレクタであった場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

FSTENV/FNSTENV - Store x87 FPU Environment

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
9B D9 /6	FSTENV *m14/28byte*	有効	有効	未処理のマスクされていない浮動小数点例外の有無をチェックした後、FPU 環境を *m14byte* または *m28byte* にストアする。次に、すべての浮動小数点例外をマスクする。
D9 /6	FNSTENV* *m14/28byte*	有効	有効	未処理のマスクされていない浮動小数点例外の有無をチェックしないで、FPU 環境を *m14byte* または *m28byte* にストアする。次に、すべての浮動小数点例外をマスクする。

影響を受ける FPU フラグ

C0、C1、C2、C3 フラグは未定義。

浮動小数点例外

なし。

IA-32e モードでの操作

レガシーモードと同じ。

保護モード例外

#GP(0)	デスティネーションが書き込み不可能なセグメントにある場合。
	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタを使用してメモリにアクセスしたが、その内容が NULL セグメント・セレクタであった場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

FSTSW/FNSTSW - Store x87 FPU Status Word

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
9B DD /7	FSTSW *m2byte*	有効	有効	未処理のマスクされていない浮動小数点例外の有無をチェックした後、FPU ステータス・ワードを *m2byte* にストアする。
9B DF E0	FSTSW AX	有効	有効	未処理のマスクされていない浮動小数点例外の有無をチェックした後、FPU ステータス・ワードを AX レジスタにストアする。
DD /7	FNSTSW* *m2byte*	有効	有効	未処理のマスクされていない浮動小数点例外の有無をチェックしないで、FPU ステータス・ワードを *m2byte* にストアする。
DF E0	FNSTSW* AX	有効	有効	未処理のマスクされていない浮動小数点例外の有無をチェックしないで、FPU ステータス・ワードを AX レジスタにストアする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C0、C1、C2、C3 フラグは未定義。

浮動小数点例外

なし。

保護モード例外

#GP(0)	デスティネーションが書き込み不可能なセグメントにある場合。
	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタを使用してメモリにアクセスしたが、その内容が NULL セグメント・セレクタであった場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

FSUB/FSUBP/FISUB - Subtract

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
D8 /4	FSUB *m32fp*	有効	有効	ST(0) から *m32fp* を引き、結果を ST(0) にストアする。
DC /4	FSUB *m64fp*	有効	有効	ST(0) から *m64fp* を引き、結果を ST(0) にストアする。
D8 E0+i	FSUB ST(0), ST(i)	有効	有効	ST(0) から ST(i) を引き、結果を ST(0) にストアする。
DC E8+i	FSUB ST(i), ST(0)	有効	有効	ST(i) から ST(0) を引き、結果を ST(i) にストアする。
DE E8+i	FSUBP ST(i), ST(0)	有効	有効	ST(i) から ST(0) を引き、結果を ST(i) にストアし、レジスタスタックをポップする。
DE E9	FSUBP	有効	有効	ST(1) から ST(0) を引き、結果を ST(1) にストアし、レジスタスタックをポップする。
DA /4	FISUB *m32int*	有効	有効	ST(0) から *m32int* を引き、結果を ST(0) にストアする。
DE /4	FISUB *m16int*	有効	有効	ST(0) から *m16int* を引き、結果を ST(0) にストアする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1	スタック・アンダーフローが発生した場合は 0 にセットされる。
	不正確結果例外（#P）フォルトが発生した場合は、丸めの方向を示す。0 ← 切り上げなし、1 ← 切り上げ。
C0、C2、C3	未定義。

浮動小数点例外

#IS	スタック・アンダーフローが発生した場合。
#IA	オペランドが SNaN 値であるか、またはそのフォーマットがサポートされていない場合。
	両方のオペランドの符号が同じで、絶対値が無限大の場合。
#D	ソース・オペランドがデノーマル値である場合。
#U	結果が小さすぎて、デスティネーション・フォーマットで表現できない場合。
#O	結果が大きすぎて、デスティネーション・フォーマットで表現できない場合。
#P	値がデスティネーション・フォーマットでは正確に表現できない場合。

保護モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタを使用してメモリにアクセスしたが、その内容が NULL セグメント・セレクタであった場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

FSUBR/FSUBRP/FISUBR - Reverse Subtract

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
D8 /5	FSUBR *m32fp*	有効	有効	*m32fp* から ST(0) を引き、結果を ST(0) にストアする。
DC /5	FSUBR *m64fp*	有効	有効	*m64fp* から ST(0) を引き、結果を ST(0) にストアする。
D8 E8+i	FSUBR ST(0), ST(i)	有効	有効	ST(i) から ST(0) を引き、結果を ST(0) にストアする。
DC E0+i	FSUBR ST(i), ST(0)	有効	有効	ST(0) から ST(i) を引き、結果を ST(i) にストアする。
DE E0+i	FSUBRP ST(i), ST(0)	有効	有効	ST(0) から ST(i) を引き、結果を ST(i) にストアし、レジスタスタックをポップする。
DE E1	FSUBRP	有効	有効	ST(0) から ST(1) を引き、結果を ST(1) にストアし、レジスタスタックをポップする。
DA /5	FISUBR *m32int*	有効	有効	*m32int* から ST(0) を引き、結果を ST(0) にストアする。
DE /5	FISUBR *m16int*	有効	有効	*m16int* から ST(0) を引き、結果を ST(0) にストアする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1	スタック・アンダーフローが発生した場合は 0 にセットされる。
	不正確結果例外（#P）フォルトが発生した場合は、丸めの方向を示す。0 ← 切り上げなし、1 ← 切り上げ。
C0、C2、C3	未定義。

浮動小数点例外

#IS	スタック・アンダーフローが発生した場合。
#IA	オペランドが SNaN 値であるか、またはそのフォーマットがサポートされていない場合。
	両方のオペランドの符号が同じで、絶対値が無限大の場合。
#D	ソース・オペランドがデノーマル値である場合。
#U	結果が小さすぎて、デスティネーション・フォーマットで表現できない場合。
#O	結果が大きすぎて、デスティネーション・フォーマットで表現できない場合。
#P	値がデスティネーション・フォーマットでは正確に表現できない場合。

保護モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタを使用してメモリにアクセスしたが、その内容が NULL セグメント・セレクタであった場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

FTST - TEST

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
D9 E4	FTST	有効	有効	ST(0) を 0.0 と比較する。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1　スタック・アンダーフローが発生した場合は 0 にセットされ、発生しなかった場合は 0 にクリアされる。

C0、C2、C3　以下の表を参照。

比較結果	C3	C2	C0
ST(0)>0.0	0	0	0
ST(0)<0.0	0	0	1
ST(0) ← 0.0	1	0	0
順序化不可能	1	1	1

浮動小数点例外

#IS　スタック・アンダーフローが発生した場合。

#IA　ソース・オペランドが NaN 値であるか、またはそのフォーマットがサポートされていない場合。

#D　ソース・オペランドがデノーマル値である場合。

保護モード例外

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

実アドレスモード例外

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

FUCOM/FUCOMP/FUCOMPP - Unordered Compare Floating Point Values

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
DD E0+i	FUCOM ST(i)	有効	有効	ST(0) を ST(i) と比較する。
DD E1	FUCOM	有効	有効	ST(0) を ST(1) と比較する。
DD E8+i	FUCOMP ST(i)	有効	有効	ST(0) を ST(i) と比較し、レジスタスタックをポップする。
DD E9	FUCOMP	有効	有効	ST(0) を ST(1) と比較し、レジスタスタックをポップする。
DA E9	FUCOMPP	有効	有効	ST(0) を ST(1) と比較し、レジスタスタックを 2 回ポップする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1　スタック・アンダーフローが発生した場合は 0 にセットされる。

C0、C2、C3　前のページの表を参照。

浮動小数点例外

#IS　スタック・アンダーフローが発生した場合。

#IA　一方または両方のオペランドが SNaN 値であるか、またはそれらのフォーマットがサポートされていない場合。QNaN 値が検出されても、無効オペランド例外は発生しない。

#D　一方または両方のオペランドがデノーマル値である場合。

保護モード例外

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

実アドレスモード例外

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

FWAIT - Wait

「WAIT/FWAIT - Wait」を参照のこと。

FXAM - Examine

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
D9 E5	FXAM	有効	有効	ST(0) の値または数値をクラスに分類する。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1　　ST(0) の値の符号に設定される。

C0、C2、C3　　下表を参照。

クラス	C3	C2	C0
サポートされていない	0	0	0
NaN	0	0	1
通常の有限値	0	1	0
無限大	0	1	1
ゼロ	1	0	0
空	1	0	1
デノーマル数	1	1	0

浮動小数点例外

なし。

保護モード例外

#NM　　CR0 の EM または TS がセットされた場合。

#MF　　未処理の x87 FPU 例外がある場合。

実アドレスモード例外

#NM　　CR0 の EM または TS がセットされた場合。

#MF　　未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#NM　　CR0 の EM または TS がセットされた場合。

#MF　　未処理の x87 FPU 例外がある場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

FXCH - Exchange Register Contents

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
D9 C8+i	FXCH ST(i)	有効	有効	ST(0) と ST(i) の内容を入れ換える。
D9 C9	FXCH	有効	有効	ST(0) と ST(1) の内容を入れ換える。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1　　0 にクリアされる。

C0、C2、C3　　未定義。

浮動小数点例外

#IS　　スタック・アンダーフローが発生した場合。

保護モード例外

#NM　　CR0 の EM または TS がセットされた場合。

#MF　　未処理の x87 FPU 例外がある場合。

実アドレスモード例外

#NM　　CR0 の EM または TS がセットされた場合。

#MF　　未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#NM　　CR0 の EM または TS がセットされた場合。

#MF　　未処理の x87 FPU 例外がある場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

FXRSTOR - Restore x87 FPU, MMX, SSE, and SSE2 State

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
0F AE /1	FXRSTOR *m512byte*	有効	有効	x87 FPU レジスタ状態、MMX レジスタ状態、XMM レジスタ状態、MXCSR レジスタ状態を *m512byte* からリストアする。

x87 FPU および SIMD 浮動小数点例外

なし。

IA-32e モードでの操作

IA-32eモードでのセーブ/リストア・フォーマットについては、FXSAVEを参照のこと。

保護モード例外

#GP(0)	CS、DS、ES、FS、またはGSセグメント内のメモリ・オペランドの実効アドレスが無効の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが16バイトに合っていない場合（下記のアライメント・チェック例外 [#AC] を参照のこと）。
	MXCSRの予約されているビットをセットしようとした場合。
#SS(0)	SSセグメント内のアドレスが無効の場合。
#MF	未処理のx87 FPU例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0のTSがセットされた場合。
#UD	CR0のEMがセットされた場合。
	CPUID機能フラグFXSRが0の場合。
	命令の前にLOCKプリフィックスが置かれている場合。
#AC	この例外がディスエーブルにされている場合、前述したようにメモリ・オペランドが16バイト境界上にアライメントされていなければ、一般保護例外（#GP）が報告される。アライメント・チェック例外（#AC）がイネーブルになっている場合（およびCPLが3の場合）、#ACは必ずしも報告されるとは限らない。これは、以下に示すように、プロセッサによって異なる。#ACが報告されないすべてのプロセッサでは、#GPが所定の場所で報告される。また、アライメント・チェックの幅も、プロセッサによって異なる。例えば、あるプロセッサでは、2バイトのミスアライメントに対して#ACが報告され、2バイト以外のすべてのミスアライメント（4、8、16バイトのミスアライメント）に対しては#GPが報告される。

実アドレスモード例外

#GP(0)	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。
#UD	CR0 の EM がセットされた場合。
	CPUID 機能フラグ FXSR が 0 の場合。
	命令の前に LOCK オーバーライド・プリフィックスが置かれている場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。
#AC	現行特権レベルが 3 のときに、アライメントの合っていないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#UD	CR0 の EM がセットされた場合。
	CPUID 機能フラグ FXSR が 0 の場合。
	命令の前に LOCK プリフィックスが置かれている場合。
#AC	この例外がディスエーブルにされている場合、前述したようにメモリ・オペランドが 16 バイト境界上にアライメントされていなければ、一般保護例外（#GP）が報告される。アライメント・チェック例外（#AC）がイネーブルになっている場合（および CPL が 3 の場合）、#AC は必ずしも報告されるとは限らない。これは、以下に示すように、プロセッサによって異なる。#AC が報告されないすべてのプロセッサでは、#GP が所定の場所で報告される。また、アライメント・チェックの幅も、プロセッサによって異なる。例えば、あるプロセッサでは、2 バイトのミスアライメントに対して #AC が報告され、2 バイト以外のすべてのミスアライメント（4、8、16 バイトのミスアライメント）に対しては #GP が報告される。

FXSAVE - Save x87 FPU, MMX, SSE, and SSE2 State

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
0F AE /0	FXSAVE *m512byte*	有効	有効	x87 FPU レジスタ状態、MMX® テクノロジ・レジスタ状態、XMM レジスタ状態、MXCSR レジスタ状態を *m512byte* に保存する。

説明

x87 FPU、MMXテクノロジ、XMM、MXCSRの各レジスタの現在の状態を、デスティネーション・オペランドで指定された512バイトのメモリ・ロケーションにセーブする。表2-12.は、レガシーモードにおけるメモリ内の状態情報のレイアウトを示している。

表 2-12. レガシー FXSAVE マップのレイアウト

15 14	13 12	11 10	9 8	7 6	5	4	3 2	1 0	
予約	CS	FPU IP		FOP		FTW	FSW	FCW	0
MXCSR_MASK		MXCSR		予約	DS		FPU DP		16
予約			ST0/MM0						32
予約			ST1/MM1						48
予約			ST2/MM2						64
予約			ST3/MM3						80
予約			ST4/MM4						96
予約			ST5/MM5						112
予約			ST6/MM6						128
予約			ST7/MM7						144
XMM0									160
XMM1									176
XMM2									192
XMM3									208
XMM4									224
XMM5									240
XMM6									256
XMM7									272
予約									288
予約									304
予約									320
予約									336
予約									352

表 2-12. レガシー FXSAVE マップのレイアウト（続き）

15 14	13 12	11 10	9 8	7 6	5	4	3 2	1 0	
予約									368
予約									384
予約									400
予約									416
予約									432
予約									448
予約									464
予約									480
予約									496

デスティネーション・オペランドには、メモリイメージの最初のバイトが格納される。デスティネーション・オペランドは、16バイト境界上にアライメントされていなければならない。デスティネーション・オペランドのアライメントが合っていないと、一般保護 (#GP) 例外（場合によっては、アライメント・チェック例外 [#AC]）が発生する。

FXSAVE 命令が使用されるのは、オペレーティング・システムがコンテキスト・スイッチを実行する必要がある場合、または、例外ハンドラが、x87 FPU、MMX、XMM、MXCSR の各レジスタの現在の状態をセーブおよび検査する必要がある場合である。表 2-12. の各フィールドを以下に説明する。

FCW　x87 FPU 制御ワード（16 ビット）。x87 FPU 制御ワードのレイアウトについては、『IA-32 インテル® アーキテクチャ・ソフトウェア・デベロッパーズ・マニュアル、上巻』の図 8-6. を参照のこと。

FSW　x87 FPU ステータス・ワード（16 ビット）。x87 FPU ステータス・ワードのレイアウトについては、『IA-32 インテル® アーキテクチャ・ソフトウェア・デベロッパーズ・マニュアル、上巻』の図 8-4. を参照のこと。

FTW　x87 FPU タグワード（8 ビット）。ここにセーブされるタグ情報は短縮形式である。詳しくは、以降の段落で説明する。x87 FPU タグワードのレイアウトについては、『IA-32 インテル® アーキテクチャ・ソフトウェア・デベロッパーズ・マニュアル、上巻』の図 8-7. を参照のこと。

FOP　x87 FPU オペコード（16 ビット）。このフィールドの下位 11 ビットにはオペコードが入っており、上位 5 ビットは予約されている。x87 FPU オペコード・フィールドのレイアウトについては、『IA-32 インテル® アーキテクチャ・ソフトウェア・デベロッパーズ・マニュアル、上巻』の図 8-8. を参照のこと。

FPU IP

x87 FPU 命令ポインタ・オフセット（32 ビット）。このフィールドの内容は、FXSAVE 命令の実行時にプロセッサのアドレス指定モードが 32 ビットまたは 16 ビットのどちらであったかによって異なる。

- 32 ビットモード - 32 ビットの IP オフセット。
- 16ビットモード - 下位16ビットはIPオフセット。上位16ビットは予約済み。

x87 FPU 命令ポインタの説明については、『IA-32 インテル® アーキテクチャ・ソフトウェア・デベロッパーズ・マニュアル、上巻』の第 8 章の「x87 FPU 命令とオペランド（データ）ポインタ」を参照のこと。

CS

x87 FPU 命令ポインタセレクタ（16 ビット）。

FPU DP

x87 FPU 命令オペランド（データ）ポインタ・オフセット（32 ビット）。このフィールドの内容は、FXSAVE 命令の実行時にプロセッサのアドレス指定モードが 32 ビットまたは 16 ビットのどちらであったかによって異なる。

- 32 ビットモード - 32 ビットの IP オフセット。
- 16ビットモード - 下位16ビットはIPオフセット。上位16ビットは予約済み。

x87 FPU オペランド・ポインタの説明については、『IA-32 インテル® アーキテクチャ・ソフトウェア・デベロッパーズ・マニュアル、上巻』の第 8 章の「x87 FPU 命令とオペランド（データ）ポインタ」を参照のこと。

DS

x87 FPU 命令オペランド（データ）ポインタ・セレクタ（16 ビット）。

MXCSR

MXCSR レジスタ状態（32 ビット）。MXCSR レジスタのレイアウトについては、『IA-32 インテル® アーキテクチャ・ソフトウェア・デベロッパーズ・マニュアル、上巻』の図 10-3. を参照のこと。制御レジスタ CR4 の OSFXSR ビットが設定されていない場合、FXSAVE 命令を実行しても、このレジスタの値がセーブされない場合がある。この動作は、プロセッサによって異なる。

MXCSR_MASK

MXCSR_MASK（32 ビット）。この値を使用して、すべての予約ビットが 0 に設定されるように、MXCSR レジスタへ書き込む値を調整できる。予約ビットを 0 に設定すると、FXRSTOR 命令または LDMXCSR 命令によって MXCSR レジスタに書き込む際に、一般保護例外（#GP）の発生を防止できる。MXCSR_MASK 値の決定や使用の方法については、『IA-32 インテル® アーキテクチャ・ソフトウェア・デベロッパーズ・マニュアル、上巻』の第 11 章の「MXCSR レジスタへの書き込みのガイドライン」を参照のこと。

ST0/MM0 ～ ST7/MM7	x87 FPU/MMX レジスタ。これらの 80 ビットのフィールドには、FXSAVE 命令を実行する前のプロセッサの状態に応じて、x87 FPU データレジスタまたは MMX レジスタの値が入る。FXSAVE 命令の前に x87 FPU 命令が実行されていた場合は、x87 FPU データレジスタがセーブされる。MMX 命令（または MMX レジスタ上で動作していた SSE、SSE2 命令）が実行されていた場合は、MMX レジスタの値がセーブされる。MMX レジスタの値がセーブされると、このフィールドの上位 16 ビットは予約される。
XMM0 ～ XMM7	XMM レジスタ（1 フィールド当たり 128 ビット）。制御レジスタ CR4 の OSFXSR ビットを設定しないと、FXSAVE 命令を実行しても、これらのレジスタの値がセーブされない場合がある。この動作は、プロセッサによって異なる。

FXSAVE 命令は、x87 FPU タグワードの短縮形を FTW フィールドにセーブする（タグワードを完全な形でセーブする FSAVE 命令とは異なる）。タグ情報は、トップオブスタック（TOS）の順序ではなく、物理的なレジスタの順序（R0 から R7）でセーブされる。ただし、FXSAVE 命令では、タグごとに単一のビット（1 は有効、0 は空）しかセーブされない。例えば、現在、タグワードが次のように設定されていると仮定する。

R7	R6	R5	R4	R3	R2	R1	R0
11	xx	xx	xx	11	11	11	11

ここで、11B は空のスタック要素を表し、「xx」については、00B が有効、01B がゼロ、10B が特殊を表す。

この例の場合、FXSAVE 命令は、次に示すような 8 ビットの情報しかセーブしない。

R7	R6	R5	R4	R3	R2	R1	R0
0	1	1	1	0	0	0	0

1 は有効、ゼロ、特殊のタグについてセーブされ、0 は空のタグについてセーブされる。

FXSAVE 命令の動作は、以下の点で、FSAVE 命令とは異なっている。

- FXSAVE 命令は、未処理のマスクされていない浮動小数点例外のチェックを行わない（この点については、FXSAVE 命令と FNSAVE 命令の動作は同じである）。
- x87 FPU、MMX、XMM、MXCSR の各レジスタの状態が FXSAVE 命令によってセーブされると、プロセッサは、これらのレジスタの内容を保持する。この動作のために、アプリケーション・プログラムにおいて FXSAVE 命令を使って「きれいにした」x87 FPU 状態をプロシージャに渡すことはできない。FXSAVE 命令では、現在の状態が保持されるためである。x87 FPU 状態をきれいにするには、アプリケーションにおいて FXSAVE 命令の後に FINIT 命令を明示的に実行し、x87 FPU 状態を再度初期化する必要がある。

- FXSAVE 命令でセーブされるメモリイメージのフォーマットは、現在のアドレス指定モード（32 ビットまたは 16 ビット）や、動作モード（保護、実アドレス、またはシステム管理）にかかわらず、一定である。この点は、アドレス指定モードや動作モードによってメモリイメージのフォーマットが変わる FSAVE 命令とは異なっている。FXSAVE 命令でセーブされたメモリイメージは、イメージ・フォーマットが異なるため、FRSTOR 命令で正しく復元することはできない。また、これと同様に、FSAVE 命令でセーブされた状態は FXRSTOR 命令で正しく復元することはできない。

FSAVE 命令フォーマットの FTW については、以下の表を使用することにより、FTW の有効ビットと、ストアされている 80 ビットの FP データ（ストアされているデータが MMX レジスタの内容ではない場合）から再作成できる。

表 2-13. 状態セーブマップ

指数 すべて 1	指数 すべて 0	小数 すべて 0	J ビットおよび M ビット	FTW 有効ビット	x87 FTW	
0	0	0	0x	1	特殊	10
0	0	0	1x	1	有効	00
0	0	1	00	1	特殊	10
0	0	1	10	1	有効	00
0	1	0	0x	1	特殊	10
0	1	0	1x	1	特殊	10
0	1	1	00	1	ゼロ	01
0	1	1	10	1	特殊	10
1	0	0	1x	1	特殊	10
1	0	0	1x	1	特殊	10
1	0	1	00	1	特殊	10
1	0	1	10	1	特殊	10
上記のすべての有効な組み合わせ				0	空	11

J ビットは、仮数の小数点の左側のビットの 2 進整数として定義される。M ビットは、仮数の小数点以下の部分の最上位ビット（すなわち、小数点のすぐ右側のビット）として定義される。

M ビットが仮数の小数点以下の部分の最上位ビットのとき、その小数がすべて 0 の場合は、M ビットは 0 でなければならない。

操作

```
DEST ← Save(x87 FPU, MMX, XXM7-XMM0, MXCSR);
```

IA-32e モードでの操作

IA-32e モードには 2 つの fxsave/fxrstor フォーマットがあり、プロセッサが互換モードであるか 64 ビットモードであるかによって異なる。64 ビットモードでは、XMM0 から XMM15 までの SSE レジスタがすべてセーブされる。互換モードでは、XMM0 から XMM7 までのレガシー SSE レジスタが、レガシー FXSAVE マップに基づいてセーブされる。64 ビットモードの場合、64 ビット FXSAVE マップには、REX.W ビットの値に応じて 2 種類のレイアウトが存在する。この 2 種類の違いは、FPU IP ポインタと FPU DP ポインタである。REX.W = 0 の場合、FPU IP は 32 ビットの IP を持つ CS としてセーブされ、FPU DP は 32 ビットの DP を持つ DS としてセーブされる。REX.W = 1 の場合、FPU IP と FPU DP は両方とも、セグメント・セレクタを持たない 64 ビット値である。IA-32e モードのセーブ・フォーマットを以下の表 2-14. と表 2-15. に示す。

表 2-14. REX.W がセットされた場合の 64 ビット FXSAVE マップのレイアウト

15 14	13 12	11 10	9 8	7 6	5	4	3 2	1 0	
FPU IP				FOP		FTW	FSW	FCW	0
MXCSR_MASK		MXCSR		FPU DP					16
予約			ST0/MM0						32
予約			ST1/MM1						48
予約			ST2/MM2						64
予約			ST3/MM3						80
予約			ST4/MM4						96
予約			ST5/MM5						112
予約			ST6/MM6						128
予約			ST7/MM7						144
XMM0									160
XMM1									176
XMM2									192
XMM3									208
XMM4									224
XMM5									240
XMM6									256
XMM7									272
XMM8									288
XMM9									304
XMM10									320
XMM11									336
XMM12									352
XMM13									368
XMM14									384
XMM15									400
予約									416
予約									432
予約									448
予約									464
予約									480
予約									496

表 2-15. REX.W がクリアされた場合の 64 ビット FXSAVE マップのレイアウト

15 14	13 12	11 10	9 8	7 6	5	4	3 2	1 0	
予約	CS	FPU IP		FOP		FTW	FSW	FCW	0
MXCSR_MASK		MXCSR		予約	DS		FPU DP		16
予約			ST0/MM0						32
予約			ST1/MM1						48
予約			ST2/MM2						64
予約			ST3/MM3						80
予約			ST4/MM4						96
予約			ST5/MM5						112
予約			ST6/MM6						128
予約			ST7/MM7						144
XMM0									160
XMM1									176
XMM2									192
XMM3									208
XMM4									224
XMM5									240
XMM6									256
XMM7									272
XMM8									288
XMM9									304
XMM10									320
XMM11									336
XMM12									352
XMM13									368
XMM14									384
XMM15									400
予約									416
予約									432
予約									448
予約									464
予約									480
予約									496

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合（下記のアライメント・チェック例外 [#AC] の説明を参照のこと）。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#UD	CR0 の EM がセットされた場合。
	CPUID 機能フラグ FXSR が 0 の場合。
	命令の前に LOCK オーバーライド・プリフィックスが置かれている場合。
#AC	この例外がディスエーブルにされている場合、前述したようにメモリ・オペランドが 16 バイト境界上にアライメントされていなければ、一般保護例外（#GP）が報告される。アライメント・チェック例外（#AC）がイネーブルになっている場合（および CPL が 3 の場合）、#AC は必ずしも報告されるとは限らない。これは、以下に示すように、プロセッサによって異なる。#AC が報告されないすべてのプロセッサでは、#GP が所定の場所で報告される。また、アライメント・チェックの幅も、プロセッサによって異なる。例えば、あるプロセッサでは、2 バイトのミスアライメントに対して #AC が報告され、2 バイト以外のすべてのミスアライメント（4、8、16 バイトのミスアライメント）に対しては #GP が報告される。

実アドレスモード例外

#GP(0)	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#MF	未処理の x87 FPU 例外がある場合。
#NM	CR0 の TS がセットされた場合。
#UD	CR0 の EM がセットされた場合。
	CPUID 機能フラグ FXSR が 0 の場合。
	命令の前に LOCK オーバーライド・プリフィックスが置かれている場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。

#AC	現行特権レベルが 3 のときに、アライメントの合っていないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#MF	未処理の x87 FPU 例外がある場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#UD	CR0 の EM がセットされた場合。
	CPUID 機能フラグ FXSR が 0 の場合。
	命令の前に LOCK プリフィックスが置かれている場合。
#AC	この例外がディスエーブルにされている場合、前述したようにメモリ・オペランドが 16 バイト境界上にアライメントされていなければ、一般保護例外（#GP）が報告される。アライメント・チェック例外（#AC）がイネーブルになっている場合（および CPL が 3 の場合）、#AC は必ずしも報告されるとは限らない。これは、以下に示すように、プロセッサによって異なる。#AC が報告されないすべてのプロセッサでは、#GP が所定の場所で報告される。また、アライメント・チェックの幅も、プロセッサによって異なる。例えば、あるプロセッサでは、2 バイトのミスアライメントに対して #AC が報告され、2 バイト以外のすべてのミスアライメント（4、8、16 バイトのミスアライメント）に対しては #GP が報告される。

プロセッサに関する注意

命令境界上で一般保護（#GP）例外およびページフォルト（#PF）例外の両方が発生した場合、プロセッサがこれらの例外を報告する順序は、『IA-32 インテル® アーキテクチャ・ソフトウェア・デベロッパーズ・マニュアル、下巻』の表 5-2. に示すとおりである。別の IA-32 プロセッサの FXSAVE 命令では、この順序は異なる。

FXTRACT - Extract Exponent and Significand

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
D9 F4	FXTRACT	有効	有効	ST(0) の値を指数と仮数に分け、指数を ST(0) にストアし、仮数をレジスタスタックにプッシュする。

IA-32e モードでの操作

レガシーモードと同じ。

影響を受ける FPU フラグ

C1	スタック・アンダーフローが発生した場合は 0 にセットされ、スタック・オーバーフローが発生した場合は 1 にセットされる。
C0、C2、C3	未定義。

浮動小数点例外

#IS	スタック・アンダーフローが発生した場合。
	スタック・オーバーフローが発生した場合。
#IA	ソース・オペランドが SNaN 値であるか、またはそのフォーマットがサポートされていない場合。
#Z	ST(0) レジスタのオペランドが ±0。
#D	ソース・オペランドがデノーマル値である場合。

保護モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

実アドレスモード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#NM	CR0 の EM または TS がセットされた場合。
#MF	未処理の x87 FPU 例外がある場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

FYL2X - Compute y ∗ $\log_2 x$

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
D9 F1	FYL2X	有効	有効	ST(1) を (ST(1) ∗ $\log_2$ST(0)) で置き換え、レジスタスタックをポップする。

影響を受ける FPU フラグ

C1　スタック・アンダーフローが発生した場合は 0 にセットされる。

不正確結果例外（#P）が発生した場合は、丸めの方向を示す。0 ← 切り上げなし、1 ← 切り上げ。

C0、C2、C3　未定義。

IA-32e モードでの操作

レガシーモードと同じ。

浮動小数点例外

#IS　スタック・アンダーフローが発生した場合。

#IA　いずれかのオペランドが SNaN であるか、またはそのフォーマットがサポートされていない場合。

ST(0) レジスタのソース・オペランドが負の有限値（–0 は除く）である場合。

#Z　ST(0) レジスタのソース・オペランドが ±0 である場合。

#D　ソース・オペランドがデノーマル値である場合。

#U　結果が小さすぎて、デスティネーション・フォーマットで表現できない場合。

#O　結果が大きすぎて、デスティネーション・フォーマットで表現できない場合。

#P　値がデスティネーション・フォーマットでは正確に表現できない場合。

保護モード例外

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

実アドレスモード例外

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

FYL2XP1 - Compute y ∗ $\log_2$(x +1)

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
D9 F9	FYL2XP1	有効	有効	ST(1) を (ST(1) ∗ $\log_2$(ST(0) + 1.0)) で置き換え、レジスタスタックをポップする。

影響を受ける FPU フラグ

C1　スタック・アンダーフローが発生した場合は 0 にセットされる。

不正確結果例外（#P）が発生した場合は、丸めの方向を示す。0 ← 切り上げなし、1 ← 切り上げ。

C0、C2、C3　未定義。

IA-32e モードでの操作

レガシーモードと同じ。

浮動小数点例外

#IS　スタック・アンダーフローが発生した場合。

#IA　いずれかのオペランドが SNaN であるか、またはそのフォーマットがサポートされていない場合。

#D　ソース・オペランドがデノーマル値である場合。

#U　結果が小さすぎて、デスティネーション・フォーマットで表現できない場合。

#O　結果が大きすぎて、デスティネーション・フォーマットで表現できない場合。

#P　値がデスティネーション・フォーマットでは正確に表現できない場合。

保護モード例外

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

実アドレスモード例外

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

仮想 8086 モード例外

#NM　CR0 の EM または TS がセットされた場合。

#MF　未処理の x87 FPU 例外がある場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

HADDPD - Horizontal Add Packed Double-Precision Floating-Point Values

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
66 0F 7C /r	HADDPD *xmm1, xmm2/m128*	有効	有効	*xmm2/m128* と *xmm1* のパックド倍精度浮動小数点値を水平方向に加算する。

IA-32e モードでの操作

XMM8 ～ XMM15 レジスタへのアクセスが可能である。

SIMD 浮動小数点例外

オーバーフロー、アンダーフロー、無効、精度、デノーマル。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE3 が 0 の場合。

実アドレスモード例外

#GP(0)	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。

#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE3 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE3 が 0 の場合。

HADDPS - Horizontal Add Packed Single-Precision Floating-Point Values

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
F2 0F 7C /r	HADDPS *xmm1, xmm2/m128*	有効	有効	*xmm2/m128* と *xmm1* のパックド単精度浮動小数点値を水平方向に加算する。

IA-32e モードでの操作

XMM8 ～ XMM15 レジスタへのアクセスが可能である。

SIMD 浮動小数点例外

オーバーフロー、アンダーフロー、無効、精度、デノーマル。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE3 が 0 の場合。

実アドレスモード例外

#GP(0)	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。

#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE3 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE3 が 0 の場合。

HLT - Halt

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
F4	HLT	有効	有効	Halt

影響を受けるフラグ

なし。

IA-32e モードでの操作

レガシーモードと同じ。

保護モード例外

#GP(0)　　現行特権レベルが 0 でない場合。

実アドレスモード例外

なし。

仮想 8086 モード例外

#GP(0)　　現行特権レベルが 0 でない場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

HSUBPD - Horizontal Subtract Packed Double-Precision Floating-Point Values

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
66 0F 7D /r	HSUBPD *xmm1, xmm2/m128*	有効	有効	*xmm2/m128* と *xmm1* のパック ド倍精度浮動小数点値を水平方向に減算する。

IA-32e モードでの操作

XMM8 ～ XMM15 レジスタへのアクセスが可能である。

SIMD 浮動小数点例外

オーバーフロー、アンダーフロー、無効、精度、デノーマル。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE3 が 0 の場合。

実アドレスモード例外

#GP(0)	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。

#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE3 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE3 が 0 の場合。

HSUBPS - Horizontal Subtract Packed Single-Precision Floating-Point Values

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
F2 0F 7D /r	HSUBPS *xmm1, xmm2/m128*	有効	有効	*xmm2/m128* と *xmm1* のパック単精度浮動小数点値を水平方向に減算する。

IA-32e モードでの操作

XMM8 ～ XMM15 レジスタへのアクセスが可能である。

SIMD 浮動小数点例外

オーバーフロー、アンダーフロー、無効、精度、デノーマル。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE3 が 0 の場合。

実アドレスモード例外

#GP(0)	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。

#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE3 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
	セグメントに関係なく、メモリ・オペランドのアライメントが 16 バイトに合っていない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#XM	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 1 の場合。
#UD	マスクされていない SIMD 浮動小数点例外が発生し、CR4 の OSXMMEXCPT が 0 の場合。
	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE3 が 0 の場合。

IDIV - Signed Divide

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
F6 /7	IDIV *r/m8*	有効	有効	AX を *r/m8* で符号付き除算する。結果は次のようにストアされる。 AL ← 商、AH ← 剰余
REX + F6 /7	IDIV *r/m8**	有効	N.E.	AX を *r/m8* で符号付き除算する。結果は次のようにストアされる。 AL ← 商、AH ← 剰余
F7 /7	IDIV *r/m16*	有効	有効	DX:AX を *r/m16* で符号付き除算する。結果は次のようにストアされる。 AX ← 商、DX ← 剰余
F7 /7	IDIV *r/m32*	有効	有効	EDX:EAX を *r/m32* で符号付き除算する。結果は次のようにストアされる。EAX ← 商、EDX ← 剰余
REX.W + F7 /7	IDIV *r/m64*	有効	N.E.	RDX:RAX を *r/m64* で符号付き除算する。結果は次のようにストアされる。RAX ← 商、RDX ← 剰余

* 64 ビットモードでは、REX プリフィックスを使用する場合、バイトレジスタ AH、BH、CH、DH にアクセスするように r/m8 をコード化することはできない。1.4.2.2. 節も参照のこと。

影響を受けるフラグ

CF、OF、SF、ZF、AF、PF フラグは未定義。

IA-32e モードでの操作

64 ビットに拡張される。
デフォルトの操作サイズは 32 ビットである。
RAX には 64 ビットの商がストアされ、RDX には 64 ビットの剰余がストアされる。
新しいレジスタ R8 ～ R15 へのアクセスが可能である。

保護モード例外

#DE	ソース・オペランド（除数）が 0 の場合。
	デスティネーションに対して符号付きの結果（商）が大きすぎる場合。
#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタを使用してメモリにアクセスしたが、その内容が NULL セグメント・セレクタであった場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#DE	ソース・オペランド（除数）が 0 の場合。
	デスティネーションに対して符号付きの結果（商）が大きすぎる場合。
#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。

仮想 8086 モード例外

#DE	ソース・オペランド（除数）が 0 の場合。
	デスティネーションに対して符号付きの結果（商）が大きすぎる場合。
#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#DE	ソース・オペランド（除数）が 0 の場合。
	商が大きすぎて、指定されたレジスタにストアできない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

IMUL - Signed Multiply

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
F6 /5	IMUL *r/m8**	有効	有効	AX← AL ∗ *r/m* バイト
F7 /5	IMUL *r/m16*	有効	有効	DX:AX ← AX ∗ *r/m* ワード
F7 /5	IMUL *r/m32*	有効	有効	EDX:EAX ← EAX ∗ *r/m32*
REX.W + F7 /5	IMUL *r/m64*	有効	N.E.	RDX:RAX ← RAX ∗ *r/m64*
0F AF /*r*	IMUL *r16,r/m16*	有効	有効	ワードレジスタ ← ワードレジスタ ∗ *r/m16*
0F AF /*r*	IMUL *r32,r/m32*	有効	有効	ダブルワード・レジスタ ← ダブルワード・レジスタ ∗ *r/m32*
REX.W + 0F AF /*r*	IMUL *r64,r/m64*	有効	N.E.	クワッドワード・レジスタ ← クワッドワード・レジスタ ∗ *r/m* クワッドワード
6B /*r ib*	IMUL *r16,r/m16,imm8*	有効	有効	ワードレジスタ ← *r/m16* ∗ 符号拡張即値バイト
6B /*r ib*	IMUL *r32,r/m32,imm8*	有効	有効	ダブルワード・レジスタ ← *r/m32* ∗ 符号拡張即値バイト
REX.W + 6B /*r ib*	IMUL *r64,r/m64,imm8*	有効	N.E.	クワッドワード・レジスタ ← *r/m64* ∗ 符号拡張即値バイト
6B /*r ib*	IMUL *r16,imm8*	有効	有効	ワードレジスタ ← ワードレジスタ ∗ 符号拡張即値バイト
6B /*r ib*	IMUL *r32,imm8*	有効	有効	ダブルワード・レジスタ ← ダブルワード・レジスタ ∗ 符号拡張即値バイト
REX.W + 6B /*r ib*	IMUL *r64,imm8*	有効	N.E.	クワッドワード・レジスタ ← クワッドワード・レジスタ ∗ 符号拡張即値バイト
69 /*r iw*	IMUL *r16,r/m16,imm16*	有効	有効	ワードレジスタ ← *r/m16* ∗ 即値ワード
69 /*r id*	IMUL *r32,r/m32,imm32*	有効	有効	ダブルワード・レジスタ ← *r/m32* ∗ 即値ダブルワード
REX.W + 69 /*r id*	IMUL *r64,r/m64,imm32*	有効	N.E.	クワッドワード・レジスタ ← *r/m64* ∗ 即値ダブルワード
69 /*r iw*	IMUL *r16,imm16*	有効	有効	ワードレジスタ ← *r/m16* ∗ 即値ワード
69 /*r id*	IMUL *r32,imm32*	有効	有効	ダブルワード・レジスタ ← *r/m32* ∗ 即値ダブルワード
REX.W + 69 /*r id*	IMUL *r64,imm32*	有効	N.E.	クワッドワード・レジスタ ← *r/m64* ∗ 即値ダブルワード

* 64 ビットモードでは、REX プリフィックスを使用する場合、バイトレジスタ AH、BH、CH、DH にアクセスするように r/m8 をコード化することはできない。1.4.2.2. 節も参照のこと。

影響を受けるフラグ

この命令の 1 オペランド形式では、結果の上位半分への有効ビットのキャリーがあるときにCFおよびOFフラグがセットされ、結果が結果の下位半分に収まるときはクリアされる。命令の 2 および 3 オペランド形式では、デスティネーション・オペランド・サイズに合わせるために結果を切り捨てなければならないときはCFおよびOFフラグがセットされ、結

果がデスティネーション・オペランド・サイズに収まるときはクリアされる。SF、ZF、AF、PF フラグは未定義。

IA-32e モードでの操作

64 ビットに拡張される。
デフォルトの操作サイズは 32 ビットである。
新しいレジスタ R8 ～ R15 へのアクセスが可能である。

保護モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタを使用してメモリにアクセスしたが、その内容が NULL セグメント・セレクタであった場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

IN - Input from Port

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
E4 *ib*	IN AL,*imm8*	有効	有効	I/O ポートアドレス *imm8* から AL にバイトを入力する。
E5 *ib*	IN AX,*imm8*	有効	有効	I/O ポートアドレス *imm8* から AX にワードを入力する。
E5 *ib*	IN EAX,*imm8*	有効	有効	I/O ポートアドレス *imm8* から EAX にダブルワードを入力する。
REX.W + E5 *ib*	IN RAX,*imm8*	N.P.	N.E.	REX は後続の命令を変更しない。
EC	IN AL,DX	有効	有効	DX 内の I/O ポートから AL にバイトを入力する。
ED	IN AX,DX	有効	有効	DX 内の I/O ポートから AX にワードを入力する。
ED	IN EAX,DX	有効	有効	DX 内の I/O ポートから EAX にダブルワードを入力する。
REX.W + ED	IN RAX,DX	N.P.	N.E.	REX は後続の命令を変更しない。

影響を受けるフラグ

なし。

IA-32e モードでの操作

レガシーモードと同じ。デフォルトの操作サイズは32 ビットである。

保護モード例外

#GP(0)	CPL が I/O 特権レベル（IOPL）より大きく（低い特権をもつ）、アクセスされる I/O ポートの TSS にある対応する I/O パーミッション・ビットのいずれかが 1 である場合。

実アドレスモード例外

なし。

仮想 8086 モード例外

#GP(0)	アクセスされる I/O ポートの TSS にある対応する I/O パーミッション・ビットのいずれかが 1 である場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#GP(0)	CPL が I/O 特権レベル（IOPL）より大きく（低い特権をもつ）、アクセスされる I/O ポートの TSS にある対応する I/O パーミッション・ビットのいずれかが 1 である場合。

INC - Increment by 1

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
FE /0	INC *r/m8*	有効	有効	*r/m* バイトを 1 インクリメントする。
REX + FE /0	INC *r/m8**	有効	N.E.	*r/m* バイトを 1 インクリメントする。
FF /0	INC *r/m16*	有効	有効	*r/m* ワードを 1 インクリメントする。
FF /0	INC *r/m32*	有効	有効	*r/m* ダブルワードを 1 インクリメントする。
REX.W + FF /0	INC *r/m64*	有効	N.E.	*r/m* クワッドワードを 1 インクリメントする。
40+ *rw*	INC *r16*	N.E.	有効	ワードレジスタを 1 インクリメントする。
40+ *rd*	INC *r32*	N.E.	有効	ダブルワード・レジスタを 1 インクリメントする。

* 64 ビットモードでは、REX プリフィックスを使用する場合、バイトレジスタ AH、BH、CH、DH にアクセスするように r/m8 をコード化することはできない。1.4.2.2. 節も参照のこと。

影響を受けるフラグ

CF フラグは影響を受けない。OF、SF、ZF、AF、PF フラグが結果に従ってセットされる。

IA-32e モードでの操作

64 ビットに拡張される。
デフォルトの操作サイズは 32 ビットである。
64 ビットモードでは、オペコード 40H ～ 47H が REX プリフィックスである。
新しいレジスタ R8 ～ R15 へのアクセスが可能である。

保護モード例外

#GP(0)	デスティネーション・オペランドが書き込み不可能なセグメントにある場合。
	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタを使用してメモリにアクセスしたが、その内容が NULL セグメント・セレクタであった場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

INS/INSB/INSW/INSD - Input from Port to String

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
6C	INS m8, DX	有効	有効	バイトを DX で指定された I/O ポートから ES:(E)DI で指定されたメモリ・ロケーションに入力する。
REX.W + 6C	INS m8, DX	有効	N.E.	バイトを DX で指定された I/O ポートから RDI で指定されたメモリ・ロケーションに入力する。
6D	INS m16, DX	有効	有効	ワードを DX で指定された I/O ポートから ES:(E)DI で指定されたメモリ・ロケーションに入力する。
6D	INS m32, DX	有効	有効	ダブルワードを DX で指定された I/O ポートから ES:(E)DI で指定されたメモリ・ロケーションに入力する。
REX.W + 6D	INS m32, DX	N.P.	N.E.	デフォルト・サイズを DX で指定された I/O ポートから RDI で指定されたメモリ・ロケーションに入力する。
6C	INSB	有効	有効	バイトを DX で指定された I/O ポートから ES:(E)DI で指定されたメモリ・ロケーションに入力する。
REX.W + 6C	INSB	有効	N.E.	バイトを DX で指定された I/O ポートから RDI で指定されたメモリ・ロケーションに入力する。
6D	INSW	有効	有効	ワードを DX で指定された I/O ポートから ES:(E)DI で指定されたメモリ・ロケーションに入力する。
6D	INSD	有効	有効	ダブルワードを DX で指定された I/O ポートから ES:(E)DI で指定されたメモリ・ロケーションに入力する。
REX.W + 6D	INSD	N.P.	N.E.	デフォルト・サイズを DX で指定された I/O ポートから RDI で指定されたメモリ・ロケーションに入力する。

影響を受けるフラグ

なし。

IA-32e モードでの操作

デフォルトのオペランド・サイズは 32 ビットであり、REX.W によって拡張されることはない。
64 ビットモードでは RDI を使用できる。

保護モード例外

#GP(0)	CPL が I/O 特権レベル（IOPL）より大きく（低い特権をもつ）、アクセスされる I/O ポートの TSS にある対応する I/O パーミッション・ビットのいずれかが 1 である場合。
	デスティネーションが書き込み不可能なセグメントにある場合。
	ES セグメントにイリーガルなメモリ・オペランドの実効アドレスが指定された場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。

仮想 8086 モード例外

#GP(0)	アクセスされる I/O ポートの TSS にある対応する I/O パーミッション・ビットのいずれかが 1 である場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	CPL が I/O 特権レベル（IOPL）より大きく（低い特権をもつ）、アクセスされる I/O ポートの TSS にある対応する I/O パーミッション・ビットのいずれかが 1 である場合。
	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

INT *n*/INTO/INT 3 - Call to Interrupt Procedure

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
CC	INT 3	有効	有効	割り込み 3 - デバッガへのトラップ。
CD *ib*	INT *imm8*	有効	有効	割り込みベクタ番号の即値バイトによる指定。
CE	INTO	なし	有効	割り込み 4 - オーバーフロー・フラグが 1 である場合。

影響を受けるフラグ

EFLAGS レジスタがスタックにプッシュされる。INT 命令が実行されるときのプロセッサの動作モードに応じて、IF、TF、NT、AC、RF、VM フラグがクリアされることがある。割り込みがタスクゲートを使用する場合は、新しいタスクの TSS 内の EFLAGS イメージによる制御のもとに、任意のフラグがセットまたはクリアされることがある。

IA-32e モードでの操作

デフォルトの操作サイズは 32 ビットである。

保護モード例外

#GP(0)	IDT、割り込みゲート、トラップゲート、またはタスクゲート内の命令ポインタがコード・セグメントの範囲を超えている場合。
#GP（セレクタ）	割り込みゲート、トラップゲート、またはタスクゲート内のセグメント・セレクタが NULL の場合。
	割り込みゲート、トラップゲート、タスクゲート、コード・セグメント、または TSS のセグメント・セレクタ・インデックスがそのディスクリプタ・テーブルの範囲外の場合。
	割り込みベクタ番号が IDT の範囲外の場合。
	IDT ディスクリプタが、割り込みディスクリプタ、トラップ・ディスクリプタ、またはタスク・ディスクリプタのいずれでもない場合。
	INT *n*、INT 3、または INTO 命令によって割り込みが発生し、割り込みディスクリプタ、トラップ・ディスクリプタ、またはタスク・ディスクリプタの DPL が CPL より小さい場合。
	割り込みゲートまたはトラップゲート内のセグメント・セレクタの指示先がコード・セグメントのセグメント・ディスクリプタでない場合。
	TSS のセグメント・セレクタのローカル / グローバル・ビットがローカルとしてセットされている場合。
	TSS のセグメント・ディスクリプタが、TSS がビジーであるか、または使用不可能であることを示している場合。
#SS(0)	リターンアドレス、フラグ、またはエラーコードをスタックにプッシュして、スタック・セグメントの範囲を超えたが、スタックスイッチが行われなかった場合。

#SS（セレクタ）　SS レジスタがロードされようとしたとき、指示先のセグメントが存在しないとマークされていた場合。

スタックスイッチが発生したとき、リターンアドレス、フラグ、エラーコード、またはスタック・セグメント・ポインタをプッシュして、新しいスタック・セグメントの範囲を超えた場合。

#NP（セレクタ）　コード・セグメント、割り込みゲート、トラップゲート、タスクゲート、または TSS が存在しない場合。

#TS（セレクタ）　TSS 内のスタック・セグメント・セレクタの RPL が、割り込みゲートまたはトラップゲートによってアクセスされるコード・セグメントの DPL に等しくない場合。

TSS 内のスタック・セグメント・セレクタによって指されているスタック・セグメント・ディスクリプタの DPL が割り込みゲートまたはトラップゲートのコード・セグメント・ディスクリプタの DPL に等しくない場合。

TSS 内のスタック・セグメント・セレクタが NULL の場合。

TSS のスタック・セグメントが書き込み可能なデータ・セグメントでない場合。

スタック・セグメントのセグメント・セレクタ・インデックスがディスクリプタ・テーブルの範囲外の場合。

#PF（フォルトコード）　ページフォルトが発生した場合。

実アドレスモード例外

#GP　メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。

割り込みベクタ番号が IDT の範囲外の場合。

#SS　プッシュ時スタック範囲違反の場合。

リターンアドレス、フラグ、またはエラーコードをスタックにプッシュして、スタック・セグメントの範囲を超えた場合。

仮想 8086 モード例外

#GP(0)　（INT *n*、INTO、または BOUND 命令の場合）IOPL が 3 より小さいか、または割り込みゲート、トラップゲート、またはタスクゲートのディスクリプタの DPL が 3 に等しくない場合。

IDT、割り込みゲート、トラップゲート、またはタスクゲート内の命令ポインタがコード・セグメントの範囲を超えている場合。

#GP（セレクタ）	割り込みゲート、トラップゲート、またはタスクゲート内のセグメント・セレクタが NULL の場合。
	割り込みゲート、トラップゲート、タスクゲート、コード・セグメント、または TSS のセグメント・セレクタ・インデックスがそのディスクリプタ・テーブルの範囲外の場合。
	割り込みベクタ番号が IDT の範囲外の場合。
	IDT ディスクリプタが、割り込みディスクリプタ、トラップ・ディスクリプタ、またはタスク・ディスクリプタのいずれでもない場合。
	INT *n* 命令によって割り込みが発生し、割り込みディスクリプタ、トラップ・ディスクリプタ、またはタスク・ディスクリプタの DPL が CPL より小さい場合。
	割り込みゲートまたはトラップゲート内のセグメント・セレクタの指示先がコード・セグメントのセグメント・ディスクリプタでない場合。
	TSS のセグメント・セレクタのローカル / グローバル・ビットがローカルとしてセットされている場合。
#SS（セレクタ）	SS レジスタがロードされようとしたとき、指示先のセグメントが存在しないとマークされていた場合。
	リターンアドレス、フラグ、エラーコード、スタック・セグメント・ポインタ、またはデータ・セグメントをプッシュして、スタック・セグメントの範囲を超えた場合。
#NP（セレクタ）	コード・セグメント、割り込みゲート、トラップゲート、タスクゲート、または TSS が存在しない場合。
#TS（セレクタ）	TSS 内のスタック・セグメント・セレクタの RPL が、割り込みゲートまたはトラップゲートによってアクセスされるコード・セグメントの DPL に等しくない場合。
	TSS のスタック・セグメントに対するスタック・セグメント・ディスクリプタの DPL が、割り込みゲートまたはトラップゲートのコード・セグメント・ディスクリプタの DPL に等しくない場合。
	TSS 内のスタック・セグメント・セレクタが NULL の場合。
	TSS のスタック・セグメントが書き込み可能なデータ・セグメントでない場合。
	スタック・セグメントのセグメント・セレクタ・インデックスがディスクリプタ・テーブルの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#BP	INT 3 命令が実行された場合。
#OF	INTO 命令が実行され、OF フラグがセットされた場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#GP(0)	64 ビットの割り込みゲートまたは 64 ビットのトラップゲート内の命令ポインタが非標準の場合。
#GP（セレクタ）	64 ビットの割り込みゲートまたはトラップゲート内のセグメント・セレクタが NULL の場合。
	割り込みベクタ番号が IDT の範囲外の場合。
	割り込みベクタ番号の指示先が、非標準領域内にあるゲートの場合。
	割り込みベクタ番号の指示先が、64 ビットの割り込みゲートまたは 64 ビットのトラップゲートではないディスクリプタの場合。
	ゲートセレクタの指示先のディスクリプタがディスクリプタ・テーブルの範囲外の場合。
	ゲートセレクタの指示先のディスクリプタが非標準領域内にある場合。
	ゲートセレクタの指示先のディスクリプタがコード・セグメントでない場合。
	ゲートセレクタの指示先のディスクリプタで、L ビットがセットされていないか、L ビットと D ビットの両方がセットされている場合。
	ゲートセレクタの指示先のディスクリプタで、DPL > CPL の場合。
#SS(0)	スタックスイッチが行われなかったときに、古い EFLAGS、CS セレクタ、EIP、またはエラーコードをプッシュした結果、非標準領域内に達した場合。
#SS（セレクタ）	スタックスイッチ（IST による CPL の変更または非 CPL）が行われたときに、古い SS セレクタ、ESP、EFLAGS、CS セレクタ、EIP、またはエラーコードをプッシュした結果、非標準領域内に達した場合。
#NP（セレクタ）	64 ビットの割り込みゲート、64 ビットのトラップゲート、またはコード・セグメントが存在しない場合。
#TS（セレクタ）	TSS から RSP をロードしようとした結果、非標準領域にアクセスした場合。
	TSS からの RSP がディスクリプタ・テーブルの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。

INVD - Invalidate Internal Caches

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
0F 08	INVD	有効	有効	内部キャッシュをフラッシュする。外部キャッシュのフラッシュを開始させる。

影響を受けるフラグ

なし。

IA-32e モードでの操作

レガシーモードと同じ。

保護モード例外

#GP(0)　　現行特権レベルが 0 でない場合。

実アドレスモード例外

なし。

仮想 8086 モード例外

#GP(0)　　INVD 命令は仮想 8086 モードで実行することはできない。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

INVLPG - Invalidate TLB Entry

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
0F 01/7	INVLPG m	有効	有効	*m* があるページの TLB エントリを無効にする。

影響を受けるフラグ

なし。

IA-32e モードでの操作

レガシーモードと同じ。

保護モード例外

#GP(0)	現行特権レベルが 0 でない場合。
#UD	オペランドがレジスタである場合。

実アドレスモード例外

#UD	オペランドがレジスタである場合。

仮想 8086 モード例外

#GP(0)	INVLPG 命令は仮想 8086 モードで実行することはできない。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	現行特権レベルが 0 でない場合。
	メモリアドレスが非標準形式の場合。
#UD	オペランドがレジスタである場合。

IRET/IRETD - Interrupt Return

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
CF	IRET	有効	有効	割り込みリターン（オペランド・サイズ：16 ビット）
CF	IRETD	有効	有効	割り込みリターン（オペランド・サイズ：32 ビット）
REX.W + CF	IRETQ	有効	N.E.	割り込みリターン（オペランド・サイズ：64 ビット）

影響を受けるフラグ

プロセッサの動作モードに応じて、EFLAGS レジスタ内のすべてのフラグおよびフィールドが修正される可能性がある。ネストされたタスクから以前のタスクへのリターンを実行した場合は、EFLAGS レジスタは前のタスクの TSS にストアされていた EFLAGS イメージに従って修正される。

IA-32e モードでの操作

64 ビットに拡張される。
デフォルトの操作サイズは 32 ビットである。

保護モード例外

#GP(0)	リターンコードまたはスタック・セグメント・セレクタが NULL の場合。
	リターン命令ポインタがリターン・コード・セグメントの範囲内にない場合。
#GP（セレクタ）	セグメント・セレクタ・インデックスがディスクリプタ・テーブルの範囲外の場合。
	リターン・コード・セグメント・セレクタの RPL が CPL より大きい場合。
	コンフォーミング・コード・セグメントの DPL がリターン・コード・セグメント・セレクタの RPL より大きい場合。
	非コンフォーミング・コード・セグメントの DPL がコード・セグメント・セレクタの RPL に等しくない場合。
	スタック・セグメント・ディスクリプタの DPL がリターン・コード・セグメント・セレクタの RPL に等しくない場合。
	スタック・セグメントが書き込み可能なデータ・セグメントでない場合。
	スタック・セグメント・セレクタの RPL がリターン・コード・セグメント・セレクタの RPL に等しくない場合。
	コード・セグメントのセグメント・ディスクリプタが、それがコード・セグメントであることを示していない場合。

	TSS のセグメント・セレクタのローカル / グローバル・ビットがローカルとしてセットされている場合。
	TSS のセグメント・ディスクリプタが、TSS がビジーであるか、または使用不可能であることを示している場合。
#SS(0)	スタックのトップバイトがスタックの範囲内にない場合。
#NP（セレクタ）	リターンコードまたはスタック・セグメントが存在しない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	現行特権レベルが 3 のときに、アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照が行われた場合。

実アドレスモード例外

#GP	リターン命令ポインタがリターン・コード・セグメントの範囲内にない場合。
#SS	スタックのトップバイトがスタックの範囲内にない場合。

仮想 8086 モード例外

#GP(0)	リターン命令ポインタがリターン・コード・セグメントの範囲内にない場合。
	IOPL が 3 に等しくない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#SS(0)	スタックのトップバイトがスタックの範囲内にない場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照が行われた場合。
#GP(0)	INVLPG 命令は仮想 8086 モードで実行することはできない。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#GP(0)	EFLAGS.NT ビットがセットされた場合。
	リターン・コード・セグメント・セレクタが NULL の場合。
	スタック・セグメント・セレクタが、互換モードに戻る NULL の場合。
	スタック・セグメント・セレクタが、CPL3 の 64 ビットモードに戻る NULL の場合。
	NULL のスタック・セグメント・セレクタの RPL が、CPL3 以外の 64 ビットモードに戻る CPL に等しくない場合。
	リターン命令ポインタがリターン・コード・セグメントの範囲内にない場合。
	リターン命令ポインタが非標準の場合。

#GP（セレクタ）	セグメント・セレクタ・インデックスがディスクリプタ・テーブルの範囲外の場合。
	セグメント・ディスクリプタのメモリアドレスが非標準の場合。
	コード・セグメントのセグメント・ディスクリプタが、それがコード・セグメントであることを示していない場合。
	指定された新しいコード・セグメント・ディスクリプタで、D ビットと L ビットの両方がセットされている場合。
	非コンフォーミング・コード・セグメントの DPL がコード・セグメント・セレクタの RPL に等しくない場合。
	CPL がコード・セグメント・セレクタの RPL がより大きい場合。
	コンフォーミング・コード・セグメントの DPL がリターン・コード・セグメント・セレクタの RPL より大きい場合。
	スタック・セグメントが書き込み可能なデータ・セグメントでない場合。
	スタック・セグメント・ディスクリプタの DPL がリターン・コード・セグメント・セレクタの RPL に等しくない場合。
	スタック・セグメント・セレクタの RPL がリターン・コード・セグメント・セレクタの RPL に等しくない場合。
#SS(0)	スタックから値をポップしようとした結果、SS の範囲に違反した場合。
	スタックから値をポップしようとした結果、非標準のアドレスが参照された場合。
#NP（セレクタ）	リターンコードまたはスタック・セグメントが存在しない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	現行特権レベルが 3 のときに、アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照が行われた場合。

J*cc* - Jump if Condition Is Met

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
77 *cb*	JA *rel8*	有効	有効	より上（CF=0 および ZF=0）の場合 short ジャンプする。
73 *cb*	JAE *rel8*	有効	有効	より上か等しい（CF=0）場合 short ジャンプする。
72 *cb*	JB *rel8*	有効	有効	より下（CF=1）の場合 short ジャンプする。
76 *cb*	JBE *rel8*	有効	有効	より下か等しい（CF=1 または ZF=1）場合 short ジャンプする。
72 *cb*	JC *rel8*	有効	有効	キャリーがある（CF=1）場合 short ジャンプする。
E3 *cb*	JCXZ *rel8*	有効	有効	CX レジスタが 0 の場合 short ジャンプする。
E3 *cb*	JECXZ *rel8*	有効	有効	ECX レジスタが 0 の場合 short ジャンプする。
74 *cb*	JE *rel8*	有効	有効	等しい（ZF=1）場合 short ジャンプする。
7F *cb*	JG *rel8*	有効	有効	より大きい（ZF=0 および SF=OF）場合 short ジャンプする。
7D *cb*	JGE *rel8*	有効	有効	より大きいか等しい（SF=OF）場合 short ジャンプする。
7C *cb*	JL *rel8*	有効	有効	より小さい（SF<>OF）場合 short ジャンプする。
7E *cb*	JLE *rel8*	有効	有効	より小さいか等しい（ZF=1 または SF<>OF）場合 short ジャンプする。
76 *cb*	JNA *rel8*	有効	有効	より上でない（CF=1 または ZF=1）場合 short ジャンプする。
72 *cb*	JNAE *rel8*	有効	有効	より上でなく等しくない（CF=1）場合 short ジャンプする。
73 *cb*	JNB *rel8*	有効	有効	より下でない（CF=0）場合 short ジャンプする。
77 *cb*	JNBE *rel8*	有効	有効	より下でなく等しくない（CF=0 および ZF=0）場合 short ジャンプする。
73 *cb*	JNC *rel8*	有効	有効	キャリーがない（CF=0）場合 short ジャンプする。
75 *cb*	JNE *rel8*	有効	有効	等しくない（ZF=0）場合 short ジャンプする。
7E *cb*	JNG *rel8*	有効	有効	より大きくない（ZF=1 または SF<>OF）場合 short ジャンプする。
7C *cb*	JNGE *rel8*	有効	有効	より大きくなく等しくない（SF<>OF）場合 short ジャンプする。
7D *cb*	JNL *rel8*	有効	有効	より小さくない（SF=OF）場合 short ジャンプする。
7F *cb*	JNLE *rel8*	有効	有効	より小さくなく等しくない（ZF=0 および SF=OF）場合 short ジャンプする。

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
71 *cb*	JNO *rel8*	有効	有効	オーバーフローがない（OF=0）場合 short ジャンプする。
7B *cb*	JNP *rel8*	有効	有効	パリティがない（PF=0）場合 short ジャンプする。
79 *cb*	JNS *rel8*	有効	有効	符号がない（SF=0）場合 short ジャンプする。
75 *cb*	JNZ *rel8*	有効	有効	ゼロでない（ZF=0）場合 short ジャンプする。
70 *cb*	JO *rel8*	有効	有効	オーバーフローがある（OF=1）場合 short ジャンプする。
7A *cb*	JP *rel8*	有効	有効	パリティがある（PF=1）場合 short ジャンプする。
7A *cb*	JPE *rel8*	有効	有効	パリティが偶数（PF=1）の場合 short ジャンプする。
7B *cb*	JPO *rel8*	有効	有効	パリティが奇数（PF=0）の場合 short ジャンプする。
78 *cb*	JS *rel8*	有効	有効	符号がある（SF=1）場合 short ジャンプする。
74 *cb*	JZ *rel8*	有効	有効	ゼロ（ZF ← 1）の場合 short ジャンプする。
0F 87 *cw*	JA *rel16*	N.S.	有効	より上（CF=0 および ZF=0）の場合 near ジャンプする。64 ビットモードでは未サポート。
0F 87 *cd*	JA *rel32*	有効	有効	より上（CF=0 および ZF=0）の場合 near ジャンプする。
0F 83 *cw*	JAE *rel16*	N.S.	有効	より上か等しい（CF=0）場合 near ジャンプする。64 ビットモードでは未サポート。
0F 83 *cd*	JAE *rel32*	有効	有効	より上か等しい（CF=0）場合 near ジャンプする。
0F 82 *cw*	JB *rel16*	N.S.	有効	より下（CF=1）の場合 near ジャンプする。64 ビットモードでは未サポート。
0F 82 *cd*	JB *rel32*	有効	有効	より下（CF=1）の場合 near ジャンプする。
0F 86 *cw*	JBE *rel16*	N.S.	有効	より下か等しい（CF=1 または ZF=1）場合 near ジャンプする。64 ビットモードでは未サポート。
0F 86 *cd*	JBE *rel32*	有効	有効	より下か等しい（CF=1 または ZF=1）場合 near ジャンプする。
0F 82 *cw*	JC *rel16*	N.S.	有効	キャリーがある（CF=1）場合 near ジャンプする。64 ビットモードでは未サポート。
0F 82 *cd*	JC *rel32*	有効	有効	キャリーがある（CF=1）場合 near ジャンプする。
0F 84 *cw*	JE *rel16*	N.S.	有効	等しい（ZF=1）場合 near ジャンプする。64 ビットモードでは未サポート。

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
0F 84 *cd*	JE *rel32*	有効	有効	等しい（ZF=1）場合 near ジャンプする。
0F 84 *cw*	JZ *rel16*	N.S.	有効	ゼロ（ZF=1）の場合 near ジャンプする。64 ビットモードでは未サポート。
0F 84 *cd*	JZ *rel32*	有効	有効	ゼロ（ZF=1）の場合 near ジャンプする。
0F 8F *cw*	JG *rel16*	N.S.	有効	より大きい（ZF=0 および SF=OF）場合 near ジャンプする。64 ビットモードでは未サポート。
0F 8F *cd*	JG *rel32*	有効	有効	より大きい（ZF=0 および SF=OF）場合 near ジャンプする。
0F 8D *cw*	JGE *rel16*	N.S.	有効	より大きいか等しい（SF=OF）場合 near ジャンプする。64 ビットモードでは未サポート。
0F 8D *cd*	JGE *rel32*	有効	有効	より大きいか等しい（SF=OF）場合 near ジャンプする。
0F 8C *cw*	JL *rel16*	N.S.	有効	より小さい（SF<>OF）場合 near ジャンプする。64 ビットモードでは未サポート。
0F 8C *cd*	JL *rel32*	有効	有効	より小さい（SF<>OF）場合 near ジャンプする。
0F 8E *cw*	JLE *rel16*	N.S.	有効	より小さいか等しい（ZF=1 または SF<>OF）場合 near ジャンプする。64 ビットモードでは未サポート。
0F 8E *cd*	JLE *rel32*	有効	有効	より小さいか等しい（ZF=1 または SF<>OF）場合 near ジャンプする。
0F 86 *cw*	JNA *rel16*	N.S.	有効	より上でない（CF=1 または ZF=1）場合 near ジャンプする。64 ビットモードでは未サポート。
0F 86 *cd*	JNA *rel32*	有効	有効	より上でない（CF=1 または ZF=1）場合 near ジャンプする。
0F 82 *cw*	JNAE *rel16*	N.S.	有効	より上でなく等しくない（CF=1）場合 near ジャンプする。64 ビットモードでは未サポート。
0F 82 *cd*	JNAE *rel32*	有効	有効	より上でなく等しくない（CF=1）場合 near ジャンプする。
0F 83 *cw*	JNB *rel16*	N.S.	有効	より下でない（CF=0）場合 near ジャンプする。64 ビットモードでは未サポート。
0F 83 *cd*	JNB *rel32*	有効	有効	より下でない（CF=0）場合 near ジャンプする。
0F 87 *cw*	JNBE *rel16*	N.S.	有効	より下でなく等しくない（CF=0 および ZF=0）場合 near ジャンプする。64 ビットモードでは未サポート。
0F 87 *cd*	JNBE *rel32*	有効	有効	より下でなく等しくない（CF=0 および ZF=0）場合 near ジャンプする。

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
0F 83 *cw*	JNC *rel16*	N.S.	有効	キャリーがない（CF=0）場合 near ジャンプする。64 ビットモードでは未サポート。
0F 83 *cd*	JNC *rel32*	有効	有効	キャリーがない（CF=0）場合 near ジャンプする。
0F 85 *cw*	JNE *rel16*	N.S.	有効	等しくない（ZF=0）場合 near ジャンプする。64 ビットモードでは未サポート。
0F 85 *cd*	JNE *rel32*	有効	有効	等しくない（ZF=0）場合 near ジャンプする。
0F 8E *cw*	JNG *rel16*	N.S.	有効	より大きくない（ZF=1 または SF<>OF）場合 near ジャンプする。64 ビットモードでは未サポート。
0F 8E *cd*	JNG *rel32*	有効	有効	より大きくない（ZF=1 または SF<>OF）場合 near ジャンプする。
0F 8C *cw*	JNGE *rel16*	N.S.	有効	より大きくなく等しくない（SF<>OF）場合 near ジャンプする。64 ビットモードでは未サポート。
0F 8C *cd*	JNGE *rel32*	有効	有効	より大きくなく等しくない（SF<>OF）場合 near ジャンプする。
0F 8D *cw*	JNL *rel16*	N.S.	有効	より小さくない（SF=OF）場合 near ジャンプする。64 ビットモードでは未サポート。
0F 8D *cd*	JNL *rel32*	有効	有効	より小さくない（SF=OF）場合 near ジャンプする。
0F 8F *cw*	JNLE *rel16*	N.S.	有効	より小さくなく等しくない（ZF=0 および SF=OF）場合 near ジャンプする。64 ビットモードでは未サポート。
0F 8F *cd*	JNLE *rel32*	有効	有効	より小さくなく等しくない（ZF=0 および SF=OF）場合 near ジャンプする。
0F 81 *cw*	JNO *rel16*	N.S.	有効	オーバーフローがない（OF=0）場合 near ジャンプする。64 ビットモードでは未サポート。
0F 81 *cd*	JNO *rel32*	有効	有効	オーバーフローがない（OF=0）場合 near ジャンプする。
0F 8B *cw*	JNP *rel16*	N.S.	有効	パリティがない（PF=0）場合 near ジャンプする。64 ビットモードでは未サポート。
0F 8B *cd*	JNP *rel32*	有効	有効	パリティがない（PF=0）場合 near ジャンプする。
0F 89 *cw*	JNS *rel16*	N.S.	有効	符号がない（SF=0）場合 near ジャンプする。64 ビットモードでは未サポート。
0F 89 *cd*	JNS *rel32*	有効	有効	符号がない（SF=0）場合 near ジャンプする。
0F 85 *cw*	JNZ *rel16*	N.S.	有効	ゼロでない（ZF=0）場合 near ジャンプする。64 ビットモードでは未サポート。
0F 85 *cd*	JNZ *rel32*	有効	有効	ゼロでない（ZF=0）場合 near ジャンプする。

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
0F 80 *cw*	JO *rel16*	N.S.	有効	オーバーフローがある（OF=1）場合 near ジャンプする。64 ビットモードでは未サポート。
0F 80 *cd*	JO *rel32*	有効	有効	オーバーフローがある（OF=1）場合 near ジャンプする。
0F 8A *cw*	JP *rel16*	N.S.	有効	パリティがある（PF=1）場合 near ジャンプする。64 ビットモードでは未サポート。
0F 8A *cd*	JP *rel32*	有効	有効	パリティがある（PF=1）場合 near ジャンプする。
0F 8A *cw*	JPE *rel16*	N.S.	有効	パリティが偶数（PF=1）の場合 near ジャンプする。64 ビットモードでは未サポート。
0F 8A *cd*	JPE *rel32*	有効	有効	パリティが偶数（PF=1）の場合 near ジャンプする。
0F 8B *cw*	JPO *rel16*	N.S.	有効	パリティが奇数（PF=0）の場合 near ジャンプする。64 ビットモードでは未サポート。
0F 8B *cd*	JPO *rel32*	有効	有効	パリティが奇数（PF=0）の場合 near ジャンプする。
0F 88 *cw*	JS *rel16*	N.S.	有効	符号がある（SF=1）場合 near ジャンプする。64 ビットモードでは未サポート。
0F 88 *cd*	JS *rel32*	有効	有効	符号がある（SF=1）場合 near ジャンプする。
0F 84 *cw*	JZ *rel16*	N.S.	有効	ゼロ（ZF=1）の場合 near ジャンプする。64 ビットモードでは未サポート。
0F 84 *cd*	JZ *rel32*	有効	有効	ゼロ（ZF=1）の場合 near ジャンプする。

影響を受けるフラグ

なし。

IA-32e モードでの操作

オペランド・サイズは64 ビットに固定される。
JMP Short では、RIP = RIP + 64 ビットに符号拡張された 8 ビットのオフセット。
JMP Near では、RIP = RIP + 64 ビットに符号拡張された 32 ビットのオフセット。

保護モード例外

#GP(0)　　ジャンプ先のオフセットが CS セグメントの範囲を超えている場合。

実アドレスモード例外

#GP	ジャンプ先のオフセットが CS セグメントの範囲を超えているか、または 0 ～ FFFFH の実効アドレス空間外の場合。この状態は、32 ビット・アドレス・サイズ・オーバーライド・プリフィックスを使用した場合に生じることがある。

仮想 8086 モード例外

実アドレスモードと同じ例外。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#GP(0)	メモリアドレスが非標準形式の場合。

JMP - Jump

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
EB *cb*	JMP *rel8*	有効	有効	RIP = RIP + 64 ビットに符号拡張された 8 ビットのディスプレースメントとして、short ジャンプする。
E9 *cw*	JMP *rel16*	N.S.	有効	次の命令との相対分量分だけ相対 near ジャンプする。64 ビットモードでは未サポート。
E9 *cd*	JMP *rel32*	有効	有効	RIP = RIP + 64 ビットに符号拡張された 32 ビットのディスプレースメントとして、相対 near ジャンプする。
FF /4	JMP *r/m16*	N.S.	有効	アドレス = 符号拡張された *r/m16* として、絶対間接 near ジャンプする。64 ビットモードでは未サポート。
FF /4	JMP *r/m32*	N.S.	有効	アドレス = 符号拡張された *r/m32* として、絶対間接 near ジャンプする。64 ビットモードでは未サポート。
FF /4	JMP *r/m64*	有効	N.E.	RIP = レジスタまたはメモリからの 64 ビット・オフセットとして、絶対間接 near ジャンプする。
EA *cd*	JMP *ptr16:16*	無効	有効	オペランドで指定されるアドレスに絶対 far ジャンプする。
EA *cp*	JMP *ptr16:32*	無効	有効	オペランドで指定されるアドレスに絶対 far ジャンプする。
FF /5	JMP *m16:16*	有効	有効	*m16:16* で指定されるアドレスに絶対間接 far ジャンプする。
FF /5	JMP *m16:32*	有効	有効	*m16:32* で指定されるアドレスに絶対間接 far ジャンプする。 32 ビットモードの場合、セレクタの指示先がゲートであれば、RIP はゼロ拡張された 32 ビットのディスプレースメントであり、ゲートから取得される。指示先がゲートでなければ、RIP はゼロ拡張された 32 ビットのオフセットであり、命令で参照される far ポインタから取得される。
FF /5	JMP *m16:64*	有効	N.E.	64 ビットモードの場合、セレクタの指示先がゲートであれば、RIP は 64 ビットのディスプレースメントであり、ゲートから取得される。指示先がゲートでなければ、RIP はゼロ拡張された 32 ビットのオフセットであり、命令で参照される far ポインタから取得される。

影響を受けるフラグ

タスクスイッチが行われた場合はすべてのフラグが影響を受け、タスクスイッチが行われなかった場合はどのフラグも影響を受けない。

IA-32e モードでの操作

64 ビットに拡張される。
オペランド・サイズは 64 ビットに固定される。

保護モード例外

#GP(0)	ターゲット・オペランド、コールゲート、または TSS 内のオフセットがコード・セグメントの範囲を超えている場合。
	デスティネーション・オペランド、コールゲート、タスクゲート、または TSS 内のセグメント・セレクタが NULL の場合。
	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタを使用してメモリにアクセスしたが、その内容が NULL セグメント・セレクタであった場合。
#GP（セレクタ）	セグメント・セレクタ・インデックスがディスクリプタ・テーブルの範囲外の場合。
	デスティネーション・オペランド内のセグメント・セレクタによって指されているセグメント・ディスクリプタが、コンフォーミング・コード・セグメント、非コンフォーミング・コード・セグメント、コールゲート、タスクゲート、タスク・ステート・セグメントのいずれのディスクリプタでもない場合。
	非コンフォーミング・コード・セグメントの DPL が CPL に等しくない場合。
	（コールゲートを使用しない場合）セグメントのセグメント・セレクタの RPL が CPL より大きい場合。
	コンフォーミング・コード・セグメントの DPL が CPL より大きい場合。
	コールゲート、タスクゲート、または TSS のセグメント・ディスクリプタからの DPL が CPL より小さいか、あるいはコールゲート、タスクゲート、または TSS のセグメント・セレクタの RPL より小さい場合。
	コールゲート内のセレクタのセグメント・ディスクリプタが、それがコード・セグメントであることを示していない場合。
	タスクゲート内のセグメント・セレクタのセグメント・ディスクリプタが使用可能な TSS を示していない場合。
	TSS のセグメント・セレクタのローカル / グローバル・ビットがローカルとしてセットされている場合。
	TSS のセグメント・ディスクリプタが、TSS がビジーであるか、または使用不可能であることを示している場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NP（セレクタ）	アクセスされるコード・セグメントが存在しない場合。
	コールゲート、タスクゲート、または TSS が存在しない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。

#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合（メモリからターゲットをフェッチするときだけ発生する）。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。

仮想 8086 モード例外

#GP(0)	ターゲット・オペランドがコード・セグメントの範囲を超えている場合。
	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合（メモリからターゲットをフェッチするときだけ発生する）。

互換モード例外

64 ビットモード例外と同じ。

64 ビットモード例外

#GP(0)	メモリアドレスが非標準の場合。
	デスティネーション・オペランド内のターゲット・オフセットが非標準の場合。
	デスティネーション・オペランド内のターゲット・オフセットが新しいコード・セグメントの範囲外の場合。
	デスティネーション・オペランド内のセグメント・セレクタが NULL の場合。
	64 ビットゲート内のコード・セグメント・セレクタが NULL の場合。
	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタを使用してメモリにアクセスしたが、その内容が NULL セグメント・セレクタであった場合。

#GP（セレクタ）	コード・セグメントまたは 64 ビット・コール・ゲートがディスクリプタ・テーブルの範囲外の場合。
	コード・セグメントまたは 64 ビット・コール・ゲートが非標準領域に重複する場合。
	64 ビット・コール・ゲートからのセグメント・ディスクリプタが非標準領域内にある場合。
	デスティネーション・オペランド内のセグメント・セレクタによって指されているセグメント・ディスクリプタが、コンフォーミング・コード・セグメント、非コンフォーミング・コード・セグメント、64 ビット・コール・ゲートのいずれのディスクリプタでもない場合。
	デスティネーション・オペランド内のセグメント・セレクタによって指されているセグメント・ディスクリプタがコード・セグメントであり、D ビットと L ビットの両方がセットされている場合。
	非コンフォーミング・コード・セグメントの DPL が CPL に等しくないか、またはセグメントのセグメント・セレクタの RPL が CPL より大きい場合。
	コンフォーミング・コード・セグメントの DPL が CPL より大きい場合。
	64 ビット・コール・ゲートからの DPL が、CPL、または 64 ビット・コール・ゲートの RPL より小さい場合。
	64 ビット・コール・ゲートの上位タイプ・フィールドが 0x0 でない場合。
	64 ビット・コール・ゲートからのセグメント・セレクタがディスクリプタ・テーブルの範囲外の場合。
	64 ビットゲートにあるセレクタの指示先のコード・セグメント・ディスクリプタで、L ビットがセットされておらず、D ビットがクリアされていない場合。
	64 ビット・コール・ゲートからのセグメント・セレクタのセグメント・ディスクリプタが、そのコールゲートがコード・セグメントであることを示していない場合。
	コード・セグメントが非コンフォーミングであり、CPL != DPL の場合。
	コード・セグメントがコンフォーミングであり、CPL < DPL の場合。
#NP（セレクタ）	コード・セグメントまたは 64 ビット・コール・ゲートが存在しない場合。
#UD	（64 ビットモードのみ）far ジャンプがメモリ内の絶対アドレスを直接指定している場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

LAHF - Load Status Flags into AH Register

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
9F	LAHF	無効	有効	AH ← EFLAGS(SF:ZF:0:AF:0:PF:1:CF) をロードする。

影響を受けるフラグ

なし（すなわち、EFLAGS レジスタ内のフラグの状態は影響を受けない）。

IA-32e モードでの操作

64 ビットモードでは無効である。

保護モード例外

なし。

実アドレスモード例外

なし。

仮想 8086 モード例外

なし。

互換モード例外

なし。

64 ビットモード例外

#UD	64 ビットモードの場合。

LAR - Load Access Rights Byte

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
0F 02 /*r*	LAR *r16,r/m16*	有効	有効	*r16* ← FF00H でマスクされた *r/m16*
0F 02 /*r*	LAR *r32,r/m32*	有効	有効	*r32* ← 00FxFF00H でマスクされた *r/m32*
0F 02 /*r*	LAR *r64,r/m64*	有効	N.E.	*r64* ← 00FxFF00H でマスクされたゼロ拡張の *r/m32*

影響を受けるフラグ

アクセス権が正常にロードされた場合はZF フラグが 1 にセットされ、ロードできなかった場合は 0 にクリアされる。

IA-32e モードでの操作

レガシーモードと同じ。
デフォルトのオペランド・サイズは32 ビットである。
新しいレジスタ R8 ～ R15 へのアクセスが可能である。

保護モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタを使用してメモリにアクセスしたが、その内容が NULL セグメント・セレクタであった場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合（メモリからターゲットをフェッチするときだけ発生する）。

実アドレスモード例外

#UD	実アドレスモードでは、LAR 命令は認識されない。

仮想 8086 モード例外

#UD	LAR 命令は仮想 8086 モードで実行することはできない。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合（メモリからターゲットをフェッチするときだけ発生する）。

LDDQU - Load Unaligned Double Quadword

オペコード	命令	64 ビット モード	互換 / レガシー モード	説明
F2 0F F0 /r	LDDQU *xmm1*, *mem*	有効	有効	*mem* からアライメントの合っていないデータをロードし、*xmm1* にダブル・クワッドワードを返す。

操作

メモリアドレスのアライメントが16バイトに合っていない場合でも、プロセッサによっては、最大32バイトをロードし、16バイトをデスティネーションに返すことができる。

IA-32e モードでの操作

XMM8 ～ XMM15 へのアクセスが可能である。

SIMD 浮動小数点例外

なし。

保護モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#UD	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE3 が 0 の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。

実アドレスモード例外

#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#UD	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE3 が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）ページフォルトが発生した場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#NM	CR0 の TS がセットされた場合。
#UD	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE3 が 0 の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。

LDMXCSR - Load MXCSR Register

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
0F,AE,/2	LDMXCSR *m32*	有効	有効	*m32* から MXCSR レジスタをロードする。

IA-32e モードでの操作

レガシーモードと同じ。

同等のインテル C/C++ コンパイラ組み込み関数

_mm_setcsr(unsigned int i)

数値例外

なし。

保護モード例外

#GP(0)	CS、DS、ES、FS、または GS セグメント内のメモリ・オペランドの実効アドレスが無効の場合。
	MXCSR の予約されているビットをセットしようとした場合。
#SS(0)	SS セグメント内のアドレスが無効の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#UD	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP(0)	MXCSR の予約されているビットをセットしようとした場合。
#GP(0)	オペランドの一部が 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#NM	CR0 の TS がセットされた場合。
#UD	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。

仮想 8086 モード例外

実アドレスモードと同じ例外。

#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
	MXCSR の予約されているビットをセットしようとした場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#NM	CR0 の TS がセットされた場合。
#UD	CR0 の EM がセットされた場合。
	CR4 の OSFXSR が 0 の場合。
	CPUID 機能フラグ SSE が 0 の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

LDS/LES/LFS/LGS/LSS - Load Far Pointer

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
C5 /*r*	LDS *r16,m16:16*	無効	有効	メモリから DS:*r16* に far ポインタをロードする。
C5 /*r*	LDS *r32,m16:32*	無効	有効	メモリから DS:*r32* に far ポインタをロードする。
0F B2 /*r*	LSS *r16,m16:16*	有効	有効	メモリから SS:*r16* に far ポインタをロードする。
0F B2 /*r*	LSS *r32,m16:32*	有効	有効	メモリから SS:*r32* に far ポインタをロードする。
0F B2 /*r*	LSS *r64,m16:64*	有効	N.E.	メモリから SS:*r64* に far ポインタをロードする。
C4 /*r*	LES *r16,m16:16*	無効	有効	メモリから ES:*r16* に far ポインタをロードする。
C4 /*r*	LES *r32,m16:32*	無効	有効	メモリから ES:*r32* に far ポインタをロードする。
0F B4 /*r*	LFS *r16,m16:16*	有効	有効	メモリから FS:*r16* に far ポインタをロードする。
0F B4 /*r*	LFS *r32,m16:32*	有効	有効	メモリから FS:*r32* に far ポインタをロードする。
0F B4 /*r*	LFS *r64,m16:64*	有効	N.E.	メモリから FS:*r64* に far ポインタをロードする。
0F B5 /*r*	LGS *r16,m16:16*	有効	有効	メモリから GS:*r16* に far ポインタをロードする。
0F B5 /*r*	LGS *r32,m16:32*	有効	有効	メモリから GS:*r32* に far ポインタをロードする。
0F B5 /*r*	LGS *r64,m16:64*	有効	N.E.	メモリから GS:*r64* に far ポインタをロードする。

影響を受けるフラグ

なし。

IA-32e モードでの操作

レガシーモードと同じ。
デフォルトのオペランド・サイズは32 ビットである。
新しいレジスタ R8 ～ R15 へのアクセスが可能である。

保護モード例外

#UD　　ソース・オペランドがメモリ・ロケーションでない場合。

#GP(0)	SS レジスタに NULL セレクタがロードされた場合。
	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタを使用してメモリにアクセスしたが、その内容が NULL セグメント・セレクタであった場合。
#GP（セレクタ）	SS レジスタがロードされようとしたとき、次のいずれかが真であった場合。(1) セグメント・セレクタ・インデックスがディスクリプタ・テーブルの範囲内でない。(2) セグメント・セレクタの RPL が CPL に等しくない。(3) セグメントが書き込み不可能なデータ・セグメントである。(4) DPL が CPL に等しくない。
	DS、ES、FS、または GS レジスタに NULL でないセグメント・セレクタがロードされようとしたとき、次のいずれかが真であった場合。(1) セグメント・セレクタ・インデックスがディスクリプタ・テーブルの範囲内でない。(2) セグメントがデータ・セグメントでも読み出し可能なコード・セグメントでもない。(3) セグメントがデータ・セグメントまたは非コンフォーミング・コード・セグメントであり、かつ RPL と CPL が共に DPL より大きい。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#SS（セレクタ）	SS レジスタがロードされようとしたとき、セグメントが存在しないとマークされていた場合。
#NP（セレクタ）	DS、ES、FS、または GS レジスタに NULL でないセグメント・セレクタがロードされようとしたとき、セグメントが存在しないとマークされていた場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#UD	ソース・オペランドがメモリ・ロケーションでない場合。

仮想 8086 モード例外

#UD	ソース・オペランドがメモリ・ロケーションでない場合。
#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#GP(0)	メモリアドレスが非標準形式の場合。
	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタを使用してメモリにアクセスしたが、その内容が NULL セグメント・セレクタであった場合。
	互換モードで、SS レジスタに NULL セレクタがロードされようとした場合。
	CPL3 の 64 ビットモードで、SS レジスタに NULL セレクタがロードされようとした場合。
	CPL3 以外の 64 ビットモードで RPL が CPL に等しくないときに、SS レジスタに NULL セレクタがロードされようとした場合。
#GP（セレクタ）	DS、ES、FS、または GS レジスタに NULL でないセグメント・セレクタがロードされようとしたとき、次のいずれかが真であった場合。(1) セグメント・セレクタ・インデックスがディスクリプタ・テーブルの範囲内でない。(2) ディスクリプタのメモリアドレスが非標準である。(3) セグメントがデータ・セグメントでも読み出し可能なコード・セグメントでもない。(4) セグメントがデータ・セグメントまたは非コンフォーミング・コード・セグメントであり、かつ RPL と CPL が共に DPL より大きい。
	SS レジスタがロードされようとしたとき、次のいずれかが真であった場合。(1) セグメント・セレクタ・インデックスがディスクリプタ・テーブルの範囲内でない。(2) ディスクリプタのメモリアドレスが非標準である。(3) セグメント・セレクタの RPL が CPL に等しくない。(4) セグメントが書き込み不可能なデータ・セグメントである。(5) DPL が CPL に等しくない。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
	メモリ・オペランドの実効アドレスが非標準の場合。
#SS（セレクタ）	SS レジスタがロードされようとしたとき、セグメントが存在しないとマークされていた場合。
#NP（セレクタ）	DS、ES、FS、または GS レジスタに NULL でないセグメント・セレクタがロードされようとしたとき、セグメントが存在しないとマークされていた場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。
#UD	ソース・オペランドがメモリ・ロケーションでない場合。

LEA - Load Effective Address

オペコード	命令	64 ビット モード	互換 / レガシー モード	説明
8D /*r*	LEA *r16,m*	有効	有効	*m* の実効アドレスをレジスタ *r16* にストアする。
8D /*r*	LEA *r32,m*	有効	有効	*m* の実効アドレスをレジスタ *r32* にストアする。
8D /*r*	LEA *r64,m*	有効	N.E.	*m* の実効アドレスをレジスタ *r64* にストアする。32 ビット・レジスタがゼロ拡張された結果、64 ビットになる。

影響を受けるフラグ

なし。

IA-32e モードでの操作

デフォルトのオペランド・サイズは32 ビットである。

保護モード例外

#UD	ソース・オペランドがメモリ・ロケーションでない場合。

実アドレスモード例外

#UD	ソース・オペランドがメモリ・ロケーションでない場合。

仮想 8086 モード例外

#UD	ソース・オペランドがメモリ・ロケーションでない場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
GP(0)	メモリアドレスが非標準形式の場合。
#UD	ソース・オペランドがメモリ・ロケーションでない場合。

LEAVE - High Level Procedure Exit

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
C9	LEAVE	有効	有効	SP を BP に設定し、次に BP をポップする。
C9	LEAVE	有効	有効	ESP を EBP に設定し、次に EBP をポップする。
C9	LEAVE	有効	N.E.	RSP を RBP に設定し、次に RBP をポップする。

影響を受けるフラグ

なし。

IA-32e モードでの操作

デフォルトの操作サイズは64 ビットである。
64 ビットモードでは、32 ビットの操作である。

保護モード例外

#SS(0)	EBP レジスタの指示先のロケーションが現在のスタック・セグメントの範囲内にない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	EBP レジスタの指示先のロケーションが 0 ～ FFFFH の実効アドレス空間の範囲外の場合。

仮想 8086 モード例外

#GP(0)	EBP レジスタの指示先のロケーションが 0 ～ FFFFH の実効アドレス空間の範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	メモリアドレスが非標準形式の場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

LES - Load Full Pointer

「LDS/LES/LFS/LGS/LSS - Load Far Pointer」を参照のこと。

LFENCE - Load Fence

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
0F AE /5	LFENCE	有効	有効	ロード操作をシリアル化する。

IA-32e モードでの操作

レガシーモードと同じ。

同等のインテル C/C++ コンパイラ組み込み関数

void_mm_lfence(void)

例外（すべての動作モード）

なし。

LFS - Load Full Pointer

「LDS/LES/LFS/LGS/LSS - Load Far Pointer」を参照のこと。

LGDT/LIDT - Load Global/Interrupt Descriptor Table Register

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
0F 01 /2	LGDT *m16&32*	N.E.	有効	*m* を GDTR にロードする。
0F 01 /3	LIDT *m16&32*	N.E.	有効	*m* を IDTR にロードする。
0F 01 /2	LGDT *m16&64*	有効	N.E.	*m* を GDTR にロードする。
0F 01 /3	LIDT *m16&64*	有効	N.E.	*m* を IDTR にロードする。

影響を受けるフラグ

なし。

64 ビットモードでの操作

64 ビットに拡張される。
オペランド・サイズは 8+2 バイトに固定される。
8 バイトのベースと 2 バイトの範囲をロードする。

保護モード例外

#UD	ソース・オペランドがメモリ・ロケーションでない場合。
#GP(0)	現行特権レベルが 0 でない場合。
	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタを使用してメモリにアクセスしたが、その内容が NULL セグメント・セレクタであった場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。

実アドレスモード例外

#UD	ソース・オペランドがメモリ・ロケーションでない場合。
#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	現行特権レベルが 0 でない場合。 メモリアドレスが非標準形式の場合。
#UD	ソース・オペランドがメモリ・ロケーションでない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。

LGS - Load Full Pointer

「LDS/LES/LFS/LGS/LSS - Load Far Pointer」を参照のこと。

LLDT - Load Local Descriptor Table Register

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
0F 00 /2	LLDT r/m16	有効	有効	セグメント・セレクタ r/m16 を LDTR にロードする。

影響を受けるフラグ

なし。

IA-32e モードでの操作

オペランド・サイズは16ビットに固定される。
64ビットのベースをロードするには、64ビットモードのディスクリプタを参照する。

保護モード例外

#GP(0)	現行特権レベルが 0 でない場合。
	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタの内容が NULL セグメント・セレクタの場合。
#GP（セレクタ）	セレクタ・オペランドの指示先がグローバル・ディスクリプタ・テーブル（GDT）でない場合、または GDT 内のエントリがローカル・ディスクリプタ・テーブルでない場合。
	セグメント・セレクタが GDT の範囲を超えている場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#NP（セレクタ）	LDT ディスクリプタが存在しない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。

実アドレスモード例外

#UD	実アドレスモードでは、LLDT 命令は認識されない。

仮想 8086 モード例外

#UD	仮想 8086 モードでは、LLDT 命令は認識される。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	現行特権レベルが 0 でない場合。
	メモリアドレスが非標準形式の場合。
#GP（セレクタ）	セレクタ・オペランドの指示先がグローバル・ディスクリプタ・テーブル（GDT）でない場合、または GDT 内のエントリがローカル・ディスクリプタ・テーブルでない場合。
	セグメント・セレクタが GDT の範囲を超えている場合。
#NP（セレクタ）	LDT ディスクリプタが存在しない場合。
#PF（フォルトコード）	ページフォルトが発生した場合。

LIDT - Load Interrupt Descriptor Table Register

「LGDT/LIDT - Load Global/Interrupt Descriptor Table Register」を参照のこと。

LMSW - Load Machine Status Word

オペコード	命令	64 ビット モード	互換/ レガシー モード	説明
0F 01 /6	LMSW r/m16	有効	有効	r/m16 を CR0 のマシン・ステータス・ワードにロードする。

影響を受けるフラグ

なし。

IA-32e モードでの操作

レガシーモードと同じ。
オペランド・サイズは 16 ビットに固定される。

保護モード例外

#GP(0)	現行特権レベルが 0 でない場合。
	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタを使用してメモリにアクセスしたが、その内容が NULL セグメント・セレクタであった場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。

仮想 8086 モード例外

#GP(0)	現行特権レベルが 0 でない場合。
	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	現行特権レベルが 0 でない場合。 メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。

LOCK - Assert LOCK# Signal Prefix

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
F0	LOCK	有効	有効	後の命令の実行中 LOCK# 信号をアサートする。

操作

AssertLOCK#(DurationOfAccompaningInstruction)

影響を受けるフラグ

なし。

IA-32e モードでの操作

レガシーモードと同じ。

保護モード例外

#UD	LOCK プリフィックスが、上記の「説明」の項に示していない命令に使用された場合。この、LOCK プリフックスが適用されようとした命令からは、他にも例外が発生する可能性がある。

実アドレスモード例外

#UD	LOCK プリフィックスが、上記の「説明」の項に示していない命令に使用された場合。この、LOCK プリフックスが適用されようとした命令からは、他にも例外が発生する可能性がある。

仮想 8086 モード例外

#UD	LOCK プリフィックスが、上記の「説明」の項に示していない命令に使用された場合。この、LOCK プリフックスが適用されようとした命令からは、他にも例外が発生する可能性がある。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

保護モード例外と同じ。

LODS/LODSB/LODSW/LODSD/LODSQ - Load String

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
AC	LODS m8	有効	有効	レガシーモードでは、アドレス DS:(E)SI のバイトを AL にロードする。64 ビットモードでは、アドレス (R)SI のバイトを AL にロードする。
AD	LODS m16	有効	有効	レガシーモードでは、アドレス DS:(E)SI のワードを AX にロードする。64 ビットモードでは、アドレス (R)SI のワードを AX にロードする。
AD	LODS m32	有効	有効	レガシーモードでは、アドレス DS:(E)SI のダブルワードを EAX にロードする。64 ビットモードでは、アドレス (R)SI のダブルワードを EAX にロードする。
REX.W + AD	LODS m64	有効	N.E.	アドレス (R)SI のクワッドワードを RAX にロードする。
AC	LODSB	有効	有効	レガシーモードでは、アドレス DS:(E)SI のバイトを AL にロードする。64 ビットモードでは、アドレス (R)SI のバイトを AL にロードする。
AD	LODSW	有効	有効	レガシーモードでは、アドレス DS:(E)SI のワードを AX にロードする。64 ビットモードでは、アドレス (R)SI のワードを AX にロードする。
AD	LODSD	有効	有効	レガシーモードでは、アドレス DS:(E)SI のダブルワードを EAX にロードする。64 ビットモードでは、アドレス (R)SI のダブルワードを EAX にロードする。
REX.W + AD	LODSQ	有効	N.E.	アドレス (R)SI のクワッドワードを RAX にロードする。

影響を受けるフラグ

なし。

IA-32e モードでの操作

64 ビットに拡張される。
デフォルトのオペランド・サイズは 32 ビットである。

保護モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタの内容が NULL セグメント・セレクタの場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#GP	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。

仮想 8086 モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、アライメントが合わないメモリ参照を行った場合。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

LOOP/LOOP*cc* - Loop According to ECX Counter

オペコード	命令	64 ビットモード	互換 / レガシーモード	説明
E2 *cb*	LOOP *rel8*	有効	有効	カウントをデクリメントし、カウント ≠ 0 の場合 short ジャンプする。
REX.W + E2 *cb*	LOOP *rel8*	有効	N.E.	カウントをデクリメントし、カウント ≠ 0 の場合 short ジャンプする。JMP Short では、RIP = RIP + 64 ビットに符号拡張された 8 ビットのオフセット。
E1 *cb*	LOOPE *rel8*	有効	有効	カウントをデクリメントし、カウント ≠ 0 でかつ ZF=1 の場合 short ジャンプする。
REX.W + E1 *cb*	LOOPE *rel8*	有効	N.E.	カウントをデクリメントし、カウント ≠ 0 でかつ ZF=1 の場合 short ジャンプする。JMP Short では、RIP = RIP + 64 ビットに符号拡張された 8 ビットのオフセット。
E0 *cb*	LOOPNE *rel8*	有効	有効	カウントをデクリメントし、カウント ≠ 0 でかつ ZF=0 の場合 short ジャンプする。
REX.W + E0 *cb*	LOOPNZ *rel8*	有効	N.E.	カウントをデクリメントし、カウント ≠ 0 でかつ ZF=0 の場合 short ジャンプする。JMP Short では、RIP = RIP + 64 ビットに符号拡張された 8 ビットのオフセット。

影響を受けるフラグ

なし。

IA-32e モードでの操作

64 ビットに拡張される。
オペランド・サイズは 64 ビットである。
JMP Short では、RIP = RIP + 64 ビットに符号拡張された 8 ビットのオフセット。

保護モード例外

#GP(0)　ジャンプ先のオフセットが CS セグメントの範囲を超えている場合。

実アドレスモード例外

#GP　ジャンプ先のオフセットが CS セグメントの範囲を超えているか、または 0 ～ FFFFH の実効アドレス空間外の場合。この状態は、32 ビット・アドレス・サイズ・オーバーライド・プリフィックスを使用した場合に生じることがある。

仮想 8086 モード例外

実アドレスモードと同じ例外。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#GP(0)	メモリアドレスが非標準形式の場合。

LSL - Load Segment Limit

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
0F 03 /*r*	LSL *r16,r/m16*	有効	有効	「*r16* ← セグメント範囲、セレクタ *r/m16*」のロードを行う。
0F 03 /*r*	LSL *r32,r/m32*	有効	有効	「*r32* ← セグメント範囲、セレクタ *r/m32*」のロードを行う。
REX.W + 0F 03 /*r*	LSL *r64,r/m32*	有効	有効	「*r64* ← ゼロ拡張されたセグメント範囲、セレクタ *r/m64*」のロードを行う。

影響を受けるフラグ

セグメント範囲が正常にロードされた場合はZF フラグが 1 にセットされ、ロードできなかった場合は0にクリアされる。

IA-32e モードでの操作

レガシーモードと同じ。
デフォルトのオペランド・サイズは32 ビットである。

保護モード例外

#GP(0)	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタを使用してメモリにアクセスしたが、その内容が NULL セグメント・セレクタであった場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

実アドレスモード例外

#UD	実アドレスモードでは、LSL 命令は認識されない。

仮想 8086 モード例外

#UD	仮想 8086 モードでは、LSL 命令は認識されない。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	メモリアドレスが非標準形式の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。
#AC(0)	アライメント・チェックがイネーブルにされていて、現行特権レベルが 3 のときにアライメントが合わないメモリ参照を行った場合。

LSS - Load Full Pointer

「LDS/LES/LFS/LGS/LSS - Load Far Pointer」を参照のこと。

LTR - Load Task Register

オペコード	命令	64 ビットモード	互換/レガシーモード	説明
0F 00 /3	LTR r/m16	有効	有効	r/m16 をタスクレジスタにロードする。

影響を受けるフラグ

なし。

IA-32e モードでの操作

オペランド・サイズは16ビットに固定される。
64ビットのベースをロードするには、64ビットモードのディスクリプタを参照する。

保護モード例外

#GP(0)	現行特権レベルが 0 でない場合。
	メモリ・オペランドの実効アドレスが CS、DS、ES、FS、または GS セグメントの範囲外の場合。
	DS、ES、FS、または GS レジスタを使用してメモリにアクセスしたが、その内容が NULL セグメント・セレクタであった場合。
#GP（セレクタ）	ソースセレクタの指示先のセグメントが TSS でないか、またはそのセグメントのタスクがすでにビジーの場合。
	セレクタが LDT を指しているか、または GDT の範囲を超えている場合。
#NP（セレクタ）	TSS が存在しないとマークされている場合。
#SS(0)	メモリ・オペランドの実効アドレスが SS セグメントの範囲外の場合。
#PF（フォルトコード）	ページフォルトが発生した場合。

実アドレスモード例外

#UD	実アドレスモードでは、LTR 命令は認識されない。

仮想 8086 モード例外

#UD	仮想 8086 モードでは、LTR 命令は認識されない。

互換モード例外

保護モード例外と同じ。

64 ビットモード例外

#SS(0)	SS セグメントを参照するメモリアドレスが非標準形式の場合。
#GP(0)	現行特権レベルが 0 でない場合。
	メモリアドレスが非標準形式の場合。
#GP（セレクタ）	ソースセレクタの指示先のセグメントが TSS でないか、またはそのセグメントのタスクがすでにビジーの場合。
	セレクタが LDT を指しているか、または GDT の範囲を超えている場合。
#NP（セレクタ）	TSS が存在しないとマークされている場合。
#PF（フォルトコード）	ページフォルトが発生した場合。