RISC-V IOMMUアーキテクチャ

仕様

IOMMUタスクグループ

バージョン v1.0.0, 2023-07-25：批准

目次

前文

この文書は批准される。

変更は許されない。希望する、あるいは必要とされる変更は、それに続く新たな延長の対象とすることができる。批准された延長が改訂されることはない。

著作権およびライセンス情報

投稿者

このRISC-V仕様は、（アルファベット順に）直接または間接的に貢献されたものである：

アーロン・ダービン、アレン・バウム、アヌプ・パテル、ダニエル・グレイシア・ペレス、デイビッド・クルックマイヤー、グレッグ・フェイバー、アフマド・ファワル、ゲルニー・D・ハント、ジョン・ハウザー、ジョシュ・シャイド、マット・エヴァンス、マヌエル・ロドリゲス、ニック・コシフィディス、ポール・ドナヒュー、ポール・ウォームズリー、ペリーヌ・ペレス、フィリップ・トムシッチ、リウール・ドゥクーソ、スコット・ネルソン、シーチー・チャオ、スニルV．L, Tomasz Jeznach, Vassilis Papaefstathiou, Vedvyas Shanbhogue

第1章.はじめに

システムMMU（SMMU）と呼ばれることもある入出力メモリ管理ユニット（IOMMU）は、直接メモリ・アクセスが可能な入出力（I/O）デバイスをシステム・メモリに接続するシステム・レベルのメモリ管理ユニット（MMU）である。

IOMMUを介してシステムに接続された各I/Oデバイスに対して、ソフトウェアはIOMMUでデバイスコンテキストを設定することができる。デバイスコンテキストは、デバイスに特定の仮想アドレス空間とその他のデバイスごとのパラメータを関連付ける。IOMMU で各デバイスに個別のデバイスコンテキストを与えることで、各デバイスを個別のオペレーティングシステム（ゲスト OS またはメイン（ホスト）OS）に個別に設定できます。デバイスがメモリ・アクセスを開始するたびに、IOMMU は何らかの形で一意のデバイス識別子によって元のデバイスを識別します。例えば、PCIe [1]の場合、発信デバイスは、PCI バス番号（8 ビット）、デバイス番号（5 ビット）、ファンクション番号（3 ビット）の一意の 16 ビットのトリプレット（ルーティング識別子または RID と総称される）、および IOMMU が複数の階層をサポートする場合はオプションで最大 8 ビットのセグメント番号によって識別される。本仕様書では、このような一意のデバイス識別子を device_id と呼び、最大 24 ビット幅の識別子をサポートする。

階層はPCI ExpressのI/O相互接続トポロジで、バス/デバイス/ファンクション番号のタプルと呼ばれる構成空間アドレスが一意である。また、フリットモードでは、セグメント番号がファンクションのIDに含まれることもあります。

デバイスによっては、プロセス・アドレス空間をデバイスと共有する機能である共有仮想アドレッシングをサポートしている場合があります。プロセス・アドレス空間をデバイスと共有することで、DMAをコア・カーネルのメモリ管理に依存することができ、アプリケーションやデバイス・ドライバから複雑さを取り除くことができます。デバイスにバインドした後、アプリケーションは静的または動的に割り当てられたバッファに対してDMAを実行するよう指示できます。このようなアドレッシングをサポートするために、ソフトウェアは1つ以上のプロセス・コンテキストをデバイス・コンテキストに設定することができます。このようなデバイスによって開始されるすべてのメモリ・アクセスには一意のプロセス識別子が付随し、IOMMUは一意のデバイス識別子と組み合わせて、ソフトウェアによってデバイス・コンテキストに設定された適切なプロセス・コンテキストを特定するために使用します。例えば、PCIe の場合、プロセスコンテキストは、一意の 20 ビットのプロセスアドレス空間識別子（PASID）によって識別される。この仕様では、このような一意のプロセス識別子を process_id と呼び、最大 20 ビット幅の識別子をサポートする。

IOMMUは、IOVAをSPAに変換し、DMAのメモリ保護を実施するために、2段階のアドレス変換プロセスを採用している。アドレス変換とメモリ保護を実行するために、IOMMUはCPUのMMUが第1段階と第2段

階のアドレス変換に使用するのと同じページテーブルフォーマットを使用します。CPUのMMUと同じページテーブルフォーマットを使用することで、DMAのメモリ管理の複雑さをある程度取り除くことができる。また、同じフォーマットを使用することで、CPUのMMUとIOMMUの両方で同じページテーブルを同時に使用することができる。

2段階のアドレス変換を無効にするオプションはないが、その段階の仮想メモリスキームをベア、つまりアドレス変換もメモリ保護も行わないように設定することで、どちらかの段階を効果的に無効にすることができる。

IOMMU が採用する仮想メモリー方式は、デバイスごとに次のように設定できる。

IOMMU。デバイスは I/O 仮想アドレス（IOVA）を使用して DMA を実行します。デバイスに選択されている仮想メモリ・スキームによって、デバイスが使用する IOVA はスーパーバイザ物理アドレス（SPA）、ゲスト物理アドレス（GPA）、または仮想アドレス（VA）になります。

両ステージで選択された仮想メモリ方式が Bare の場合、IOVA は SPA となる。IOMMU によるアドレス変換や保護は行われない。

ファーストステージで選択された仮想メモリスキームが Bare であり、セカンドステージのスキームが Bare でない場合、IOVA は GPA となる。ファーストステージは事実上無効となる。セカンドステージは GPA を SPA に変換し、設定されたメモリ保護を実行する。このような構成は、デバイス制御が仮想マシンに渡されるが、VM 内のゲスト OS が第 1 段階のアドレス変換を使用せず、そのようなデバイスからのメモリ・アクセスをさらに制約する場合に一般的に採用される。RISC-Vハートと比較すると、この構成は、RISC-VハートでGステージがアクティブでVSステージがベアに設定された2ステージのアドレス変換が有効であることと類似している。

ファーストステージで選択された仮想メモリスキームがBareではなく、セカンドステージのスキームがBareの場合、IOVAはVAとなる。セカンドステージは事実上無効となる。ファーストステージはVAをSPAに変換し、設定されたメモリ保護を実行する。この構成は、IOMMU がネイティブ OS によって使用される場合、またはデバイスの制御がハイパーバイザ自身によって保持される場合に通常採用される。RISC-Vハートと比較すると、この構成は、RISC-Vハートでシングルステージアドレス変換が有効であることと類似している。

どちらのステージでも選択された仮想メモリ方式がBareの場合、IOVAはVAとなる。2 段階のアドレス変換が有効である。第 1 段階は VA を GPA に変換し、第 2 段階は GPA を SPA に変換する。各ステージは、設定されたメモリ保護を実施する。このような構成は、デバイス制御が仮想マシンにパススルーされ、VMのゲストOSが第1段階のアドレス変換を使用して、そのようなデバイスによってアクセスされるメモリと関連する特権およびメモリ保護をさらに制約する場合に一般的に採用される。RISC-Vハートと比較すると、この構成は、GステージとVSステージの両方がアクティブなRISC-Vハートで、2ステージのアドレス変換が有効であることに似ています（Bareではありません）。

IOMMUにおけるDMAアドレス変換は、ソフトウェアが提供するデータ構造を使用してSPAを決定するのに必要な時間によってアクセス時間が長くなる可能性があるため、DMAアクセスに一定のパフォーマンス上の影響があります。CPUのMMUにおける同様のオーバーヘッドは、通常、アドレス変換をキャッシュするTLB（Translation Look-aside Buffer）を使用することで軽減される。IOMMUは、IOMMUアドレス変換キャッシュ（IOATC）と呼ばれる同様のアドレス変換キャッシュを採用することができる。IOMMUは、アドレス変換に使用されるメモリ常駐データ構造が変更されたときに、ソフトウェアがIOATCを同期させるメカニズムを提供する。ソフトウェアは、ゲスト・ソフトコンテキスト識別子（GSCID）と呼ばれるソ

フトウェア定義コンテキスト識別子でデバイス・コンテキストを構成して、デバイスの集合が同じVMに割り当てられていることを示し、共通の仮想アドレス空間にアクセスすることができる。ソフトウェアは、共通の仮想アドレス空間を共有するプロセスの集合を識別するために、プロセスソフトコンテキスト識別子（PSCID）と呼ばれるソフトウェア定義コンテキスト識別子でプロセスコンテキストを構成してもよい。IOMMU は、GSCID と PSCID を使用して IOATC のエントリをタグ付けし、重複を回避して無効化操作を簡素化することができる。

デバイスによっては、変換プロセスに参加し、自身のメモリ・アクセス用にデバイス側 ATC（DevATC）を提供するものもある。DevATCを提供することで、デバイスはトランスレーション・キャッシングの責任を共有し、それによってIOATCにおける「スラッシング」の確率を減らすことができる。DevATCのサイズは次のとおりである。

また、プリフェッチ・トランスレーションによってDMAレイテンシを最適化するためにデバイスが使用することもある。このようなメカニズムには、プロトコルを使用したデバイスと IOMMU の密接な協力が必要です。例えばPCIeの場合、Address Translation Services（ATS）プロトコルがデバイスによって使用され、DevATC内のキャッシュにトランスレーションを要求し、ソフトウェア・アドレス変換データ構造による更新と同期させることができる。デバイスがアドレス変換プロセスに参加することで、I/Oページフォールトの使用も可能になり、コアカーネルメモリマネージャがデバイスによってアクセスされる可能性のあるすべての物理メモリを常に常駐させる必要がなくなります。例えば PCIe の場合、デバイスはページ要求インターフェイス（PRI）を実装して、変換を要求したページが利用可能でないことが判明した場合に、メモリマネージャにページを常駐させるよう動的に要求することができます。IOMMU は、PCIe ATS や PCIe PRI [1]などのサービスを実現するために、デバイスとの特別なソフトウェア・インターフェイスやプロトコルをサポートする場合があります。

IMSIC (Incoming Message-Signaled Interrupt Controller) で構築されたシステムでは、IOMMU はハイパーバイザーによって、ゲスト OS によって制御されるデバイスからのメッセージシグナル割り込み (MSI) を IMSIC 内のゲスト割り込みファイルに向けるようにプログラムされることがあります。デバイスからの MSI は単なるメモリ書き込みであるため、IOMMU が他のメモリ書き込みに適用するのと同じアドレス変換が当然適用されます。しかし、RISC-V Advanced Interrupt Architecture [2]では、ソフトウェアを簡素化するためと、メモリ常駐割り込みファイルをオプションでサポートするために、IOMMUが仮想マシンに向けたMSIを特別に扱うことを要求しています。デバイス・コンテキストは、仮想割り込みファイルへのメモリ・アクセスを識別し、デバイス・コンテキスト内でソフトウェアによって設定されたMSIアドレス変換テーブルを使用して変換されるパラメータでソフトウェアによって設定されます。

1.1. 用語集

表1.用語と定義

期間	定義
AIA	RISC-V アドバンスト割り込みアーキテクチャ [2]。
ATS / PCIe ATS	アドレス変換サービス：DevATCをサポートするPCIeプロトコル[1]。
CXL	Compute Express Linkバス規格。
DC / デバイス・コンテキスト	デバイスと、そのデバイスが割り当てられているVMを識別する、状態をハードウェアで表現したもの。
DDT	デバイス・ディレクトリ・テーブル：デバイスコンテキスト構造を見つけるために、一意のデバイス識別子を使って走査される基数木構造。

DDI	Device-directory-index (デバイス・ディレクトリ・インデックス)：リーフまたはノンリーフDDT構造へのインデックスとして使用される一意のデバイス識別子のサブフィールド。
デバイスID	DMA または割り込み要求の送信元を識別するための最大 24 ビットの識別番号。PCIeデバイスの場合は、ルーティング識別子（RID）[1]。
デバテック	デバイスのアドレス変換キャッシュ。
DMA	ダイレクト・メモリー・アクセス。
GPA	Guest Physical Address（ゲスト物理アドレス）：仮想マシンの仮想化物理メモリ空間内のアドレス。

期間	定義
GSCID	ゲストソフトコンテキスト識別子：仮想マシンに割り当てられたデバイスの集合を一意に識別するためにソフトウェアが使用する識別番号。IOMMU は IOATC エントリーに GSCID を付けることができる。同じ GSCID でプログラムされたデバイスコンテキストは、同じセカンドステージページテーブルでプログラムされている必要があります。
ゲスト	仮想マシン内のソフトウェア。
HPM	ハードウェア・パフォーマンス・モニター。
ハイパーバイザー	仮想化を制御するソフトウェアの実体。
身分証明書	識別子。
IMSIC	着信メッセージ信号割り込みコントローラ。
アイオーエーティーシー	IOMMU Address Translation Cache：アドレス変換に使われるデータ構造をキャッシュするIOMMUのキャッシュ。
IOVA	I/O仮想アドレス：デバイスによるDMA用の仮想アドレス。
三井住友海上	メッセージ信号による割り込み。
OS	オペレーティングシステム。
PASID	Process Address Space Identifier（プロセス・アドレス空間識別子）：プロセスのアドレス空間を識別する。PASID値はリクエストのPASID TLPプレフィックスで提供される。
PBMT	ページベースのメモリタイプ。
ピーピーエヌ	物理的なページ番号
PRI	Page Request Interface（ページ・リクエスト・インターフェース） - デバイスがOSのメモリ・マネージャー・サービスにページを常駐させるよ

	う要求することを可能にするPCIeプロトコル[1]。
PC	プロセスのコンテキスト。
PCIe	Peripheral Component Interconnect Expressバス規格[1]。
ピーディーアイ	Process-directory-index: リーフまたは非リーフPDT構造へのインデックスに使用される一意なプロセス識別子のサブフィールド。
PDT	プロセス・ディレクトリ・テーブル：プロセスコンテキスト構造を見つけるために一意なプロセス識別子を使って走査される基数木データ構造。
PMA	物理メモリ属性。
ピーエムピー	物理メモリ保護。
ピーピーエヌ	物理的なページ番号
PRI	Page Request Interface - PCIeプロトコル[1]で、デバイスがOSのメモリ・マネージャ・サービスにページを常駐させるよう要求できるようにする。
プロセスID	プロセス・コンテキストを識別するための最大20ビットの識別番号。PCIeデバイスの場合はPASID [1]。
期間	**定義**
ピーエスシーアイディー	Process soft-context identifier（プロセス・ソフトコンテキスト識別子）：ソフトウェアが一意のアドレス空間を識別するために使用する識別番号。IOMMU は IOATC エントリに PSCID を付けることができる。
PT	ページの表。
PTE	Page Table Entry（ページ・テーブル・エントリー）。ページ・テーブルのリーフまたはノンリーフ・エントリ。
予約	将来の使用のために予約されたレジスタまたはデータ構造フィールド。データ構造の予約フィールドは、ソフトウェアによって0に設定されなければならない。ソフトウェアはレジスタの予約フィールドを無視し、同じレジスタの他のフィールドに値を書き込むときは、これらのフィールドに保持されている値を保持しなければならない。
RID / PCIe RID	PCIeルーティング識別子[1]。

RO	読み取り専用 - レジスタ・ビットは読み取り専用であり、ソフトウェアで変更することはできない。明示的に定義されている場合、これらのビットは変化するハードウェアの状態を反映するために使用され、その結果、実行時にビットの値が変化するのを観察することができる。 ビットをセットするオプション機能が実装されていない場合、ビットはゼロにハードワイヤリングされなければならない。
RW	リード・ライト - レジスタ・ビットはリード・ライトであり、ソフトウェアによって希望の状態にセットまたはクリアされる。 ビットに関連するオプション機能が実装されていない場合、ビットはゼロにハードワイヤリングされることが許可される。
RW1C	Write-1-to-Clearステータス - レジスタ・ビットは、読み出されるとステータスを示す。セット・ビットはステータス・イベントを示し、1bを書き込むとクリアされる。RW1Cビットに0bを書き込んでも効果はない。 ビットを設定するオプション機能が実装されていない場合、ビットは読み取り専用で、ゼロにハードワイヤされていなければなりません。
RW1S	Read-Write-1-to-set：レジスタ・ビットは、読み出されるとステータスを示す。ビットは1bを書き込むことでセットされる。RW1Sビットに0bを書き込んでも効果はありません。 ビットを導入するオプション機能が実装されていない場合、ビットは読み取り専用で、ゼロにハードワイヤされていなければならない。
SOC	システムオンチップ、システムオンチップとも呼ばれる。
スパ	Supervisor Physical Address（スーパバイザ物理アドレス）：メモリおよびメモリマップド・リソースにアクセスするために使用される物理アドレス。
TLP	トランザクション層のパケット。
バージニア	バーチャルアドレス
VM	仮想マシン：実際のコンピュータ・システムの効率的で分離された複製。本仕様では、ハイパーバイザー拡張機能をサポートするRISC-Vハートが仮想化モードを1に設定して実行したときにアクセス可能な、リソースとステートの集合体を指す。
ブイエムエム	仮想マシンモニター。ハイパーバイザーとも呼ばれる。
期間	**定義**

VS	仮想スーパーバイザ：仮想化モードでのスーパーバイザ権限。
ワール	どのような値でも書き込み、正当な値を読み取る：ビットエンコーディングのサブセットに対してのみ定義されるレジスタフィールドの属性。
WPRI	Writes 値を保持、Reads 値を無視：将来の使用のために予約されているレジスタ・フィールドの属性。

1.2. 使用モデル

1.2.1. 非仮想化OS

非仮想化OSは、以下の重要なシステムレベル機能にIOMMUを使用することができる：

1. 誤動作デバイスからの不正なメモリアクセスからオペレーティングシステムを保護する。
2. 64ビット環境で32ビットデバイスをサポート（バウンスバッファの回避）
3. 断片化された物理アドレスへの連続した仮想アドレスのマッピングをサポート（スキャッター/ギャザーリストの回避）
4. 共有仮想アドレッシングのサポート

IOMMUがない場合、デバイスは特権メモリなどのあらゆるメモリにアクセスし、悪意のある、あるいは意図しない破損を引き起こす可能性がある。これは、ハードウェアのバグ、デバイス・ドライバのバグ、あるいは悪意のあるソフトウェアやハードウェアによるものかもしれない。

IOMMUは、デバイスがアクセスできるメモリを制限することで、OSがこのような意図しない破損を防御するメカニズムを提供する。図1に示されているように、OSはIOVAを変換するページ・テーブルでIOMMUを構成し、それによってアクセス可能なアドレスをページ・テーブルで許可されたものに制限することができる。

レガシー32ビット・デバイスは、4GiB以上のメモリにアクセスできない。IOMMUは、そのアドレス再マッピング機能により、デバイスが（適切なアクセス許可があれば）システム内の任意のアドレスに直接アクセスできるシンプルなメカニズムを提供する。IOMMUがなければ、OSは4GiB以下のメモリに割り当てられたバッファ（バウンス・バッファとも呼ばれる）を使ってデータをコピーするしかない。このシナリオでは、IOMMUがシステム・パフォーマンスを向上させる。

IOMMUは、すべてのメモリが連続している必要なく、I/O用にメモリの大領域を割り当てることができるので、散在/収集DMAを実行するのに便利です。連続した仮想アドレス範囲は、そのような断片化された物理アドレスと、その仮想アドレス範囲でプログラムされたデバイスにマッピングすることができます。

IOMMUは、デバイスとプロセスアドレス空間を共有する機能である共有仮想アドレッシングをサポートするために使用することができます。DMAに使用される仮想アドレスは、IOMMUによってSPAに変換される。

IOMMUを非仮想化OSで使用する場合、必要なアドレス変換と保護機能はファーストステージで十分であり、セカンドステージはBareに設定することができる。

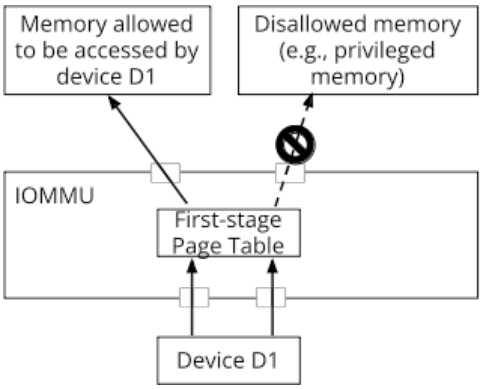


図1.非仮想化OSにおけるデバイスの分離

1.2.2. ハイパーバイザー

IOMMUは、仮想マシンで動作するゲストオペレーティングシステムが、ハイパーバイザーの介入を最小限に抑えながら、I/Oデバイスを直接制御できるようにする。

デバイスを直接制御するゲスト OS は、ゲスト物理アドレスでデバイスをプログラムします。デバイスがこれらのアドレスを使用してメモリアクセスを実行するとき、IOMMU は、ハイパーバイザが提供するアドレス変換データ構造を参照しながら、これらのゲスト物理アドレスをスーパーバイザ物理アドレスに変換する責任を負います。

図2はその概念を示している。デバイスD1はVM-1に直接割り当てられ、デバイスD2はVM-2に直接割り当てられている。VMMは、各デバイスに使用する第2ステージのページ・テーブルを設定し、D1がアクセスできるメモリをVM-1関連メモリに、D2がアクセスできるメモリをVM-2関連メモリに制限する。

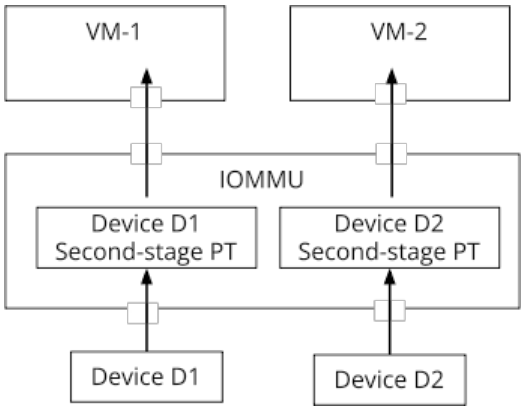


図2.直接デバイス割り当てを可能にするDMA変換

ゲスト OS が制御するデバイスからの MSI を処理するために、ハイパーバイザは IOMMU を構成して、それらの MSI を IMSIC 内のゲスト割り込みファイル（図 3 を参照）またはメモリ常駐割り込みファイルにリダイレクトします。IOMMU は、ハイパーバイザが提供する MSI アドレス変換データ構造を使用して MSI のリダイレクトを実行する責任を負います。すべての割り込みファイルは、実ファイルであれ仮想フ

ァイルであれ、自然にアライメントされた4KiBページのアドレス空間を占有するため、必要なアドレス変換は、通常のRISC-Vページベースのアドレス変換でサポートされているのと同じ、仮想（ゲスト）ページアドレスから物理ページアドレスへのアドレス変換です。

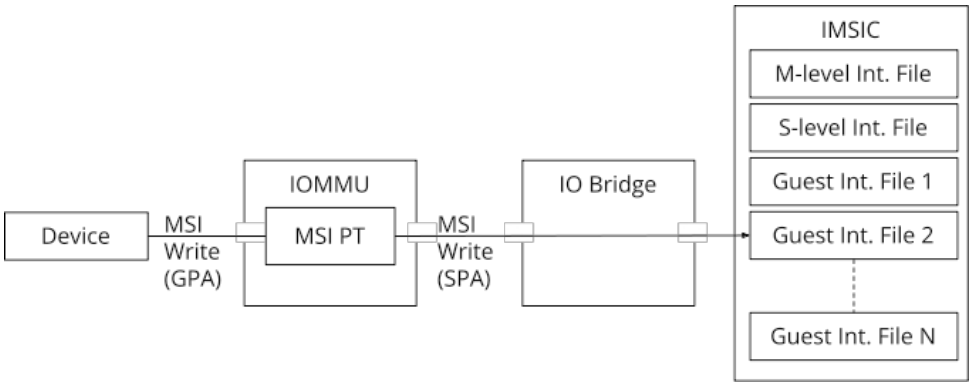


図3. ゲストがプログラムしたMSIをIMSICゲスト割り込みファイルに直接変換するためのMSIアドレス変換

1.2.3. ゲストOS

ハイパーバイザーは、ハードウェア・エミュレーションを通じて、あるいはハイパーバイザーとのソフトウェア・インターフェース（パラ仮想化とも呼ばれる）を使用するようにゲストOSを啓発することによって、仮想IOMMU機能を提供することができる。ゲストOSは、仮想IOMMUによって提供される機能を使用して、自身が制御する第1ステージのページテーブルを使用することで、非仮想化OSについて説明したのと同じ利点を利用することができる。ハイパーバイザーは、仮想マシンのアドレス空間を仮想化し、VMに渡されたデバイスからVMに関連するメモリへのメモリアクセスを格納するために、制御する第2ステージのページテーブルを確立する。

2 段階のアドレス変換が有効な場合、ゲスト OS が管理する第 1 段階のページテーブ ルを使用して IOVA が最初に GPA に変換され、ハイパーバイザーが管理する第 2 段階のページテーブ ルを使用して GPA が SPA に変換されます。

図4はそのコンセプトを示している。

IOMMU は、デバイス D1 に対して第 1 段および第 2 段のページテーブルを使用してアドレス変換を実行するように構成される。第2ステージは通常、ハイパーバイザによって使用され、GPAをSPAに変換し、デバイスD1をVM-1に関連するメモリに制限します。第 1 ステージは通常、ゲスト OS によって構成され、VA を GPA に変換し、デバイス D1 のアクセスを VM-1 メモリのサブセットに含めます。

デバイスD2ではセカンドステージのみがアクティブで、ファーストステージはBareに設定されている。

ホスト OS やハイパーバイザーは、D3 のようなデバイスを保持することもできる。デバイス D3 に必要なアドレス変換と保護機能はファーストステージで十分であり、セカンドステージは Bare に設定される。

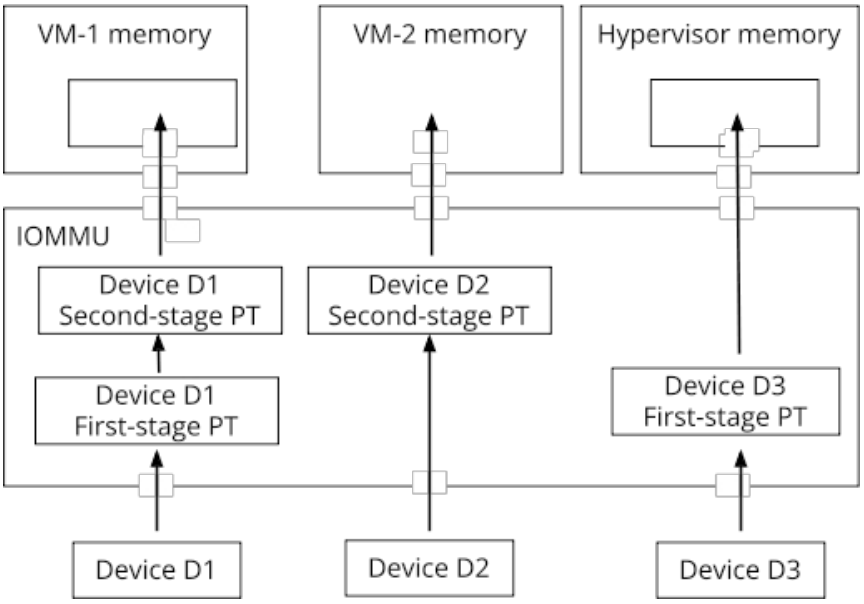


図4. ゲストOSのIOMMUにおけるアドレス変換

1.3. 配置とデータの流れ

図5は、RISC-Vハートを搭載した典型的なSOC（System on a Chip）の例である。このSOCには、メモリ・コントローラと複数のIOデバイスが組み込まれている。このSOCには、IOMMUの2つのインスタンスも組み込まれている。デバイスは IO ブリッジとシステム・インターコネクトに直接接続されるか、IO プロトコル・トランザクションからシステム・インターコネクト・トランザクションへの変換が必要な場合はルート・ポートを介して接続される。例えば、PCIe [1]の場合、ルート・ポートは、関連する仮想PCI-PCIブリッジを介して階層の一部をマッピングし、PCIe IOプロトコル・トランザクションをシステム・インターコネクト・トランザクションにマッピングするPCIeポートである。

最初の IOMMU インスタンスである IOMMU 0（IO Bridge 0 に関連）は、Root Port をシステムファブリック/インターコネクトにインターフェイスする。1つまたは複数のエンドポイントデバイスは、このルートポートを介してSoCにインターフェースされる。PCIeの場合、ルート・ポートには、IOMMUがPCIe ATSプロトコルをサポートするために使用されるIOMMUへのATSインターフェイスが組み込まれています。この例では、PCIe ATSプロトコル[1]を使用してデバイスがIOMMU 0から取得したトランスレーションを保持するデバイス側ATC（DevATC）を持つエンドポイントデバイスを示している。

ルートポートを使用した IO プロトコルからシステムファブリックプロトコルへの変換が必要ない場合、デバイスはシステムファブリックと直接インターフェイスすることができる。第 2 の IOMMU インスタンスである IOMMU 1（IO ブリッジ 1 に関連）は、ルートポートを使用せずに、デバイス（IO デバイス A および B）をシステムファブリックにインターフェイスする様子を示しています。

IO ブリッジはデバイスとシステムインターコネクトの間に置かれ、DMA トランザクションを処理する。

IO デバイスは、IO 仮想アドレス（VA、GVA または GPA）を使用して DMA トランザクションを実行できます。IO Bridge は関連する IOMMU を呼び出して、IOVA を Supervisor Physical Addresses (SPA) に変換します。

IOMMUはアウトバウンドトランザクションには呼び出されない。

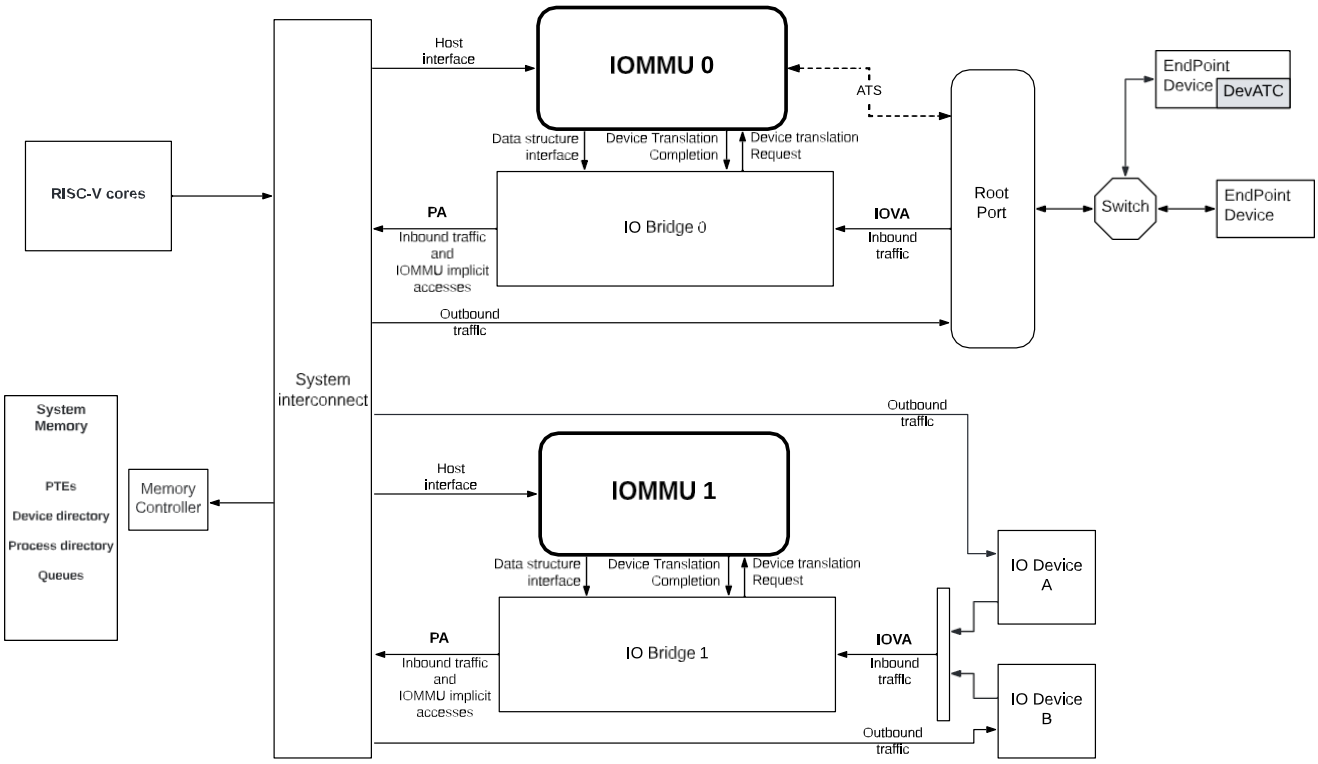


図5.SoCにおけるIOMMUの統合例。

IOMMU は、受信トランザクションのアドレス変換と保護のために IO ブリッジによって呼び出される。インバウンドトランザクションに関連するデータは、IOMMU では処理されない。IOMMU は IO ブリッジに対するルックアサイド IP のように動作し、いくつかのインターフェイスを持つ（図 6 参照）：

- ホスト・インターフェース：IOMMUのメモリ・マップされたレジスタにアクセスし、グローバル・コンフィギュレーションやメンテナンスを実行するためのインターフェース。
- デバイストランスレーションリクエストインタフェース： IO Bridge からトランスレーションリクエストを受け取るインタフェースである。このインターフェースで IO Bridge はリクエストに関する以下のような情報を提供します：
  a. トランザクションに関連付けられたハードウェア ID - device_id、および必要に応じて process_id とその有効性。IOMMU は、要求されたアドレス変換を実行するためのコンテキスト情報を取得するために、ハードウェア ID を使用する。
  b. IOVA と取引の種類（翻訳または未翻訳）。
  c. リクエストが読み取り、書き込み、実行、またはアトミック操作のどれであるか。
    i. Execute requestedは、リクエストに明示的に関連付けられなければならない(例: PCIe PASIDを使用)。明示的に要求されていない場合、デフォルトは0でなければならない。
  d. リクエストに関連付けられた特権モード。特権モードがリクエストに明示的に関連付けられていない場合(例: PCIe PASIDを使用)、デフォルトの特権モードはUserでなければならない。process_idがないリクエストの場合、特権モードはUserでなければならない。

e. リクエストによってアクセスされたバイト数。

f. IO Bridge はまた、IOMMU によって解釈されないが、IOMMU から IO Bridge への応答と一緒に返される、追加の不透明な情報（例えばタグ）を提供することができる。IOMMU は翻訳要求を順番通りに完了することが許されているため、このような情報は IO Bridge が以前の要求と完了を関連付けるために使用することができる。

- データ構造インターフェイス： IOMMUがメモリへの暗黙のアクセスに使用する。これは IO ブリッジへのリクエスタインタフェースであり、メインメモリから必要なデータ構造をフェッチするために使用される。このインターフェースは以下のアクセスに使用される：
  a. コンテキスト情報と翻訳ルールを取得するためのデバイスとプロセスのディレクトリ。
  b. IOVAを翻訳するための第一段および/または第二段のページテーブルエントリ。
  c. ソフトウェアとのインターフェイスに使用されるメモリ内キュー（コマンドキュー、フォールトキュー、ページリクエストキュー）。
- デバイス変換完了インターフェース：以前に要求されたアドレス変換に対するIOMMUからの完了応答を提供するインターフェースである。完了インターフェースは以下のような情報を提供する：
  a. リクエストが正常に完了したか、あるいはフォルトが発生したかを示す、 リクエストのステータス。
  b. リクエストが正常に完了した場合、SPA（Supervisor Physical Address）。
  c. リクエストに関連する不透明な情報（タグなど）（該当する場合）。
  d. Svpbmtがサポートされている場合、IOMMUアドレス変換ページテーブルから得られるページベースのメモリタイプ（PBMT）。IOMMUは、第1ステージのページテーブルエントリと第2ステージのページテーブルエントリの間で解決されたページベースのメモリタイプを提供する。
- ATS インターフェース：ATS インターフェースは、オプションの PCIe ATS 機能が IOMMU でサポートされている場合、PCIe ルートポートを介して ATS 対応エンドポイントと通信するために使用されます。このインターフェイスが使用される：
  a. エンドポイントからATS変換要求を受信し、エンドポイントに完了を返す。ルートポートは、要求を発信したエンドポイントがCXLタイプ1またはタイプ2のデバイスであるかどうかの表示を提供することができる。
  b. ATS「無効化要求」メッセージをエンドポイントに送信し、エンドポイントから「無効化完了」メッセージを受信する。
  c. エンドポイントから「ページ要求」と「停止マーカー」メッセージを受信し、エンドポイントに「ページ要求グループ応答」メッセージを送信する。

メモリ常駐割り込みファイル（MRIF）（RISC-V Advanced Interrupt Architecture [2]参照）に着信 MSI を記録するインターフェースは、実装固有のものです。着信 MSI を MRIF に記録し、関連する*通知* MSI を生成するための IOMMU と IO ブリッジ間の責任分担は、実装に固有です。

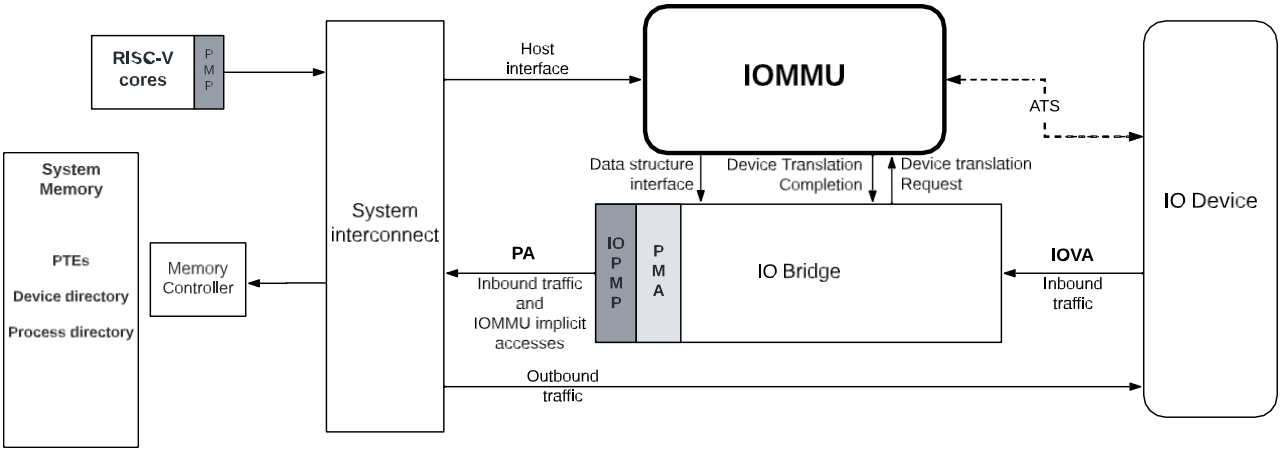


図6.IOMMUのインターフェース。

RISC-Vハーツと同様に、物理メモリ属性（PMA）と物理メモリ保護（PMP）チェックは、IOMMUがバイパス（ベア・モード）であっても、すべてのインバウンドIOトランザクションで完了しなければならない。PMA および PMP チェッカーの配置と統合は、プラットフォームの選択による。

PMA および PMP チェッカーは IOMMU の外部に存在する。上の例では、IO Bridge にある。

IOMMU 自身によるデータ構造インターフェースを介した暗黙のアクセスは PMA チェッカーによってチェックされる。PMAは物理プラットフォームの構成と密接に結びついており、多くの詳細は本質的にプラットフォーム固有である。

IOMMU がデータ構造インタフェースを使用して実行するメモリ・アクセスは、一般に、デバイス主導のメモリ・アクセスと順序付ける必要はない。

IOMMU は、アドレス変換を実行するために必要なデータ構造にアクセスするために、データ構造イン ターフェース上で暗黙のメモリ・アクセスを生成することがある。このようなアクセスは、元のデバイス主導のメモリ・アクセスによってブロックされてはならない。

IOブリッジは、IOブリッジとシステムインターコネクトによって定義された必要なハザードチェックと他のルールを満たすために、データ構造インターフェイス上でメモリアクセスの順序付けを実行することができる。

IOMMU は、解決された PBMT（PMA、IO、NC）を、デバイス変換完了インターフェース上の変換されたアドレスとともに IO ブリッジに提供する。IO Bridge の PMA チェッカーは、提供された PBMT を使用して、関連するメモリページの PMA を上書きすることができる。

PMP チェッカは、バス・アクセス・イニシエータのハードウェア ID を使用して、物理メ モリ・アクセス権限を決定することができる。IOMMU 自身が暗黙のアクセスのためのバスアクセスイニシエータである

ため、IOMMU ハードウェア ID は、PMP チェッカが適切なアクセス制御ルールを選択するために使用してもよい。

IOMMU は、IO ブリッジが提供するハードウェア ID の真正性を検証しない。

IO ブリッジおよび/またはルートポートは、ハードウェア ID を認証するための適切なメカニズムを 含まなければならない。SOC によっては、これは

> デバイスが SOC に統合され、その ID が不変であるという特性。例えば PCIe の場合、ハードウェア ID を認証するために、PCIe で定義されたアクセス制御サービス（ACS）のソース検証機能を使用することができる。IO ブリッジの他の実装固有のメソッドが、このような認証を実行するために提供されてもよい。

1.4. IOMMUの特徴

RISC-V IOMMU仕様のバージョン1.0は、以下の機能をサポートしている：

- メモリー・ベースのデバイス・コンテキストで、パラメーターとアドレス変換構造を見つける。デバイス・コンテキストは、ハードウェアが提供する一意のdevice_idを使用して配置される。サポートされるdevice_idの幅は24ビットまでである。
- ハードウェアが提供する一意のprocess_idを使用して、パラメータとアドレス変換構造を見つけるための、メモリベースのプロセスコンテキスト。サポートされるprocess_idは最大20ビットである。
- 16ビットのGSCIDと20ビットのPSCID。
- 2段階のアドレス変換。
- RISC-V Privileged仕様[3]で規定されているページベースの仮想メモリシステムにより、CPU MMUとIOMMUに共通のページテーブルを使用するか、IOMMUに個別のページテーブルを使用するかをソフトウェアが柔軟に選択できる。
- 最大57ビットの仮想アドレス幅、56ビットのシステム物理アドレス幅、59ビットのゲスト物理アドレス幅。
- PTE アクセス済みビットとダーティビットのハードウェア更新。
- RISC-V Advanced Interrupt Architecture [2]で規定されたMSIページテーブルを使用して、仮想割り込みファイルへのメモリアクセスとMSIアドレス変換を識別する。
- Svnapot と Svpbmt 拡張。
- PCIe ATS および PRI サービス [1]。翻訳要求に応答して、SPA ではなく GPA に IOVA を翻訳することをサポートする。
- ハードウェア・パフォーマンス・モニター（HPM）。
- ソフトウェアにサービスを要求するためのMSIおよびワイヤシグナル割り込み。
- ソフトウェアがデバッグをサポートするためにアドレス変換を要求するためのレジスタ・インターフェース。

IOMMUがサポートする機能は、5.3節のcapabilitiesレジスタを使用して検出することができる。

第2章 データ構造データ構造

デバイスコンテキスト（DC）と呼ばれるデータ構造は、デバイスとアドレス空間を関連付け、IOMMUがアドレス変換を実行するために使用するその他のデバイスごとのパラメータを保持するためにIOMMUによって使用される。DCの場所を特定するために、device_idを使用して走査されるデバイス・ディレクトリ・テーブル（DDT）と呼ばれる基数木データ構造が使用される。

デバイスが使用するアドレス空間は、デバイスの制御がゲストOSに渡される際に、第2段階のアドレス変換と保護を必要とする場合があります。ゲスト OS は、ゲスト OS が制御するデバイスが使用する IOVA を GPA に変換するための第 1 段階のページテーブ ルをオプションで提供することができます。第 1 段の使用が必要ない場合は、第 1 段のアドレス変換スキームを Bare に選択することで、第 1 段を効果的に無効にできます。第 2 段階は、GPA を SPA に変換するために使用されます。

デバイスの制御がハイパーバイザーまたはホストOS自身によって保持されている場合、必要なアドレス変換と保護を実行するには、第1ステージだけで十分である。第2ステージのアドレス変換スキームをBareにプログラムすることによって、第2ステージのスキームをデバイスに対して効果的に無効にすることができる。

第2段アドレス変換がBareでない場合、DCはルート第2段ページテーブルのPPN、仮想マシン単位でキャッシュされたアドレス変換の無効化を容易にするゲストソフトコンテキストID（GSCID）、および第2段アドレス変換スキームを保持する。

デバイスによっては、複数のプロセス・コンテキストをサポートしており、各コンテキストが異なるプロセス、したがって異なる仮想アドレス空間に関連付けられている場合がある。このようなデバイスのコンテキストには、アドレス空 間 を 識別するprocess_idが設定されている場合があります。このようなデバイスは、メモリ・アクセスを行う際に、process_idをdevice_idと一緒に通知して、アクセスされるアドレス空間を特定する。このようなデバイスの例としては、複数のプロセスコンテキストをサポートするGPUが挙げられ、各コンテキストは異なるユーザープロセスに関連付けられ、GPUはユーザープロセス自体によって提供される仮想アドレスを使用してメモリにアクセスすることができます。process_idに関連付けられたアドレス空間の選択をサポートするために、DCは、プロセスコンテキスト（PC）と呼ばれるデータ構造を見つけるためにprocess_idのフィールドを使用してインデックス付けされた基数木データ構造であるルートプロセスディレクトリテーブル（PDT）のPPNを保持します。

PDTがアクティブな場合、第一段階のアドレス変換のコントロールは（PC）に保持される。

PDTがアクティブでない場合、第1段アドレス変換の制御はDC本体に保持される。

第1段アドレス変換制御には、ルート第1段ページテーブルのPPN、アドレス空間単位でキャッシュされた

アドレス変換の無効化を容易にするプロセスソフトコンテキストID（PSCID）、および第1段アドレス変換スキームが含まれる。

ゲスト OS が制御するデバイスからの MSI を処理するには、IOMMU はそれらの MSI を IMSIC 内のゲスト割り込みファイルにリダイレクトできなければなりません。デバイスからの MSI は単なるメモリ書き込みであるため、IOMMU が他のメモリ書き込みに適用するのと同じアドレス変換が当然適用されます。しかし、IOMMU アーキテクチャは、ソフトウェアを簡素化するためと、メモリ常駐割り込みファイルをオプションでサポートするために、仮想マシンに向けた MSI を特別に扱うことができます。この機能をサポートするために、IOMMU アーキテクチャはデバイスコンテキストに MSI アドレスマスクとアドレスパターンを追加します。

デバイスからの MSI の変換や変換を制御するための MSI ページテーブルの実物理アドレス。仮想マシンに対するMSIのIOMMUサポートは、Advanced Interrupt Architecture仕様で規定されている。

DC はさらに、デバイスが生成することを許可されるトランザクションのタイプを制御する。このような制御の一例として、デバイスが PCIe で定義されたアドレス変換サービス（ATS）[1]の使用を許可されるかどうかがある。

デバイス・コンテキスト構造には2つの形式がある：

- **ベース・フォーマット** - 32バイトのサイズで、MSIの特別な扱いがセクション 2.3.3はIOMMUがサポートしていない。
- **拡張フォーマット** - サイズは64バイトであり、セクション2.3.3で規定されるMSIを 変換するための追加フィールドで基本フォーマットDCを拡張する。

capabilities.MSI_FLATが1の場合、拡張フォーマットが使用され、それ以外の場合はベースフォーマットが使用される。

DCを見つけるために使用されるDDTは、サポートされるdevice_idの最大幅に応じて、1レベル、2レベル、または3レベルの基数木になるように構成される。DDT基数木を横断するためのデバイス・ディレクトリ・インデックス（DDI）を取得するためのdevice_idのパーティショニングは以下の通りである：

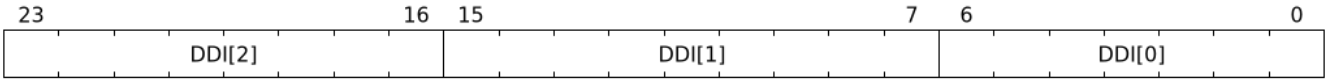


図7.基本フォーマットのdevice_idパーティショニング

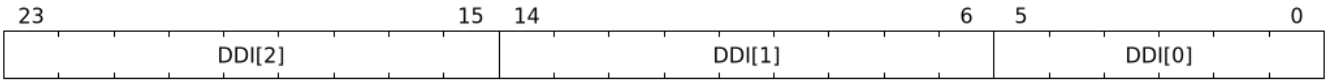


図8.拡張フォーマットdevice_idパーティショニング

PDTは、デバイスがサポートするprocess_idの最大幅に応じて、1レベル、2レベル、または3レベルの基数木になるように構成することができる。PDTの基数木を走査するためのプロセス・ディレクトリ・インデックス（PDI）を得るためのprocess_idの分割は以下の通りである：

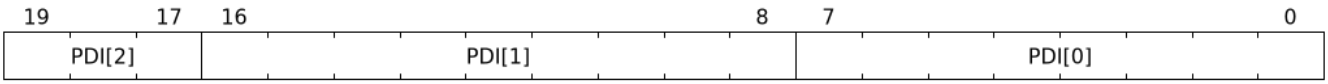


図9 PDT基数木トラバーサルにおけるprocess_idのパーティショニング

process_idパーティショニングは、各プロセス・ディレクトリ・テーブルに最大4KiB（1ページ）のメモリを必要とするように設計されている。20ビット幅のprocess_idを使用する場合、テーブルのルートは完全には入力されません。ルート・テーブルに 32 KiB を使用させるオプションも検討されましたが、これらのテーブルは実行時に割り当てられ

、1 ページより大きい連続したメモリ割り当てがゲストとハイパーバイザーのメモリ・アロケータにストレスを与える可能性があるため、採用されませんでした。

すべてのRISC-V IOMMU実装は、メインメモリにあるDDTとPDTをサポートする必要がある。I/Oメモリ内のデータ構造をサポートすることは要求されていないが、この仕様で禁止されているわけではない。

2.1. デバイス・ディレクトリ・テーブル（DDT）

DDTは、デバイス・ディレクトリ・インデックス（DDI）のビットを使ってインデックスを付けた1、2、または3レベルの基数木である。

device_idでDCを探す。

以下の図はDDT基幹ツリーを示している。ルート・デバイス・ディレクトリ・テーブルのPPNは、デバイス・ディレクトリ・テーブル・ポインタ（ddtp）と呼ばれるメモリ・マップド・レジスタに保持される。

各有効な非リーフ(NL)エントリーは8バイトサイズで、次のデバイス・ディレクトリ・テーブルの PPN を

保持する。有効なリーフ・デバイス・ディレクトリ・テーブル・エントリーはデバイス・コンテキスト

(DC)を保持する。

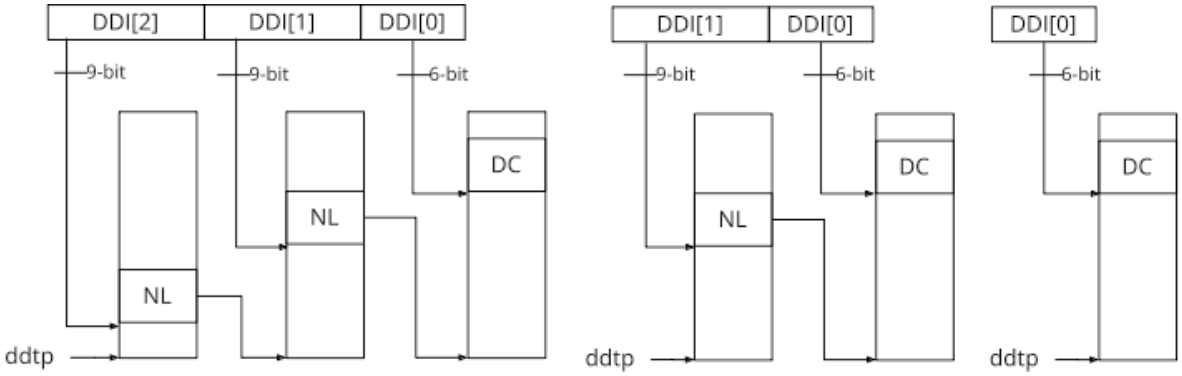


図10.拡張フォーマットDCによる3レベル、2レベル、1レベルのデバイス・ディレクトリ

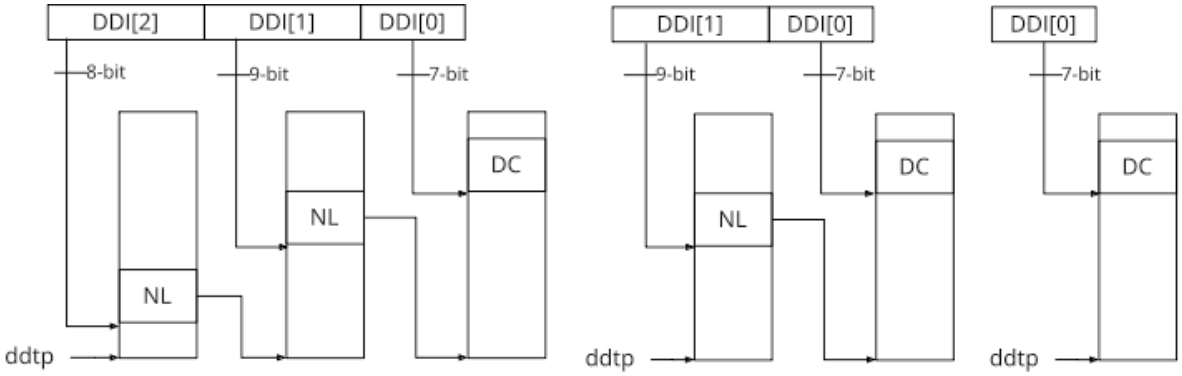


図11.基本フォーマットDCによる3レベル、2レベル、1レベルのデバイス・ディレクトリ

2.1.1. ノンリーフDDTエントリー

有効な(V==1)非リーフDDTエントリは、次のレベルのDDTのPPNを提供する。

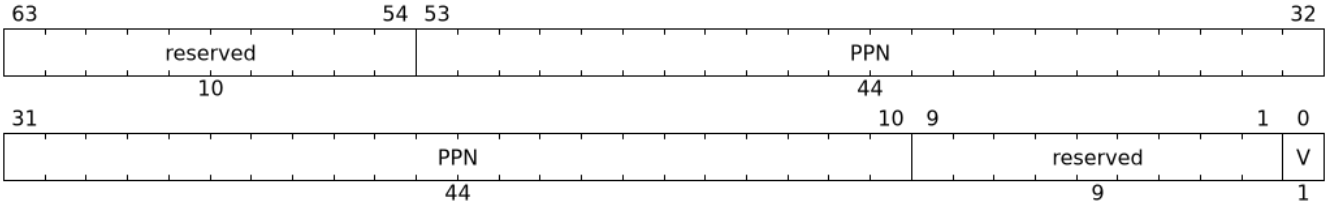


図12.非リーフのデバイス・ディレクトリ・テーブル・エントリ

2.1.2. 葉のDDTエントリー

リーフDDTページはDDI[0]でインデックスされ、デバイスコンテキスト（DC）

を保持する。ベース・フォーマットでは DC は 32 バイトである。拡張フォー

マットでは DC は 64 バイトである。

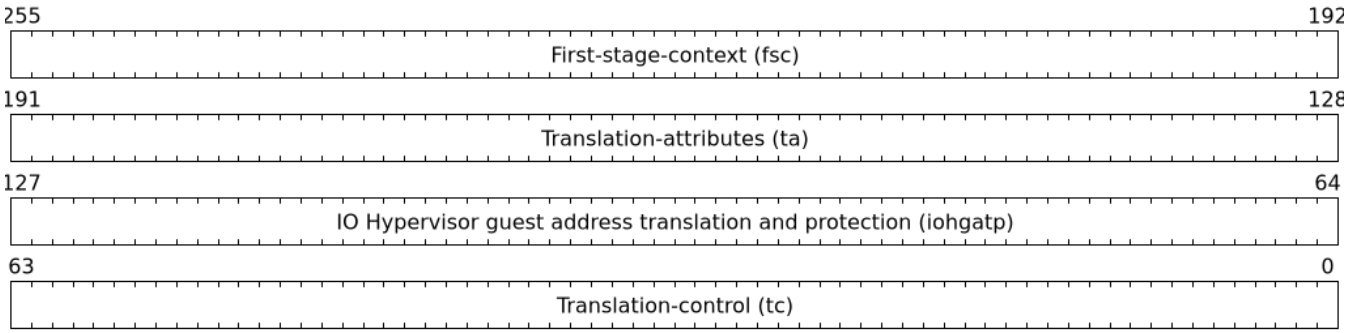


図13.ベースフォーマット・デバイスコンテキスト

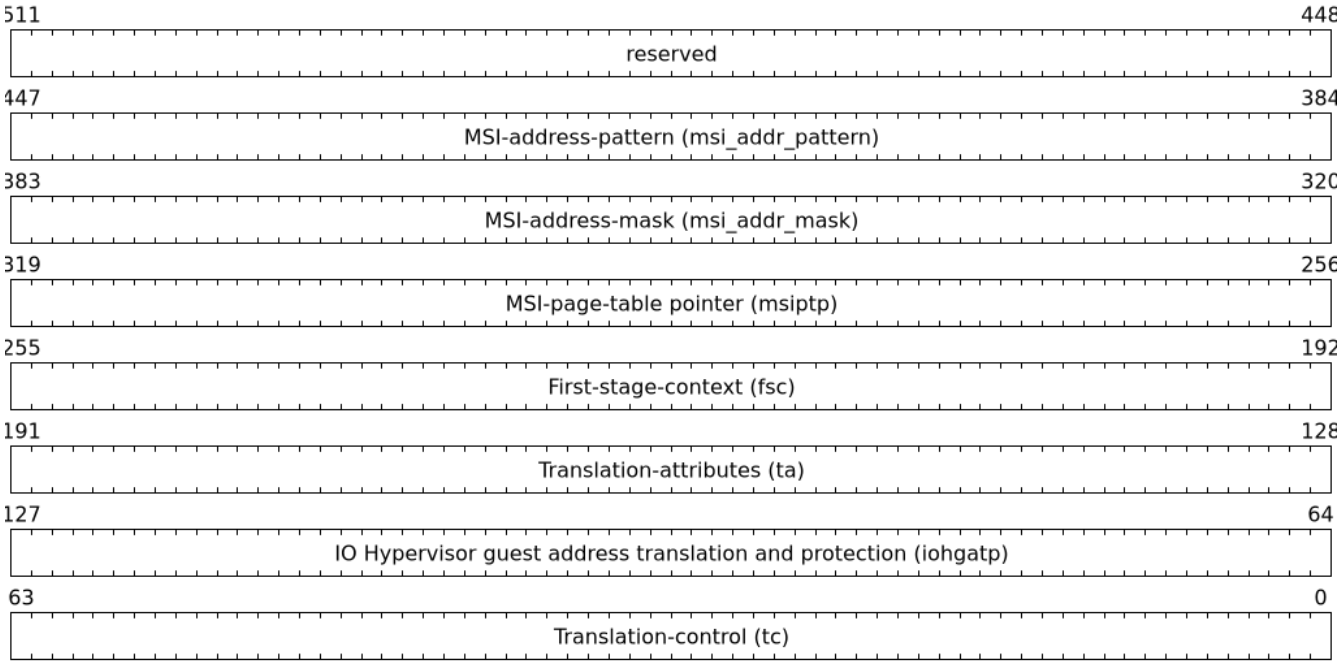


図14.拡張フォーマット・デバイス・コンテキスト

DCは、ベースフォーマットでは4つの64ビット・ダブルワードとして、拡張フォーマットでは8つの64ビット・ダブルワードとして解釈される。メモリ上の各ダブルワードのバイト順序は、リトルエンディアンま

たはビッグエンディアンで、 fctl.BE（セクション5.4）で決定されるエンディアンとなる。IOMMU は DC フィールドをどのような順序で読んでもよい。

2.1.3. デバイス・コンテキスト・フィールド

翻訳コントロール（tc）

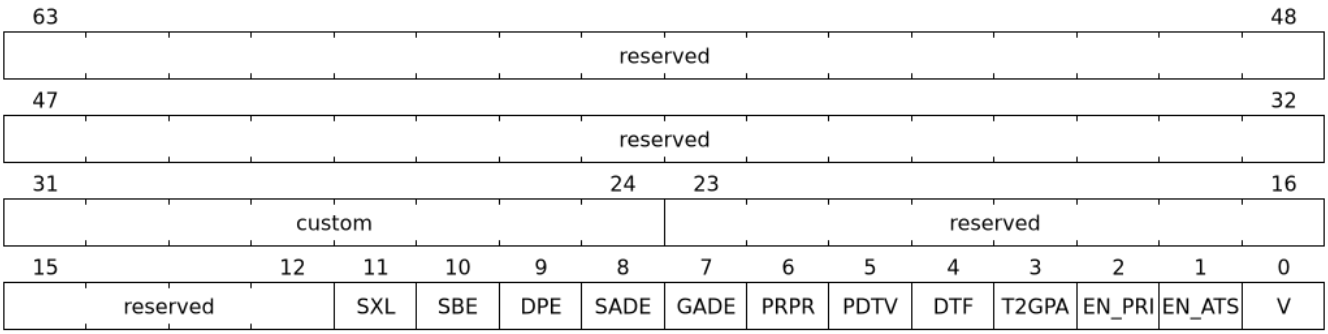


図15.翻訳制御（tc）フィールド

Vビットが1の場合、DCは有効である。Vビットが0の場合、DCの他のすべてのビットはdon't-careであり、ソフトウェアが自由に使用できる。

IOMMU が PCIe ATS 仕様[1]をサポートしている場合（capabilities レジスタを参照）、EN_ATS ビットは ATS トランザクション処理を有効にするために使用される。EN_ATS が 1 に設定されている場合、IOMMU は以下のインバウンドトランザクションをサポートする。

- トランザクション実行のための翻訳された読み取り
- 翻訳された読み取りトランザクション
- 翻訳された書き込み/AMOトランザクション
- PCIe ATS翻訳リクエスト
- PCIe ATS無効化完了メッセージ

EN_ATS ビットが 1 で、T2GPA ビットが 1 に設定されている場合、IOMMU はデバイスからの PCIe ATS 変換要求の完了時に、提供する変換の許可とサイズを決定するために 2 段階のアドレス変換を実行する。ただし、IOMMU はレスポンスにおいて、IOVA の変換として SPA ではなく GPA を返す。この動作モードでは、デバイスの ATC は IOVA の変換として GPA をキャッシュし、その後の変換されたメモリ・アクセス・トランザクションのアドレスとして GPA を使用します。通常、変換されたリクエストは SPA を使用し、IOMMU がさらに変換を実行する必要はない。しかし、T2GPA が 1 の場合、デバイスからのトランスレートされたリクエストは GPA を使用し、IOMMU によって第 2 ステージのページテーブルを使用して SPA にトランスレートされる。T2GPA 制御により、デバイスが ATS 機能を誤用し、VM と関連付けられていないメモリにアクセスしようとした場合でも、ハイパーバイザはデバイスからの DMA を封じ込めることができる。

> T2GPA が有効な場合、PCIe ATS Translation Request に応答してデバイスに提供される

アドレスは、デバイスを他のピアデバイスやホストに接続する I/O ファブリック（PCI スイッチなど）によって直接ルーティングされない。また、デバイス内のピアツーピアトランザクション（デバイスの機能間など）がサポートされている場合、そのようなアドレスはデバイス内でルーティングされません。

T2GPAを1に設定するハイパーバイザーは、プロトコル固有の手段により、IOMMUがGPAを変換し、PAに基づくトランザクションをメモリまたはPAにルーティングするように、変換されたアクセスがホストを経由してルーティングされることを保証しなければならない。

ピア・デバイス。たとえばPCIeの場合、アクセス・コントロール・サービス（ACS）は、ピアツーピア（P2P）要求を常にアップストリームにリダイレクトしてホストに送るように設定する必要がある。

1 に設定された T2GPA の使用は、PCIe ATS 変換リクエストに応答して返される変換アドレスでタグ付けされたキャ ッシュを実装するデバイスと互換性がない可能性がある。

T2GPA を 1 に設定する代わりに、認証プロトコルがデバイスによってサポートされている場合、ハイパーバイ ザはデバイスとの信頼関係を確立することができる。例えばPCIe の場合、PCIe コンポーネント測定認証（CMA）機能は、デバイスの構成とファームウェア/実行可 能ファイル（Measurement）およびハードウェア ID（Authentication）を検証し、そのような信頼関係を確立するメカニズ ムを提供する。

EN_PRI ビットが 0 の場合、デバイスからの PCIe "Page Request" メッセージは無効なリクエストである。デバイスから受信した "Page Request "メッセージは、"Page Request Group Response "メッセージで応答される。通常、ソフトウェア・ハンドラがこの応答メッセージを生成します。しかし、条件によってはIOMMU 自身が応答を生成することもある。IOMMU が生成する「ページ要求グループ応答」メッセージでは、PRG-response-PASID-required (PRPR) ビットが 1 にセットされると、関連する「ページ要求」にPASID があった場合、IOMMU の応答メッセージに PASID を含める必要があることを示す。

PASIDをサポートし、"PRG Response PASID Required "ケイパビリティビットを 1に設定している関数は、関連する "Page Request "メッセージにPASIDがあった場合、"Page Request Group Response "メッセージにPASIDが含まれることを期待する。ケイパビリティビットが 0 の場合、関数は「ページ要求グループ応答」メッセージの PASID を期待せず、PASID を含む応答を受け取った場合の関数の動作は未定義である。PRPR ビットは、「PRG Response PASID Required」ケイパビリティビットに保持されている値で構成されるべきである。

disable-translation-fault (DTF)ビットを1に設定すると、アドレス変換処理で発生したフォルトの報告が無効になる。DTFを1に設定しても、フォールト・トランザクションに応答してデバイスに生成されるエラー・レスポンスは無効にならない。DTFを1に設定しても、アドレス変換プロセスに関連しないIOMMUからのフォルト報告は無効にならない。DTFが1のときに報告されないフォルトを表11に示す。

ハイパーバイザーは、仮想マシンの異常終了など、エラーの多発につながる可能性がある状況を特定した場合、DTFを1に設定してフォールトレポートを無効にすることができる。

DC.fscフィールドは、第1段翻訳用のコンテキストを保持する。PDTVビットが1の場合、このフィールドはプロセス・ディレクトリ・テーブル・ポインタ（pdtp）を保持する。PDTVビットが0の場合、DC.fscフィールドは(iosatp)を保持する。

PDTVビットは、DCが複数のプロセスコンテキストをサポートするデバイスに関連付け られており、その結果、そのメモリアクセスで有効なprocess_idを生成する場合に 1に設定されることが期待される。例えばPCIeの場合、リクエストにPASIDがあれば、そのPASIDがprocess_idとして使われる。

PDTVが1のとき、DPEビットを1に設定して、有効なprocess_idがないリクエストを変換するためにprocess_idのデフォルト値として0を使用できるようにすることができる。PDTVが0のとき、DPEビットは将来のために予約される。

標準エクステンション。

IOMMU は、capabilities.AMO_HWAD が 1 の場合、GADE および SADE ビットの 1 設定をサポートする。capabilities.AMO_HWAD が 0 の場合、これらのビットは予約されている。

GADE が 1 の場合、IOMMU は第 2 ステージの PTE の A ビットと D ビットをアトミックに更新する。GADE が 0 の場合、IOMMU は、A ビットが 0 の場合、またはメモリアクセスがストアで D ビットが 0 の場合に、元のアクセスタイプに対応するゲストページフォールトを引き起こします。

SADE が 1 の場合、IOMMU は第 1 段 PTE の A ビットと D ビットをアトミックに更新する。SADE が 0 の場合、IOMMU は、A ビットが 0 の場合、またはメモリ・アクセスがストアで D ビットが 0 の場合に、元のアクセス・タイプに対応するページ・フォールトを引き起こす。

SBE が 0 の場合、PDT エントリーと第 1 ステージの PTE への暗黙のメモリー・アクセスはリトルエンディアンである。SBE がサポートする値は、fctl.BE フィールドの値と同じである。

SXL フィールドは、表 3 に定義されているように、サポートされているページング仮想メ モリ方式を制御する。fctl.GXLが1の場合、SXLフィールドは1でなければならない。そうでない場合、SXLフィールドの正当な値はfctl.GXLフィールドの値と同じである。

SXLが1の場合、以下のルールが適用される：

- 第 1 段階がベアでない場合、IOVA のビット 31 以降のビットが 1 に設定されていれば、元のアクセ ス種別に対応するページフォルトが発生する。
- セカンドステージが Bare でない場合、入力される GPA のビット 33 以降のビットが 1 に設定されていれば、元のアクセスタイプに対応するゲストページフォールトが発生する。

IO ハイパーバイザーのゲストアドレス変換と保護 (iohgatp)

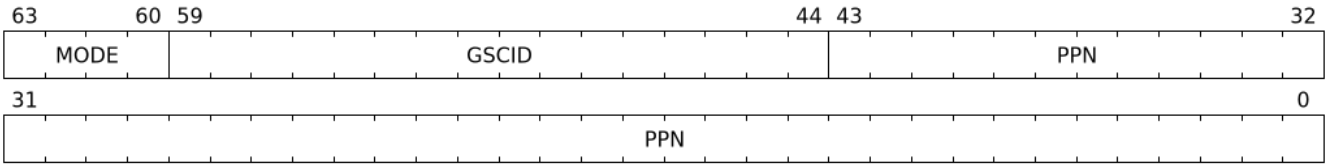


図16.IO ハイパーバイザーのゲストアドレス変換と保護（iohgatp）フィールド

iohgatp フィールドは、ルート・セカンドステージ・ページテーブルの PPN と、ゲスト・ソフトコンテキスト ID (GSCID) によって識別される仮想マシンを保持します。複数のデバイスが共通の第2ステージ・ページ・テーブルを持つVMに関連付けられている場合、ハイパーバイザーはそれぞれのiohgatpに同じGSCIDをプログラムすることが期待されています。MODE フィールドは、セカンドステージのアドレス変換スキームを選択するために使用されます。

第2ステージのページテーブルフォーマットは、Privileged仕様で定義されている通りである。fctl.GXLフィールドは、表2に定義されているように、サポートされているゲスト物理アドレスのアドレス変換スキームを制御する。

iohgatp MODEフィールドは、ページングされた仮想メモリースキームを識別し、そのエンコーディングは以下の通りである：

表2.iohgatp.MODE フィールドのエンコーディング

fctl.GXL=0		
価値	名称	説明
0	裸	翻訳もプロテクションもない。
1-7	-	標準的な使用のために予約されている。
8	Sv39x4	ページベースの41ビット仮想アドレッシング（Sv39の2ビット拡張）。
9	Sv48x4	ページベースの50ビット仮想アドレッシング（Sv48の2ビット拡張）。
10	Sv57x4	ページベースの59ビット仮想アドレッシング（Sv57の2ビット拡張）。
11-15	-	標準的な使用のために予約されている。
fctl.GXL=1		
価値	名称	説明
0	裸	翻訳もプロテクションもない。
1-7	-	標準的な使用のために予約されている。
8	Sv32x4	ページベースの34ビット仮想アドレッシング（Sv32の2ビット拡張）。
9-15	-	標準的な使用のために予約されている。

実装は、iohgatp用に定義されたすべてのモード設定をサポートする必要はない。IOMMUは、システムに統合されたハートのMMUがサポートするモード、またはそのサブセットのみをサポートする必要がある。

iohgatp.PPNによって決定されるルート・ページ・テーブルは16KiBであり、16KiB境界にアライメントされなければならない。

iohgatp の GSCID フィールドはアドレス空間を識別する。2つのDCが参照する第2ステージのページテーブルが同一でないときに、同一のGSCIDが2つのDCに設定された場合、IOMMUが最初のページテーブルと2番目のページテーブルのどちらのPTEを使用するかは予測できない。これらは期待される唯一の動作である。

翻訳属性（ta）

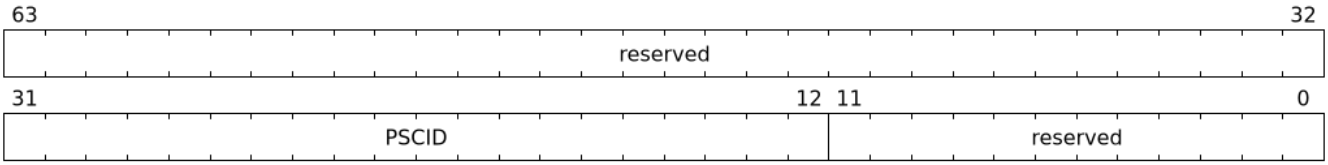


図17.翻訳属性（ta）フィールド

ta の PSCID フィールドは、プロセスのアドレス空間を識別するプロセス・ソフトコンテキスト ID を提供する。PSCID は、アドレス空間ごとのアドレス変換フェンスを容易にする。DC.tc.PDTV が 0 で iosatp.MODE フィールドが Bare でない場合、ta の PSCID フィールドがアドレス空間 ID として使用される。以下の場合

DC.tc.PDTVが1の場合、taのPSCIDフィールドは無視される。

ファーストステージコンテキスト（fsc）

DC.tc.PDTVが0の場合、DC.fscフィールドは、第一段階のアドレス変換とプロテクションのコントロールを提供するiosatpを保持する。

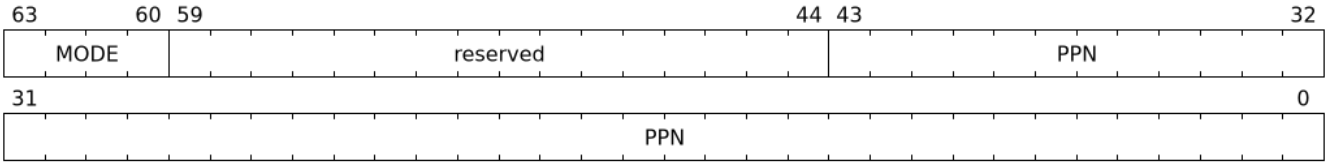


図18.IO Supervisor アドレス変換と保護(iosatp) フィールド

第1ステージのページテーブルフォーマットは、Privileged仕様で定義されている通り

である。DC.tc.SXLフィールドは、サポートされるページド仮想メモリ方式を制御する。

iosatp.MODEは、ページングされた仮想メモリ方式を識別し、表3で定義されているように符号化される。iosatp.PPNフィールドは、第1ステージページテーブルのルートページのPPNを保持する。

第 2 段階のアドレス変換が Bare でない場合、iosatp.PPN はゲスト PPN である。その後、iohgatp によって制御されるゲスト物理アドレス変換プロセスによって、ルートページの GPA がスーパーバイザ物理アドレスに変換されます。

表3.iosatp.MODE フィールドのエンコーディング。

DC.tc.SXL=0		
価値	名称	説明
0	裸	翻訳もプロテクションもない。
1-7	-	標準的な使用のために予約されている。
8	Sv39	ページベースの39ビット仮想アドレッシング。
9	Sv48	ページベースの48ビット仮想アドレッシング。
10	Sv57	ページベースの57ビット仮想アドレッシング。
11-13	-	標準的な使用のために予約されている。
14-15	-	カスタム仕様。
DC.tc.SXL=1		
価値	名称	説明
0	裸	翻訳もプロテクションもない。
1-7	-	標準的な使用のために予約されている。

8	Sv32	ページベースの32ビット仮想アドレッシング。
9-15	-	標準的な使用のために予約されている。

DC.tc.PDTV が 1 の場合、DC.fsc フィールドはプロセス・ディレクトリ・テーブル・ポインタ (pdtp) を保持する。デバイスがprocess_idによって選択された複数のプロセス・コンテキストをサポートしている場合、PDTは、仮想アドレス変換と保護のために、第1段階のページ・テーブルと関連するPSCIDを決定するために使用される。

pdtpフィールドはルートPDTのPPNを保持し、MODEフィールドはPDTのレベル数を決定する。

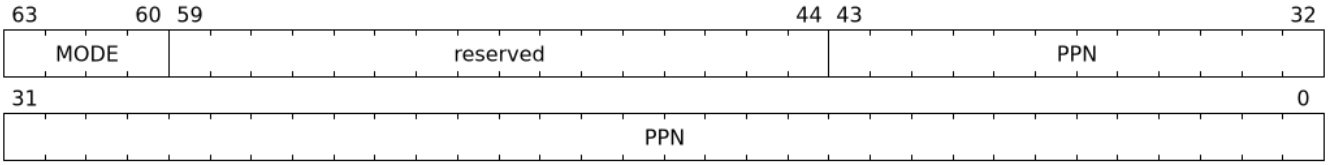


図19. プロセス・ディレクトリ・テーブル・ポインタ（pdtp）フィールド

第 2 段階のアドレス変換が Bare でない場合、pdtp.PPN フィールドはゲスト PPN を保持する。その後、iohgatp によって制御されるゲスト物理アドレス変換プロセスによって、ルート PDT の GPA がスーパーバイザ物理アドレスに変換されます。第2ステージのページテーブルを使用してPDTのアドレスを変換すると、PDTをゲストOSによって割り当てられたメモリに保持できるようになり、ゲストOSがPDTを直接編集して、第1ステージのページテーブルによって識別される仮想アドレス空間をprocess_idに関連付けることができます。

表4.pdtp.MODE フィールドのエンコード

価値	名称	説明
0	裸	第一段階のアドレス変換もプロテクションもない。
1	ピーディーエイト	8ビットのプロセスIDが有効。process_idのビット19:8は0でなければならない。
2	PD17	17ビットのプロセスIDが有効。ディレクトリには2つのレベルがある。ルートPDTページには512エントリ、リーフ・レベルには256エントリがある。process_idのビット19:17は0でなければならない。
3	PD20	20ビットのプロセスIDが有効。ディレクトリには3つのレベルがある。ルートPDTには8エントリがあり、次の非リーフレベルには512エントリがある。リーフレベルには256のエントリーがある。
4-13	-	標準的な使用のために予約されている。
14-15	-	カスタム仕様。

MSI ページテーブルポインタ (msiptp)

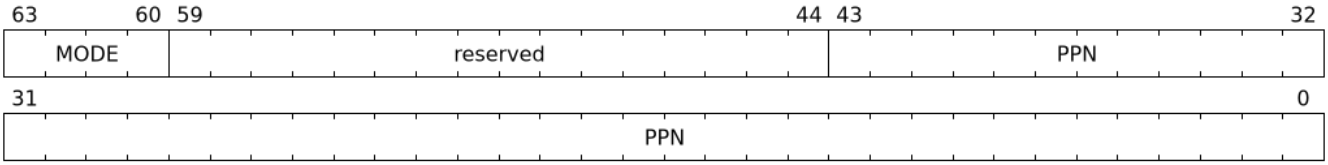


図20.MSIページテーブルポインタ（msiptp）フィールド

msiptp.PPN フィールドは、IMSIC 内のゲスト割り込みファイルに MSI を指示する ために使用されるルート MSI ページ・テーブルの PPN を保持します。MSI ページ・テーブルのフォーマットは、Advanced Interrupt Architecture 仕様で定義されています。

msiptp.MODEフィールドは、MSIアドレス変換スキームを選択するために使用される。

表5.msiptp.MODE フィールドのエンコーディング

価値	名称	説明
0	オフ	MSIアドレスマスクとパターンを使用した仮想割り込みファイルへのアクセスの認識が行われない。
1	フラット	フラットMSIページテーブル
2-13	-	標準的な使用のために予約されている。
14-15	-	カスタム仕様。

MSI アドレスマスク (msi_addr_mask) とパターン (msi_addr_pattern)

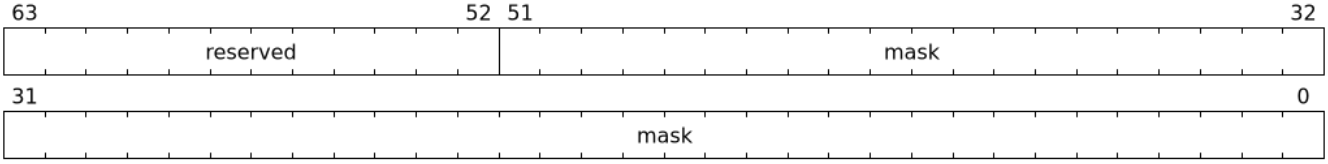


図21.MSIアドレス・マスク（msi_addr_mask）フィールド

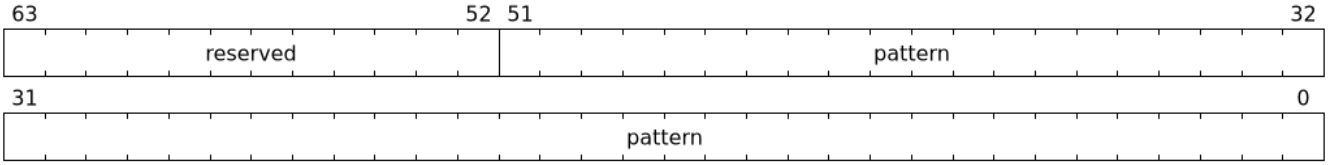


図22.MSIアドレス・パターン（msi_addr_pattern）フィールド

MSI アドレス・マスク (msi_addr_mask) およびパターン (msi_addr_pattern) フィールドは、関連する VM のゲスト物理アドレス空間内の仮想割り込みファイルの 4-KiB ページを識別するために使用されます。デバイスによって行われる着信メモリ・アクセスは、宛先のゲスト物理ページが、指定されたアドレス・マスクのすべてのビット位置で指定されたアドレス・パターンに一致する場合に、仮想割り込みファイルへのアクセスとして認識されます。詳細には、ゲスト物理アドレスAへのメモリ・アクセスは、以下の場合に仮想割込みファイルのメモリ・マップド・ページへのアクセスとして認識されます:

(A >> 12) & ~msi_addr_mask = (msi_addr_pattern & ~msi_addr_mask)

ここで、>> 12は12ビット右シフトを表し、アンパサンド（&）はビットごとの論理ANDを表し、~msi_addr_maskはアドレスマスクのビットごとの論理補数である。

2.1.4. デバイス・コンテキスト設定チェック

DC.tc.V=1のDCは、以下の条件のいずれかが真である場合、misconfiguredとみなされる。設定ミスの場合、停止し、"DDT entry misconfigured" (cause = 259)と報告する。

1. 将来の標準使用のために予約されているビットやエンコーディングが設定されている場合。
2. capabilities.ATSが0で、DC.tc.EN_ATS、またはDC.tc.EN_PRI、またはDC.tc.PRPRが1である。

3. DC.tc.EN_ATSが0、DC.tc.T2GPAが1
4. DC.tc.EN_ATSが0、DC.tc.EN_PRIが1
5. DC.tc.EN_PRIが0、DC.tc.PRPRが1
6. capabilities.T2GPAは0、DC.tc.T2GPAは1である。
7. DC.tc.T2GPAは1、DC.iohgatp.MODEはBareである。

8. DC.tc.PDTVが1であり、DC.fsc.pdtp.MODEがサポートされていないモードである。
    a. capabilities.PD20が0で、DC.fsc.ptp.MODEがPD20の場合
    b. capabilities.PD17は0、DC.fsc.ptp.MODEはPD17
    c. capabilities.PD8が0で、DC.fsc.ptp.MODEがPD8
9. DC.tc.PDTVが0であり、かつDC.fsc.iosatp.MODEエンコーディングが表3によって決定される有効なエンコーディングではないこと。
10. DC.tc.PDTVが0、DC.tc.SXLが0 DC.fsc.iosatp.MODEがサポートされているモードではない
    a. capabilities.Sv39は0、DC.fsc.iosatp.MODEはSv39
    b. capabilities.Sv48は0、DC.fsc.iosatp.MODEはSv48
    c. capabilities.Sv57は0、DC.fsc.iosatp.MODEはSv57
11. DC.tc.PDTVが0、DC.tc.SXLが1 DC.fsc.iosatp.MODEがサポートされているモードではない
    a. capabilities.Sv32 は 0 で、DC.fsc.iosatp.MODE は Sv32 です。
12. DC.tc.PDTVが0、DC.tc.DPEが1
13. DC.iohgatp.MODEエンコーディングは、表2によって決定される有効なエンコーディングではありません。
14. fctl.GXLが0であり、DC.iohgatp.MODEがサポートされていないモードである。
    a. capabilities.Sv39x4は0、DC.iohgatp.MODEはSv39x4
    b. capabilities.Sv48x4は0、DC.iohgatp.MODEはSv48x4
    c. capabilities.Sv57x4は0、DC.iohgatp.MODEはSv57x4
15. fctl.GXLが1であり、DC.iohgatp.MODEがサポートされていないモードである。
    a. capabilities.Sv32x4が0、DC.iohgatp.MODEがSv32x4
16. capabilities.MSI_FLATが1であり、DC.msiptp.MODEがOffでもFlatでもない。
17. DC.iohgatp.MODEがBareではなく、DC.iohgatp.PPNによって決定されるルートページ表が16-KiB境界にアライメントされていない。
18. capabilities.AMO_HWADが0で、DC.tc.SADEまたはDC.tc.GADEが1である。
19. capabilities.ENDが0かつfctl.BE != DC.tc.SBE
20. DC.tc.SXLの値が正当な値でない。fctl.GXLが1の場合、DC.tc.SXLは1でなければならない。fctl.GXLが0で書き込み可能な場合、DC.tc.SXLは0でも1でもよい。fctl.GXLが0で書き込み可能でない場合、DC.tc.SXLは0でなければならない。

21. DC.tc.SBEの値が正当な値でない。fctl.BEが書き込み可能な場合、DC.tc.SBEは0または1である。もしfctl.BEが

が書き込み可能でない場合、DC.tc.SBEはfctl.BEと同じでなければならない。

一部のDCフィールドは、スーパーバイザ物理アドレスまたはゲスト物理 アドレスを保持する。一部の実装では、DCの位置を特定するときに、capabilities.PASなどで決定されるサポー ト範囲よりもスーパバイザ物理アドレスの幅が広くないなど、アドレスの有効性を検証 することがある。このような実装では、"DDT entry misconfigured" (cause = 259) フォルトが発生する可能性がある。

他の実装では、これらのフィールドによって参照されるデータ構造にアクセスする必要がある場合にのみ、このようなアドレスが無効であることを検出する。そのような実装は

は、アクセスの過程でアクセス違反の故障を検出する可能性がある。

2.2. プロセス・ディレクトリ・テーブル (PDT)

PDTは1、2、または3レベルの基数木で、プロセス・ディレクトリ・インデックス（PDI）ビットの process_id。

以下の図にPDT基数ツリーを示す。ルート・プロセス・ディレクトリのページ番号は、デバイス・コンテキストのプロセス・ディレクトリ・テーブル・ポインタ（pdtp）フィールドを使用して特定される。各非リーフ（NL）エントリは、次のレベルのプロセス・ディレクトリ・テーブルのPPNを提供する。リーフの process-directory-tableエントリはプロセスコンテキスト（PC）を保持する。

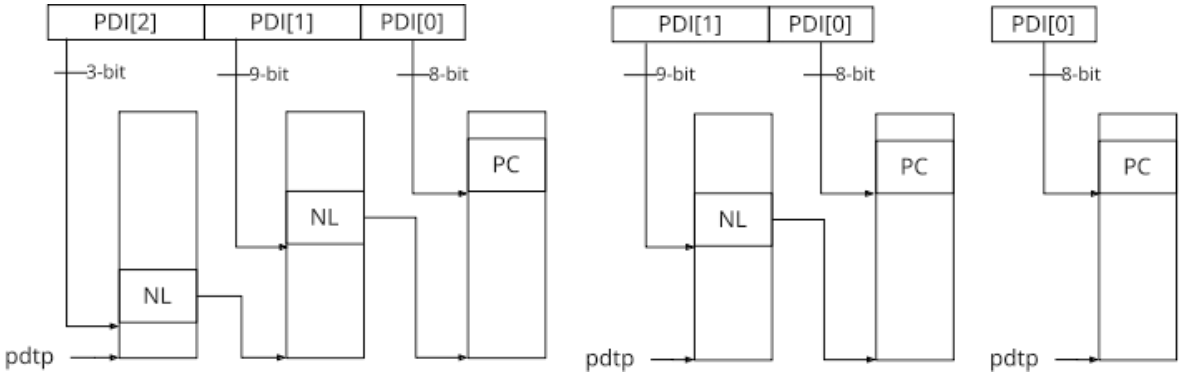


図23.3レベル、2レベル、1レベルのプロセス・ディレクトリ

2.2.1. ノンリーフPDTエントリー

有効な(V==1)非リーフPDTエントリは、次レベルPDTのPPNを保持する。

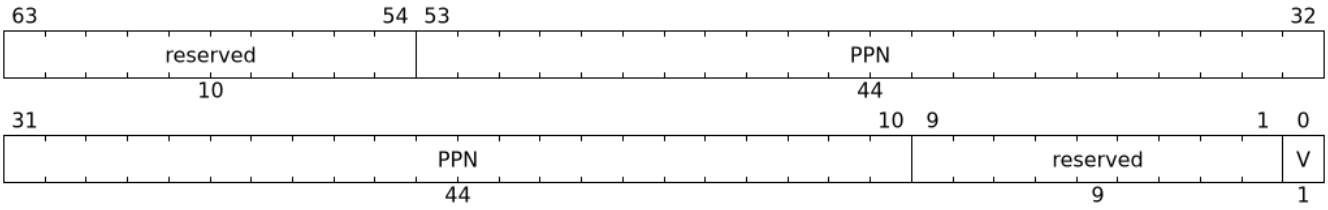


図24.非リーフ・プロセス・ディレクトリ・テーブル・エントリ

2.2.2. リーフPDTエントリー

リーフPDTページはPDI[0]でインデックスされ、16バイトのプロセスコンテキスト（PC）を保持する。

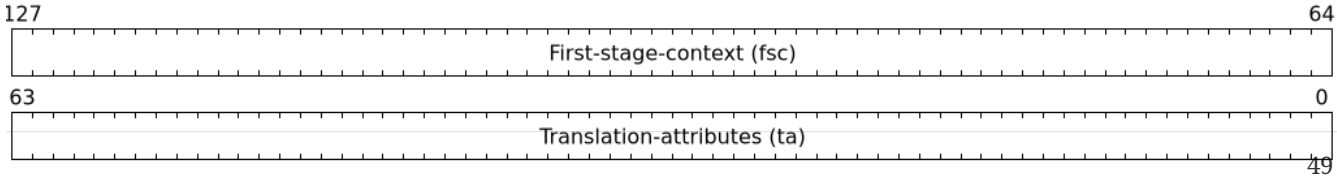


図25. プロセス・コンテキスト

PCは2つの64ビット・ダブルワードとして解釈される。の各ダブルワードのバイト順序は、次のとおりである。

メモリ、リトルエンディアンまたはビッグエンディアンは、DC.tc.SBE によって決定されるエンディアンである。IOMMU は、PC フィールドを任意の順序で読み出すことができる。

2.2.3. プロセス・コンテキスト・フィールド

翻訳属性（ta）

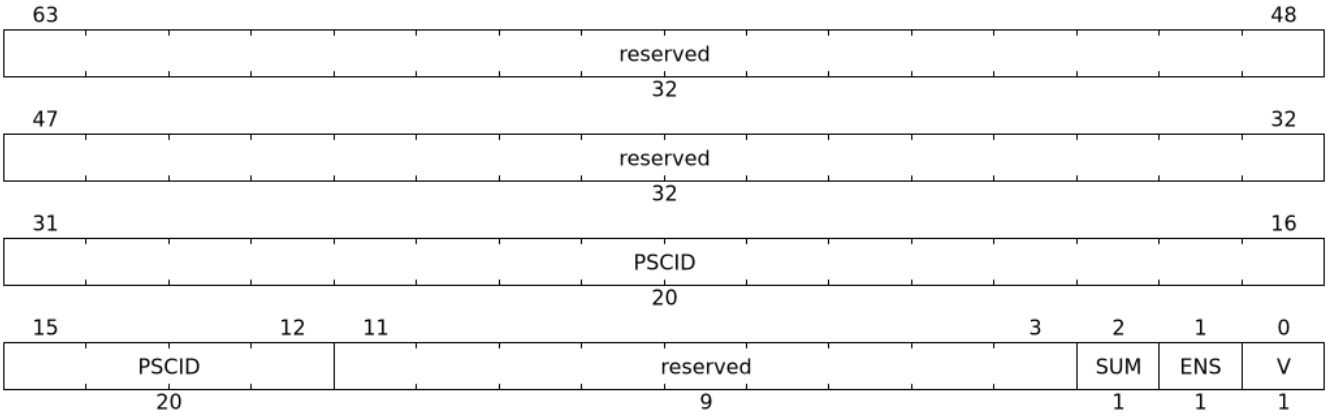


図26.翻訳属性（ta）フィールド

Vビットが1の場合、PCは有効である。Vビットが0の場合、PCの他のビットはすべてdon't careであり、ソフトウェアが自由に使用できる。

Enable-Supervisory-access（ENS）が1の場合、スーパーバイザ権限を要求するトランザクションは、このprocess_idで許可される。

ENSが1のとき、SUM（permit Supervisor User Memory access）ビットは、スーパーバイザ特権トランザクションが仮想メモリにアクセスする特権を変更する。SUM が 0 の場合、PTE の U ビットが 1 に設定されたページへのスーパーバイザ特権トランザクションは許可されません。

ENS が 1 の場合、スーパーバイザ特権トランザクションは、次のようにマップされたページに実行意図を持って読み込む。
PTEのUビットが1にセットされている場合は、SUMの値に関係なく禁止される。

ソフトウェアが割り当てるプロセスソフトコンテキストID（PSCID）は、第1ステージアドレス変換がBareでない場合、第1ステージページテーブルによって識別されるプロセスのアドレス空間IDとして使用される。

ファーストステージコンテキスト（fsc）

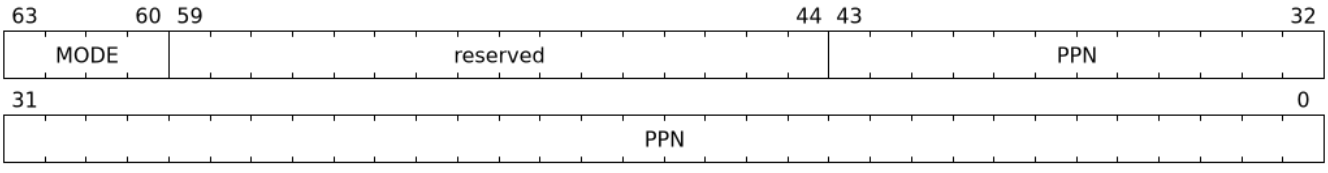


図27. プロセス第一段階のコンテキスト

PC.fscフィールドは、第一段階のアドレス変換とプロテクションのコントロールを提供する。

PC.fsc.MODEは、第1段階のページド仮想メモリ方式を決定するために使用され、そのエンコーディングは表3に定義されている通りである。DC.tc.SXL フィールドは、サポートされるページド仮想メモリ方式を制御する。PC.fsc.MODEがBareでない場合、PC.fsc.PPNフィールドは第1段ページテーブルのルートページのPPNを保持する。

第2段アドレス変換がBareでない場合、PC.fsc.PPNフィールドは第1段ページテーブルのルートのゲストPPNを保持する。第1ステージのページテーブルエントリのアドレスは、DC.iohgatpによって制御されるゲスト物理アドレス変換プロセスによって、スーパーバイザ物理アドレスに変換されます。このため、ゲストOSは、デバイスによるメモリのサブセットへのアクセスを制限し、デバイスのアクセスに対するパーミッションを指定するために、第1ステージページテーブルを直接編集することができます。

PC.ta.PSCID はアドレス空間を識別する。2つのPCが参照するページテーブルが同一でないときに、同一のPSCIDが2つのPCに設定された場合、IOMMUが最初のページテーブルと2番目のページテーブルのどちらのPTEを使用するかは予測できない。これらは期待される唯一の動作である。

2.2.4. プロセス・コンテキスト構成のチェック

PC.ta.V=1のPCは、以下の条件のいずれかが真である場合、misconfiguredとみなされる。もしmisconfiguredなら、停止して "PDT entry misconfigured"(原因=267)と報告する。

1. 将来の標準使用のために予約されているビットやエンコーディングが設定されている場合
2. PC.fsc.MODEエンコーディングが、表3によって決定される有効なものではない
3. DC.tc.SXLが0であり、PC.fsc.MODEがサポートされているモードではない
    a. capabilities.Sv39が0でPC.fsc.MODEがSv39の場合
    b. capabilities.Sv48が0で、PC.fsc.MODEがSv48の場合
    c. capabilities.Sv57が0でPC.fsc.MODEがSv57の場合
4. DC.tc.SXLが1で、PC.fsc.MODEがサポートされているモードではない
    a. capabilities.Sv32が0でPC.fsc.MODEがSv32の場合

一部のPCフィールドは、スーパバイザ物理アドレスまたはゲスト物理アドレスを保持する。例えば、スーパバイザ物理アドレスが、PC の位置を特定するときに capabilities.PAS などで決定されるサポートされるアドレスよりも広くない場合などである。このような実装では、"PDT entry misconfigured" (cause = 267) フォルトが発生する可能性がある。

他の実装では、これらのフィールドによって参照されるデータ構造にアクセスする必要がある場合にのみ、このようなアドレスが無効であることを検出する。そのような実装では、アクセスを行う過程でアクセス違反フォールトを検出することがある。

2.3. IOVAの翻訳プロセス

IOVA を変換するプロセスでは、ハードウェア ID（device_id、process_id）を使用してデバイス・コンテキスト（Device-Context）とプロセス・コンテキスト（Process-Context）を特定する。デバイス・コンテキストとプロセス・コンテキストは、ページ・テーブルのルート PPN、PSCID、GSCID、およびアドレス変換と保護プロセスに影響するその他の制御パラメータを提供する。アドレス変換キャッシュ（セクション 2.8）が実装されている場合、変換プロセスはGSCIDとPSCIDを使用して、キャッシュされた変換をアドレス空間に関連付けることができる。

IOVAの翻訳プロセスは以下の通り：

1. ddtp.iommu_mode == Off の場合は停止し、"All inbound transactions disallowed" (cause = 256) と報告する。
2. ddtp.iommu_mode==Bareで、以下の条件のいずれかが成立すれば、停止し、"Transaction type disallowed"(原因=260)を報告する。
    a. トランザクションタイプは、Translatedリクエスト（リード、ライト/AMO、リードフォーエグゼキュート）、またはPCIe ATS Translationリクエストである。
3. capabilities.MSI_FLAT が 0 の場合、IOMMU はベースフォーマットのデバイスコンテキストを使用する。DDI[0]を
   は device_id[6:0]、DDI[1]は device_id[15:7]、DDI[2]は device_id[23:16]である。
4. capabilities.MSI_FLAT が 1 の場合、IOMMU は拡張フォーマットデバイスコンテキストを使用する。DDI[0]を
   はdevice_id[5:0]、DDI[1]はdevice_id[14:6]、DDI[2]はdevice_id[23:15]である。
5. もし device_id が IOMMU モードでサポートされるものより広い場合、以下のチェックにより判断され、停止し、"Transaction type disallowed" (cause = 260)を報告する。
    a. ddtp.iommu_modeが2LVLで、DDI[2]が0ではない
    b. ddtp.iommu_modeが1LVLで、DDI[2]が0でないか、DDI[1]が0でない。
6. セクション2.3.1で指定されているように、device_idを使用してデバイスコンテキスト（DC）を特定する。
7. 以下の条件のいずれかが成立する場合、停止し、「トランザクションの種類が不許可」（原因 = 260）と報告する。
    a. トランザクションタイプがTranslatedリクエスト（リード、ライト/AMO、リードフォーエグゼキュート）であるか、PCIe ATS Translationリクエストであり、DC.tc.EN_ATSが0である。
    b. トランザクションが有効なprocess_idを持ち、DC.tc.PDTVが0である。
    c. トランザクションが有効なprocess_idを持ち、DC.tc.PDTVが1で、process_idがpdtp.MODEでサポートされるものより広い。
    d. IOMMUはトランザクションタイプをサポートしていない。
8. リクエストがTranslated requestであり、DC.tc.T2GPAが0であれば、トランスレーション処理は完了する。ステップ20に進む。
9. 要求が翻訳済み要求であり、DC.tc.T2GPA が 1 である場合、IOVA は GPA である。次のページ表情報でステップ17に進む：
    a. AをIOVAとする（IOVAはGPA）。

b. iosatp.MODEをBareとする。

i. ファーストステージが Bare の場合、PSCID の値は使用されない。

c. DC.iohgatpフィールドの値をiohgatpとする。

10. DC.tc.PDTVが0に設定されている場合は、次のページ表情報を使用してステップ17に進む：

a. DC.fsc.MODEフィールドの値をiosatp.MODEとする。

b. DC.fsc.PPNフィールドの値をiosatp.PPNとする。

c. DC.ta.PSCIDフィールドの値をPSCIDとする。

d. DC.iohgatpフィールドの値をiohgatpとする。

11. DPEが1であり、トランザクションに関連するprocess_idがない場合、process_idを次のようにする。

デフォルト値は0。

12. DPEが0であり、トランザクションに関連するprocess_idが存在しない場合は、以下のページテーブル情報を使ってステップ17に進む：
    a. iosatp.MODEをBareとする。
        i. ファーストステージが Bare の場合、PSCID の値は使用されない。
    b. DC.iohgatpフィールドの値をiohgatpとする。
13. DC.fsc.ptp.MODE=Bareの場合は、次のページ表情報を使ってステップ17に進む：
    a. iosatp.MODEをBareとする。
        i. ファーストステージが Bare の場合、PSCID の値は使用されない。
    b. DC.iohgatpフィールドの値をiohgatpとする。
14. セクション2.3.2で指定されているように、プロセスコンテキスト（PC）を見つける。
15. 以下の条件のいずれかが成立する場合、停止し、"トランザクションタイプは不許可"(原因=260)と報告する。
    a. トランザクションがスーパーバイザ特権を要求しているが、PC.ta.ENS が設定されていない。
16. 次のページ表情報を使ってステップ17に進む：
    a. PC.fsc.MODEフィールドの値をiosatp.MODEとする。
    b. PC.fsc.PPNフィールドの値をiosatp.PPNとする。
    c. PC.ta.PSCIDフィールドの値をPSCIDとする。
    d. DC.iohgatpフィールドの値をiohgatpとする。
17. トランザクションによってアクセスされる GPA を決定するために、RISC-V Privileged 仕様書[3]のセクション「2 段階アドレス変換」で規定される処理を使用する。第1段階のアドレス変換処理によって障害が検出された場合、その障害を停止して報告する。変換処理が正常に完了した場合、変換された GPA を A とする。
18. MSIページテーブルを使用したMSIアドレス変換が有効になっている場合（すなわち、DC.msiptp.MODE != Off）、セクション2.3.3で指定されたMSIアドレス変換処理が起動される。GPA A が仮想割り込みファイルのアドレスであると判断されない場合、処理はステップ 19 に進む。MSI アドレス変換処理によってフォルトが検出された場合は、停止してフォルトを報告し、それ以外の場合はステップ 20 で処理を続行する。
19. RISC-V特権仕様書[3]のセクション「2段階アドレス変換」で規定される第2段階アドレス変換処理を使用して、GPA Aを変換し、トランザクションによってアクセスされるSPAを決定する。アドレス変換プ

ロセスによってフォルトが検出された場合、停止してフォルトを報告する。

20. 翻訳作業完了

セカンドステージのPTEでUビットをチェックする場合、そのトランザクションはスーパーバイザ特権を要求していないものとして扱われる。

翻訳プロセスがフォールトを報告し、リクエストが未翻訳リクエストまたは翻訳済みリ クエストである場合、IOMMUはIOブリッジにトランザクションを中止するよう要求する。IO ブリッジでフォールトが発生したトランザクションを処理するためのガイドラインは、セクション 7.3 に記載されている。フォルトは以下の可能性がある。

セクション3.2で規定される故障/イベント報告メカニズムおよび故障記録フォーマットを使用して報告される。

フォルトがPCIe ATS変換要求によって検出された場合、IOMMUはソフトウェアにフォルトを報告したり、アボートを引き起こしたりする代わりに、PCIeプロトコルで定義された応答を提供することができる。フォールトが発生した PCIe ATS 変換要求の処理については、セクション 2.6 に規定されている。

2.3.1. デバイスコンテキストの場所を特定するプロセス

device_idを使用してトランザクション用のDevice-contextを見つけるプロセスは以下の通り：

1. ddtp.PPN×2^{12} 、i = LEVELS - 1とする。ddtp.iommu_modeが3LVLのとき、LEVELSは3である。ddtp.iommu_modeが2LVLのとき、LEVELSは2である。ddtp.iommu_modeが1LVLのとき、LEVELSは1である。
2. i == 0ならステップ8に進む。
3. ddteをアドレスa + DDI[i]×8の8バイトの値とする。ddteへのアクセスがPMAまたはPMPチェックに違反する場合、停止して「DDTエントリロードアクセスフォルト」（原因=257）を報告する。
4. ddteアクセスがデータ破損（ポイズン・データ）を検出した場合、停止して「DDTデータ破損」（原因＝268）を報告する。
5. ddte.V == 0 の場合、停止して「DDT エントリが有効ではありません」（原因 = 258）と報告する。
6. 将来の標準使用のために予約されているビットやエンコーディングがddte内に設定されている場合、停止して「DDTエントリが誤って設定されました」(cause = 259)と報告する。
7. i = i - 1とし、a = ddte.PPN x 2^{12} とする。 ステップ2に進む。
8. アドレス a + DDI[0] * DC_SIZE のバイト数を DC とする。capabilities.MSI_FLAT が 1 の場合、DC_SIZE は 64 バイトである。DCへのアクセスがPMAまたはPMPチェックに違反する場合は、停止して「DDTエントリ・ロード・アクセス・フォールト」(原因=257)を報告する。DCアクセスがデータ破損（毒データ）を検出した場合、停止して "DDTデータ破損"（原因=268）を報告する。
9. DC.tc.V == 0 の場合、停止し、"DDT entry not valid" (cause = 258) と報告する。
10. セクション2.1.4に概説されている規則によってDCが誤って設定されている場合、停止して「DDTエントリが誤って設定されています」(原因=259)と報告する。
11. デバイス・コンテキストの位置が正常に特定されました。

2.3.2. プロセスコンテキストの場所を特定するプロセス

デバイスコンテキストは PDT ルートページ PPN（pdtp.ppn）を提供する。DC.iohgatp.mode が Bare で

ない場合、pdtp.PPN と pdte.PPN はゲスト物理アドレス（GPA）であり、DC.iohgatp が指す第 2 段ページテーブルを使用してスーパーバイザ物理アドレス（SPA）に変換する必要があります。PDTへのメモリ・アクセスは、セカンド・ステージによって暗黙のリード・メモリ・アクセスとして扱われる。

process_idを使用してトランザクションのProcess-contextを検索するプロセスは以下のとおりである：

1. pdtp.PPN×2^{12} 、i = LEVELS - 1とする。pdtp.MODEがPD20のとき、LEVELSは3である。このとき pdtp.MODEがPD17の場合、LEVELSは2。pdtp.MODE が PD8 のとき、LEVELS は 1 である。
2. DC.iohgatp.mode != Bare ならば、a は GPA である。暗黙のメモリアクセスとして a を SPA に変換するプロセスを呼び出す。a の第2段アドレス変換中にフォルトが発生した場合、第2段アドレス変換プロセスによって検出されたフォルトを停止して報告する。変換されたaは

後続のステップで使用される。

3. i == 0ならステップ9に進む。
4. pdteをアドレスaの8バイトの値+PDI[i]×8とする。pdteへのアクセスがPMAまたはPMPチェックに違反する場合、停止し、「PDTエントリロードアクセスフォルト」(原因=265)を報告する。
5. pdteアクセスがデータ破損（毒入りデータ）を検出した場合、停止して「PDTデータ破損」（原因=269）を報告する。
6. pdte.V == 0なら、停止し、"PDT entry not valid" (cause = 266)と報告する。
7. 将来の標準使用のために予約されているビットやエンコーディングがpdte内に設定されている場合、停止して "PDT entry misconfigured"(原因=267)と報告する。
8. i = i - 1とし、a = pdte.PPN x 2^{12} とする。 ステップ2に進む。
9. アドレス a の 16 バイトの値 + PDI[0] x 16 を PC とする。PCへのアクセスがPMAまたはPMPチェックに違反する場合、停止して "PDT entry load access fault"(原因=265)を報告する。PCアクセスがデータ破損（ポイズン・データ）を検出した場合、停止して "PDTデータ破損"（原因=269）を報告する。
10. PC.ta.V == 0ならば、停止し、"PDT entry not valid" (cause = 266)と報告する。
11. セクション2.2.4で概説された規則によりPCが誤って設定されている場合、停止して「PDTエントリが誤って設定されています」（原因=267）と報告する。
12. Process-contextが正常に配置されました。

2.3.3. MSIのアドレスを変換するプロセス

I/Oデバイスがゲスト OS によって直接設定される場合、デバイスからのMSIは、ゲストOSの仮想マシン内の仮想IMSICをターゲットとし、実マシンには不適切で安全でないゲスト物理アドレスを使用することが予想される。IOMMUは、このようなデバイスからの特定の着信書き込みをMSIとして認識し、実マシン用に必要に応じて変換しなければならない。

変換を必要とする1つのデバイスから発信されるMSIは、1つのRISC-V仮想マシン内で実行される1つのゲストOSによってデバイスに設定されていると予想されます。VM自体がRISC-V Advanced Interrupt Architecture [2]に準拠していると仮定すると、MSIは仮想IMSICの割り込みファイルのメモリ・マップされたレジスタに書き込むことで、VM内の仮想ハートに送信されます。これらの仮想割り込みファイルはそれぞれ、VM のゲスト物理アドレス空間内の個別の 4 KB ページを占有します。このため、ゲスト物理アドレスへの書き込みが、VM内の仮想IMSICの割り込みファイルが占有するページへの書き込みであれば、仮想ハートへのMSIとして認識できます。

MSIアドレス変換がサポートされている場合（5.3節のcapabilities.MSI_FLAT）、入力されたIOVAを仮想割込

みファイルのアドレスとして識別し、MSIページテーブ ルを使用してアドレスを変換するプロセスは以下の通りである：

1. AをGPAとする。
2. セクション2.3.1に概説されているプロセスを使用して、デバイスのdevice_idを 使用して特定されたデバイスコンテキストをDCとする。
3. アドレスAがセクション2.1.3.6で指定された仮想割り込みファイルへのアクセスかどうかを判断する。

4. アドレスが仮想割り込みファイルのものであると判断されない場合は、この処理を停止し、代わりに通常の変換データ構造を使用してアドレス変換を行います。
5. I = extract(A >> 12, DC.msi_addr_mask) として、A から割り込みファイル番号 I を抽出する。ビット抽出関数extract(x, y)は、マスクyの同じ位置にある一致するビットがゼロであるxのビットをすべて破棄し、xの残りのビットを結果の最下位に連続的に詰め、xと同じビット順序を維持し、結果の最上位にあるその他のビットをゼロで埋めます。例えば、xとyのビットが次のような場合である：
   - x = a b c d e f g h
   - y = 1 0 1 0 0 1 1 0
   - この場合、extract(x, y)の値はビット0 0 0 0 a c f gとなる。
6. mを(DC.msiptp.PPN x 2^{12})とする。
7. アドレス（m｜ （I×16））の16バイトの値をmsipteとする。msipteへのアクセスがPMAまたはPMPチェックに違反する場合、停止して「MSI PTEロード・アクセス・フォールト」（原因=261）を報告する。
8. msipteアクセスがデータ破損（毒入りデータ）を検出した場合、停止して「MSI PTデータ破損」（原因=270）を報告する。
9. msipte.V == 0 なら、停止して「MSI PTE not valid」（原因 = 262）と報告する。
10. msipte.C==1の場合、PTEを解釈するための更なる処理は、実装で定義される。
11. msipte.C==0であれば、以降のステップの概略を説明する。
12. msipte.M == 0またはmsipte.M == 2の場合、停止して「MSI PTE誤設定」（原因= 263）を報告する。
13. msipte.M==3の場合、PTEは基本トランスレートモードであり、トランスレート処理は以下のようになる：
    a. 将来の標準使用のために予約されているビットまたはエンコーディングが msipte 内に設定されている場合、停止し、"MSI PTE misconfigured" (cause = 263) と報告する。
    b. msipte.PPN << 12 | A[11:0]として変換されたアドレスを計算する。
14. msipte.M == 1の場合、PTEはMRIFモードであり、変換処理は以下のようになる：
    a. capabilities.MSI_MRIF == 0 の場合、停止し、「MSI PTE misconfigured」（原因 = 263）と報告する。
    b. 将来の標準使用のために予約されているビットまたはエンコーディングが msipte 内に設定されている場合、停止し、"MSI PTE misconfigured" (cause = 263) と報告する。
    c. 宛先MRIFのアドレスはmsipte.MRIF_Address[55:9] * 512である。

d. 通知MSIの宛先アドレスはmsipte.NPPN << 12である。

e. NIDを(msipte.N10 << 10) | msipte.N[9:0]とする。通知MSIのデータ値は11ビットのNIDである。値を32ビットにゼロ拡張したもの。

15. このプロセスで決定される変換に関連するアクセス許可は、R=W=U=1、X=0の通常のRISC-Vセカンドステージ PTEと同等である。セカンドステージPTEと同様に、Uビットをチェックする場合、トランザクションはスーパーバイザ特権を要求していないものとして扱われる。

16. トランザクションが未翻訳または翻訳された実行形式の読み出しである場合、停止し、「命令アク セス・フォルト」（原因 = 1）を報告する。

17. MSIアドレス変換処理が完了した。

MRIFモードでは、受信MSIをデスティネーションMRIFに格納し、通知MSIを生成する動作がAdvanced Interrupt Architecture Specificationに定義されている。これらの動作はIOMMU自身が実行してもよいし、IOMMUが変換要求に応答してデスティネーションMRIFアドレス、通知MSIアドレス、通知MSIデータ値をI/Oブリッジに提供し、I/Oブリッジがその動作を実行してもよい。

2.4. IOMMUによるPTEアクセス（A）とダーティ（D）の更新

capabilities.AMO_HWAD が 1 のとき、IOMMU は PTE の A ビットと D ビットのアトミックな更新をサポートする。第 2 段 PTE の A および D ビットの更新が有効な場合（DC.tc.GADE=1）、および/または第 1 段 PTE の A および D ビットの更新が有効な場合（DC.tc.SADE=1）、以下の規則が適用される：

1. IOMMUによるAおよび/またはDビットの更新は、有効性、許可チェック、および原子性について、Privileged仕様で規定された規則に従わなければならない。
2. PTE更新は、IOMMUから提供された変換済みアドレスを使用するメモリアクセスがグローバルに可視になる前に、グローバルに可視になっていなければならない。具体的には、ATS変換完了でデバイスに変換済みアドレスが提供される場合、変換済みアドレスを使用するデバイスからのメモリアクセスがグローバルに可視になる前に、PTE更新がグローバルに可視になっていなければならない。

A ビットと D ビットが IOMMU によってクリアされることはない。スーパバイザ・ソフトウェアが、メモリ・ページをセカンダリ・ストレージにスワップしない場合や、ページがI/Oスペースのマッピングに使用されている場合など、アクセス済みビットおよび/またはダーティ・ビットを使用しない場合は、PTEでこれらを1に設定してパフォーマンスを向上させる必要があります。

2.5. 仮想アドレス変換処理による不具合

RISC-V Privileged仕様[3]で規定された2段階のアドレス変換中に検出された障害は、IOVA変換プロセスを停止させ、検出された障害を報告させる。

2.6. PCIe ATS変換リクエスト処理

ATS(参考文献[1])の翻訳リクエストで設定エラーが発生した場合、 リクエスタに対してCompleter Abort(CA)応答が返される。以下の原因コードはこのカテゴリーに属する：

- 命令アクセスエラー（原因 = 1）
- リード・アクセス・フォールト（原因 = 5）
- ライト/AMO アクセス・フォールト（原因 = 7）
- MSI PTE ロードアクセスフォールト（原因 = 261）
- MSI PTE の設定ミス（原因 = 263）
- PDT エントリロードアクセス障害（原因 = 265）
- PDTエントリの設定ミス（原因=267）

恒久的なエラーがある場合、またはATSトランザクションが無効になっている場合、UR（Unsupported Request）応答が生成される。以下の原因コードはこのカテゴリーに属する：

- すべてのインバウンドトランザクションが拒否された（原因 = 256）
- DDT エントリロードアクセスフォルト（原因 = 257）
- DDT エントリが無効（原因 = 258）
- DDTエントリの設定ミス（原因=259）
- トランザクションタイプが許可されない（原因 = 260）

以下の原因で翻訳が完了できなかった場合、RビットとWビットが0に設定されたSuccess Responseが生成される。これらのエラーでは、フォールトキューにフォールトは記録されない。このような完了で返される翻訳済みアドレスはUNSPECIFIEDである。

- 命令ページフォルト（原因 = 12）
- 読み取りページ障害（原因 = 13）
- ライト/AMO ページフォルト（原因 = 15）
- 命令ゲストページフォルト（原因=20）
- ゲストページの読み取りエラー（原因 = 21）
- ライト/AMO ゲスト・ページ・フォールト（原因 = 23）
- PDTエントリが有効でない（原因=266）
- MSI PTE が有効でない（原因 = 262）

変換要求が、「Privilege Mode Requested」フィールドが0に設定された PASIDを持つか、要求がPASIDを持たない場合、その要求は特権メモ リをターゲットにしていない。メモリがユーザーモードでアクセス可能かどうかを示すUビットが0であれば、RビットとWビットが0に設定されたSuccess応答が生成される。

変換要求が「Privilege Mode Requested」フィールドが1に設定された PASIDを持つ場合、要求は特権メモリをターゲットにしている。ページがユーザーモードにアクセス可能かどうかを示すUビットが1であり、プロセスコンテキストのtaフィールドのSUMビットが0である場合、RビットとWビットが0に設定されたSuccess応答が生成される。

もし翻訳が成功裏に完了できたが、要求されたパーミッションが存在しない場合（実行は要求されたが実行パーミッションがない；書き込みは要求されず、書き込みパーミッションもない；読み取りパーミッションもない）、拒否されたパーミッション（R、W、X）を0に設定し、他のパーミッションビットをペー

ジテーブルから決定された値に設定して、Success応答が返される。Xパーミッションは、Rパーミッションも許可されている場合にのみ許可される。PCIeでは、実行パーミッションが付与されている場合、読み取りパーミッションが付与される必要があるため、実行のみのトランスレーションはPCIe ATSと互換性がない。

ATS翻訳リクエストに対してSuccess応答が生成された場合、その応答がアクセスが許可されなかった、または一部のパーミッションが拒否されたことを示す場合でも、フォールト/イベント報告メカニズムを通じてソフトウェアにフォールトレコードが報告されることはない。

変換要求が、セクション2.1.3.6で定義された規則を使用してMSIアドレスであると判断された アドレスを持つが、MSI PTEがMRIFモードに設定されている場合、Success応答は次のようになる。

レスポンスのUビットが1にセットされることで、デバイスは、暗黙の4KiBメモリ範囲にアクセスするために、翻訳されていないリクエストのみを使用しなければならないことを指示する。

MSI PTE が MRIF モードで構成されている場合、データ値 D の MSI 書き込みは、IOMMU が MRIF 内の割り込み ID D の割り込み待ちビットを設定する必要がある。MRIF モードの MSI PTE を介してマップされるデバイスから GPA への変換要求は、変換されたアドレスを受信する資格がない。これは、返された応答の「Untranslated Access Only」（U）フィールドを 1 に設定することで達成される。

ATS翻訳リクエストに対してSuccess応答が生成された場合、Priv、N、CXL.io、Global、およびAMAフィールドの設定は以下のとおりである：

- リクエストにPASIDがない場合、ATS翻訳完了のPrivフィールドは常に0に設定される。PASIDが存在する場合、Privフィールドは、提供されるパーミッション がリクエストで示される特権モードに対応するため、「Privilege Mode Requested」フィールドの値に設定される。
- ATS翻訳完了のNフィールドは常に0に設定される。デバイスは、翻訳されたリクエス トにNo-snoopフラグが設定されるべきかどうかを判断するために、他の手段を使 うことができる。
- グローバルフィールドは、翻訳が正常に完了し、リクエストにPASIDが存在 した場合、第一段階のページテーブルから決定された値に設定される。MSIアドレス変換を含む他のすべての場合、このフィールドは0に設定される。
- 要求デバイスがCXLデバイスでない場合、CXL.ioは0に設定されます。
- 依頼する装置がCXLタイプ1またはタイプ2の場合
  - アドレスがMSIであると判断された場合、CXL.ioビットは1にセットされる。
  - もしデバイスコンテキストでT2GPAが1であれば、CXL.ioビットは1に設定される。
  - メモリタイプがPMAの場合、このビットの設定はUNSPECIFIEDとなる。Svpbmt拡張がサポートされていない場合、このビットの設定はUNSPECIFIEDである。
  - それ以外の場合、このビットの設定はUNSPECIFIEDである。
- AMAフィールドはデフォルトで000bに設定されている。IOMMUは、他のエンコーディングを提供する実装固有の方法をサポートするかもしれない。

IOブリッジは、変換されたアドレスのPMAに基づいて、ATS変換完了のCXL.ioビットを上書きしてもよい。他の実装では、CXL.ioビットを設定するために、変換されたアドレスのPMAを決定する実装定義の方法を提供することができる。

T2GPAを1に設定した場合、CXLタイプ1またはタイプ2デバイスは、PCIe ATS変換要求に

応答して返される変換済みアドレスでタグ付けされたキャッシュを実装するためにCXL.cacheプロトコルを使用するため、CXLタイプ1またはタイプ2デバイスと互換性がない可能性がある。IOMMUは、CXL.cacheトランザクションのアドレス変換のために呼び出されることはない。

2.7. PCIe ATSページ要求処理

ページ・リクエスト」または「ストップ・マーカー」メッセージを処理するために [1]、IOMMUは最初にデバイス・コンテキストを検索し、リクエスタに対して ATSとPRIが有効になっているかどうかを判断する。ATSとPRIが有効、すなわちEN_ATSとEN_PRIの両方が1に設定されている場合、 IOMMUはメッセージをページリクエストキュー(PQ)と呼ばれるメモリ内キューに キューイングする(セクション3.3参照)。ページ要求」の適切な処理の後、ソフトウェア・ハンドラは、デバイスに対して「ペ ージ要求グループ・レスポンス」メッセージを生成することができる。

デバイスに対して PRI が有効になっている場合でも、IOMMU は、キューが無効になっている、 キューが一杯になっている、または IOMMU がキュー・メモリにアクセスし ようとしたときにアクセス・フォールトに遭遇した、などのエラー状 態のために、PQ を通して「ページ要求」または「停止マーカ」メッセージを報告できないことがある。これらのエラー条件はセクション3.3で規定されている。

ddtp.iommu_modeがBareまたはOffの場合、IOMMUは要求元のデバイスコンテキストを見つけることができない。

EN_PRI が 0 に設定されているか、EN_ATS が 0 に設定されているか、IOMMU が EN_PRI 設定を決定するために DC を見つけることができないか、要求が PQ にキューイングできなかった場合、IOMMU の動作は「ページ要求」のタイプに依存する。

- Page Request "が応答を必要としない場合、すなわち、メッセージの "Last Request in PRG "フィールドが0に設定されている場合、そのようなメッセージは静かに破棄される。「Stop Marker "メッセージは応答を必要としないので、そのようなエラーの場合は 常に静かに破棄される。
- ページ要求」に応答が必要な場合は、IOMMU 自身がデバイスに対して「ページ要求グループ応答」メッセージを生成することができる。

IOMMU が応答を生成するとき、応答のステータスフィールドはエラーの原因に依存する。

以下の障害が発生した場合、ステータスは「応答失敗」に設定される：

- ddtp.iommu_modeがオフ
- DDT エントリロードアクセスフォルト（原因 = 257）
- DDTエントリの設定ミス（原因=259）
- DDT エントリが無効（原因 = 258）
- ページ要求キューが有効になっていない (pqcsr.pqen == 0 または pqcsr.pqon == 0)
- ページ要求キューでメモリアクセスフォールトが発生した (pqcsr.pqmf == 1) 以

下のフォールトが発生した場合、ステータスはInvalid Requestに設定される：

- ddtp.iommu_modeはBareです。
- EN_PRIが0に設定される

他のエラーは発生しなかったが、ページ要求キューが満杯(pqt == pqh - 1)またはオーバーフロー(pqcsr.pqof == 1)のため、「ページ要求」をキューに入れることができなかった場合、ステータスは「成功」に設定される。

SR-IOV VF が割り当て単位として使用されている場合、ハイパーバイザーは EN_PRI を 0 に設定することで、仮想ファンクションの 1 つからのページ要求を無効にすることができる。IOMMU プロトコル固有のロジックは、デバイス内の共有 PRI がすべての PF/VF に対して無効化されるのを回避するために、この状態（原因 = 260）を非災害的障害である無効なリクエストとしてその応答で分類する。

ストップマーカー」は、PASIDを持つ「ページリクエスト」としてエンコードされるが、 L、W、Rフィールドはそれぞれ1、0、0に設定される。

ステータスがInvalid RequestまたはSuccessであるIOMMU生成の「ページリクエストグループ応答」メッセージの場合、PRG-response-PASID-required(PRPR)ビットが1に設定されると、関連する「ページリクエスト」にPASIDがあった場合、IOMMU応答メッセージはPASIDを含むべきであることを示す。

IOMMUが生成した「ページリクエストグループレスポンス」のレスポンスコードが「レスポンス失敗」に設定されている場合、「ページリクエスト」にPASIDがあれば、レスポンスはPASID付きで生成される。

以下の条件では、PCIe ATS "Page Request "メッセージのフォルトキューにフォルトは記録されません:

- ページ要求キューが有効になっていない (pqcsr.pqen == 0 または pqcsr.pqon == 0)
- ページ要求キューでメモリアクセスフォールトが発生した (pqcsr.pqmf == 1)
- ページ・リクエスト・キューが満杯（pqt == pqh - 1）、またはオーバーフロー（pqcsr.pqof == 1）のため、"ページ・リクエスト "をキューに入れることができませんでした。

2.8. メモリ内データ構造のキャッシュ

ダイレクトメモリアクセス（DMA）変換を高速化するために、IOMMUは、デバイス・ディレクトリ・テーブル、プロセス・ディレクトリ・テーブル、ファーストステージおよびセカンドステージ変換テーブル、およびMSIページ・テーブルからのエントリを保持する変換キャッシュを使用することができる。これらのキャッシュはIOMMUアドレス変換キャッシュ（IOATC）と総称される。

本仕様では、V（有効）ビットがクリアされたファーストステージ/セカンドステージ PTE、V（有効）ビットがクリアされたノンリーフ DDT エントリ、V（有効）ビットがクリアされたデバイスコンテキスト、V（有効）ビットがクリアされたノンリーフ PDT エントリ、V（有効）ビットがクリアされたプロセスコンテキスト、または V ビットがクリアされた MSI PTE のキャッシュを許可しない。

これらの IOATC は、RISC-V ハートやデバイス DMA による明示的なロードとストアを使用したメモリ内

データ構造の変更を監視しない。ソフトウェアは、メモリ内データ構造の更新を監視するために、IOMMU コマンドを使用してキャッシュされたデータ構造エントリを無効にし、IOMMU オペレーションを同期させなければならない。より単純な実装では、メモリ内データ構造の一部または全部に IOATC を実装しないかもしれない。IOMMU コマンドは、特定のエントリまたはエントリのグループを識別するために、1 つ以上の ID を使用してキャッシュされたエントリにタグを付けることができる。

表 6.IOATC エントリーのタグ付けに使われる識別子

キャッシュされたデータ構造	エントリーのタグ付けに使用されるID	無効化コマンド
デバイス・ディレクトリ・テーブル	デバイスID	iodir.inval_ddt
プロセス・ディレクトリ・テーブル	device_id, process_id	iodir.inval_pdt
第 1 段ページ表（第 2 段がベアでない場合）	GSCID、PSCID、IOVA	IOTINVAL.VMA
ファーストステージのページ表（セカンドステージがベアである場合）	PSCID、IOVA	IOTINVAL.VMA
セカンドステージのページ表	GSCID、GPA	IOTINVAL.GVMA
MSIページテーブル	GSCID、GPA	IOTINVAL.GVMA

2.9. メモリ内データ構造エントリの更新

RISC-Vのメモリ・モデルでは、ハートからのメモリ・アクセスはシングル・コピー・アトミックでなければならない。RV32が実装されている場合、シングル・コピー・アトミック・メモリ・アクセスのサイズは最大32ビットです。RV64が実装されている場合、シングル・コピー・アトミック・メモリ・アクセスのサイズは最大64ビットです。IOMMUによって実装されるシングル・コピー・アトミック・メモリ・アクセスのサイズはUNSPECIFIEDですが、システム内の全てのハートがRV32を実装している場合は少なくとも32ビットであることが要求され、システム内のいずれかのハートがRV64を実装している場合は少なくとも64ビットであることが要求されます。

IOMMUデータ構造エントリにはVビットがあり、このビットが1にセットされるとエントリが有効であることを示す。

ソフトウェアは、Vビットが1に設定されているデータ構造エントリーを更新することができる。

- ソフトウェアが、IOMMUがサポートする最小シングルコピー・アトミックメモリアクセス未満の幅のストアセットを使用して、有効なデータ構造エントリーのフィールドを更新することは、一般的に安全ではない。
- IOMMU データ構造エントリーの更新がアトミックであるためには、ソフトウェアは、IOMMU がサポートする最小のシングルコピーのアトミックメモリアクセスに等しい幅のシングルストアを使用し

なければならない。

- あるフィールドの更新によって、そのフィールドがエントリーの他のフィールドと矛盾する場合、ソフトウエアは、エントリーの他のフィールドを更新する前に、まずVフィールドを0に設定し、セクション2.8で概説されているコマンドを使用して、IOMMUキャッシュにある可能性のあるそのエントリーの以前のコピーを無効にしなければならない。
- IOMMUは、ソフトウェアによるエントリーの更新を直ちに観測する必要はない。ソフトウェアはセクション2.8で説明されているコマンドを使用して、IOMMUのキャッシュにあるそのエントリーの以前のコピーを無効にし、エントリーの更新とIOMMUの動作を同期させなければならない。

データ構造エントリーが変更された場合、IOMMUはエントリーの古い値を使用することも、新しい値を使用することもでき、ソフトウェアがセクション2.8で概説されているコマンドを使用してそのエントリーの以前のコピーを無効にするまで、その選択は予測できない。

エントリの更新を IOMMU の動作と同期させるために、IOMMU のキャッシュにあるかもしれない。これらは期待される唯一の動作である。

2.10. メモリ内データ構造のエンディアン性

RISC-V メモリモデルでは、アドレス空間全体のバイト不変性が規定されている。ミックスエンディアン動作モードがサポートされている場合、IOブリッジとIOMMUは、所定のアドレスへのバイトアクセスがリトルエンディアンとビッグエンディアンの両方の動作モードで同じメモリ位置にアクセスするように、バイト不変アドレッシングを実装する必要があります。

メモリ内データ構造への暗黙のメモリアクセスのエンディアンは、fctl.BE
または以下のようにDC.tc.SBEによって行われる：

表7. データ構造へのメモリアクセスのエンディアン

データ構造	管理者
デバイス・ディレクトリ・テーブル	fctl.BE
セカンドステージのページ表	fctl.BE
MSIページテーブル	fctl.BE
プロセス・ディレクトリ表	DC.tc.SBE
第1ステージのページ表	DC.tc.SBE

第 1 段コンテキストの PSCID フィールドは、GSCID（2 段アドレス変換が有効な場合）と共にアドレス空間を識別する。異なるSBEを持つ2つのDCで同一のGSCIDとPSCIDを構成することは予期されず、もし行われた場合、IOMMUが第1段のPTEをビッグエンディアンまたはリトルエンディアンと解釈する可能性がある。これらは期待される唯一の動作である。

ソフトウェアは、エンディアンを共有しないIOエージェントとデータを共有するとき、必要に応じて適切なソフトウェアシーケンスを使用してバイトをスワップし、相互に合意されたデータ表現を作成しなければならない。ソフトウェアは、そのようなIOエージェントと共有されるデータがアトミックにアクセスされなければならないとき、非ネイティブエンディアン形式でアトミック操作を実行するために、LR/SCシーケンスを使用しなければならない。

第3章.インメモリ・キュー・インターフェイス

ソフトウェアとIOMMUは、3つのインメモリキューデータ構造を使って相互作用する。

- ソフトウェアがIOMMUへのコマンドをキューに入れるために使用するコマンドキュー（CQ）。
- IOMMUがフォールトやイベントをソフトウェアに知らせるために使用するフォールト／イベントキュー（FQ）。
- IOMMU が、PCIe デバイスから受け取った "Page Request" メッセージを報告するために使用するページリクエストキュー (PQ) 。このキューは、IOMMU が PCIe [1] で定義されたページリクエストインタフェースをサポートしている場合にサポートされる。

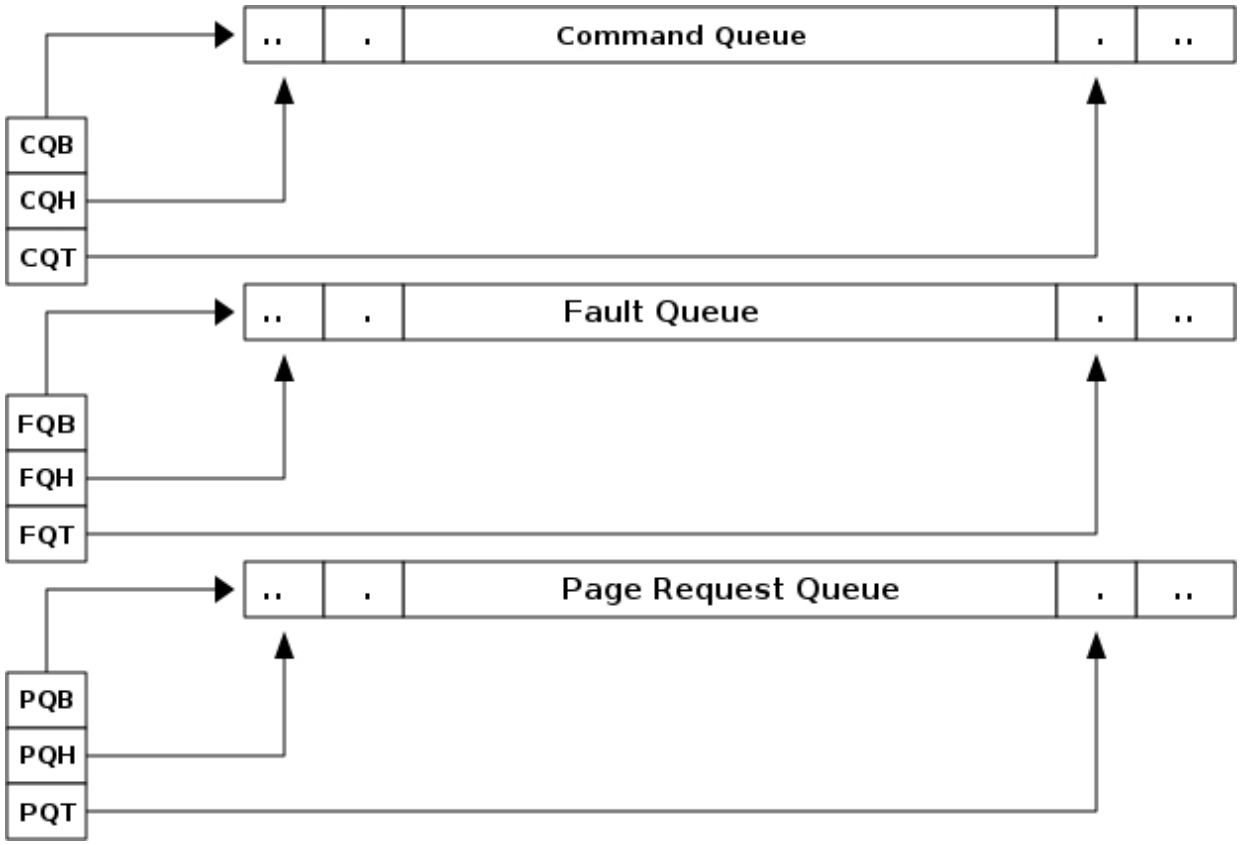


図28.IOMMU インメモリーキュー

各キューは、キューからのデータの消費者によって制御されるヘッドと、キューへのデータの生産者によって制御されるテールを持つ円形バッファである。IOMMUは、PQとFQへのレコードのプロデューサであり、テールレジスタを制御する。IOMMUは、ソフトウェアによって生成されたコマンドのCQへの消費者であり、ヘッド・レジスタを制御する。テール・レジスタは、プロデューサーによって次のエントリが書き込まれるキューへのインデックスを保持する。ヘッド・レジスタは、コンシューマが次のエントリを読み込んで処理するキューへのインデックスを保持する。

先頭と末尾が等しい場合、キューは空である。テールがヘッドから1を引いた値である場合、キューは満杯である。headとtailは、循環バッファの終端に達した時点で折り返される。

データのプロデューサーは、キューに書き込まれたデータとテール・アップデートが、テール・レジスタへのアップデートを観測するコンシューマーがすべてのデータも観測しなければならないような順序であることを保証しなければならない。

headとtailで決定されたオフセット間のキューに生成される。

すべてのRISC-V IOMMU実装は、メインメモリに配置されたインメモリキューをサポートする必要がある。I/Oメモリ内のインメモリキューのサポートは必須ではないが、本仕様では禁止されていない。

3.1. コマンドキュー（CQ）

コマンド・キューは、ソフトウェアがIOMMUで処理されるコマンドをキューに入れるために使用される。各コマンドは16バイトである。

このメモリ内キューのベースのPPNとキューのサイズは、コマンド・キュー・ベース（cqb）と呼ばれるメモリ・マップド・レジスタに設定される。

コマンド・キューの末尾は、コマンド・キュー・テール（cqt）と呼ばれるソフトウェア制御の読み書き可能なメモリ・マップド・レジスタに存在する。cqtは、ソフトウェアが次に書き込むコマンド・キュー・エントリーのインデックスである。コマンドを書き込んだ後、ソフトウェアは書き込んだコマンドの数だけcqtを進める。

コマンド・キューの先頭は、コマンド・キュー・ヘッド（cqh）と呼ばれる、読み取り専用でメモリ・マップされたIOMMU制御レジスタに存在する。cqhは、IOMMUが次に処理すべきコマンド・キューへのインデックスである。cqh==cqtの場合、コマンド・キューは空である。cqt == (cqh - 1)の場合、コマンド・キューは満杯である。

cqcsrのエラー・ビットまたはfence_w_ipビットが1 の とき、コマンド・キューからの割り込みが有効（すなわちcqcsr.cieが1）であれば、ipsrにコマンド・キュー割り込み保留（cip）ビットが設定される。

IOMMU コマンドは、オペコードによって決定される主要なコマンドグループに分類され、各グループの func3 フィールドは、そのコマンドによって呼び出される関数を指定する。オペコードは、オペランド・フィールドのフォーマットを定義する。これらのフィールドの1つ以上が、呼び出される特定の関数によって使用される可能性がある。オペコードのエンコーディング64から127は、カスタム用に指定されている。

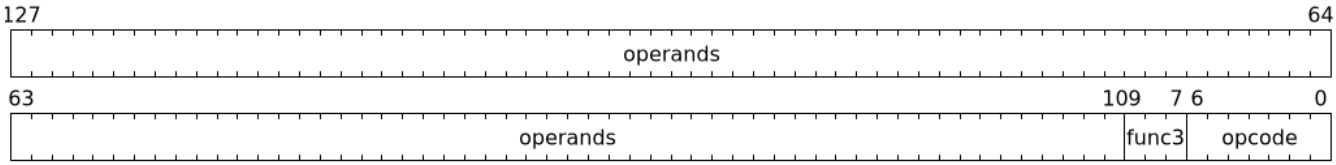


図29.IOMMU コマンドのフォーマット

コマンドは2つの64ビット・ダブルワードとして解釈される。メモリ上の各ダブルワードのバイトオーダー、リトルエンディアンまたはビッグエンディアンは、fctl.BE（セクション5.4）で決定されたエンディアンになる。

以下のコマンド・オプコードが定義されている：

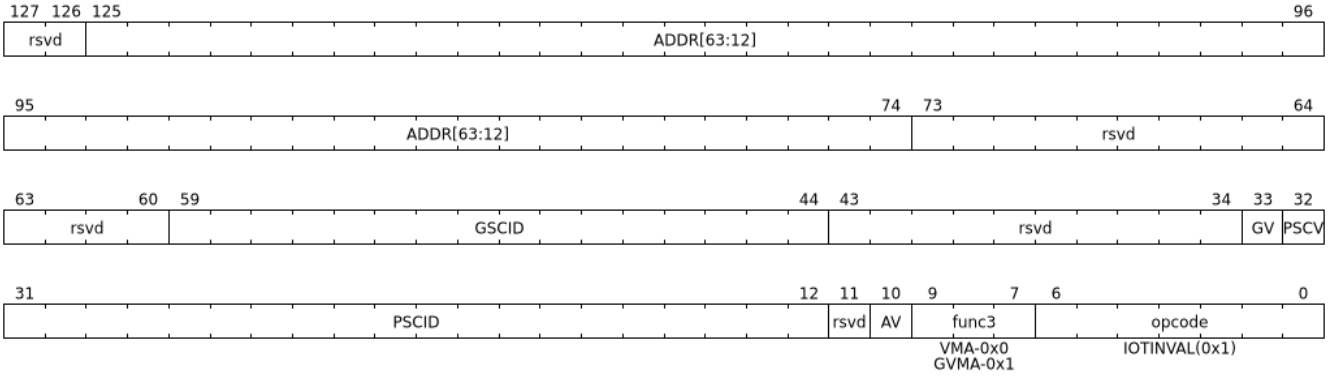
表8.IOMMUコマンドのオペコード

命令コード	エンコーディング	説明
IOTINVAL	1	IOMMU ページテーブルキャッシュ無効化コマンド。
IOFENCE	2	IOMMUコマンドキューフェンスコマンド。
命令コード	**エンコーディング**	**説明**
IOTDIR	3	IOMMU ディレクトリキャッシュ無効化コマンド。
ATS	4	IOMMU PCIe [1] ATS コマンド。
予約	5-63	将来の標準的な使用のために予約されている。
カスタム	64-127	カスタム仕様。

コマンド・オプコード0から63までの未定義の機能はすべて、将来の標準使用のために予約されている。

コマンドが予約されたエンコーディングを使用しているか、予約ビットが 1 に設定されている場合、不正なコマンドと判断される。コマンドは、IOMMU capability レジスタによって定義されているが実装されていない場合、サポートされていないと判断される。不正なコマンドまたはサポートされていないコマンドがコマンド・キューによってフェッチされ、デコードされた場合、コマンド・キューは cqcsr.cmd_ill ビットをセットし、コマンド・キューからのコマンド処理を停止する。コマンド処理を再び有効にするには、ソフトウェアがcmd_illビットに1を書き込んでクリアする必要がある。

3.1.1. IOMMUページテーブルキャッシュ無効化コマンド



IOMMU 操作は、PDT、ファーストステージおよびセカンドステージページテーブルへの暗黙の読み出しを引き起こす。このような読み出しの待ち時間を短縮するために、IOMMUは、IOMMU-address-translation-cache（IOATC）内のファーストステージおよび/またはセカンドステージページテーブルからのエントリをキャッシュすることができる。これらのキャッシュは、ソフトウェアがメモリ上のこれらのデータ構造に対して実行した変更を観察しないかもしれない。

IOMMUの翻訳テーブルキャッシュ無効化コマンドであるIOTINVAL.VMAおよびIOTINVAL.GVMAは、それぞれメモリ内の第1ステージおよび第2ステージのページテーブルデータ構造の更新をIOMMUの動作と同期させ、一致するIOATCエントリを無効にする。

GV オペランドは、ゲストソフトコンテキスト ID (GSCID) オペランドが有効かどうかを示します。PSCV オペランドは、プロセスソフトコンテキスト ID (PSCID) オペランドが有効かどうかを示します。PSCV を 1 に設定することは、IOTINVAL.VMA に対してのみ許可されます。AVオペランドは、アドレス（ADDR）オペランドが有効かどうかを示す。GVが0であるとき、ホストに関連する翻訳（すなわち、第2段がBareである翻訳）が操作される。GV が 0 のとき、GSCID オペランドは無視される。AV が 0 のとき、ADDR オペランドは無視される。PSCV オペランドが 0 のとき、PSCID オペランドは無視される。

IOTINVAL.VMAは、IOMMUから対応する第1ステージのページテーブルへの後続のすべての暗黙の読み出しの前に、ハーツが第1ステージのページテーブルに行った以前のストアがIOMMUによって観察されることを保証する。

表9.IOTINVAL.VMAの オペランドとオペレーション

GV	AV	セル速度	オペレーション
0	0	0	すべてのホストアドレス空間に対して、グローバルマッピングを含むすべてのアドレス変換キャッシュエントリを無効にします。
0	0	1	PSCIDオペランドで識別されるホストアドレス空間のすべてのアドレス変換キャッシュエントリを、グローバルマッピングを含むエントリを除いて無効にする。
0	1	0	ADDRオペランドのIOVAに対応するグローバルマッピングを含む、第1段階のリーフページテーブルエントリを含むすべてのアドレス変換キャッシュエントリを、すべてのホストアドレス空間に対して無効にする。
0	1	1	ADDRオペランドのIOVAに対応する第1段リーフ・ページ・テーブル・エントリを含み、PSCIDオペランドで識別されるホスト・アドレス空間に一致するアドレス変換キャッシュ・エントリを、グローバル・マッピングを含むエントリを除いてすべて無効にする。
1	0	0	GSCIDオペランドに関連付けられたすべてのVMアドレス空間の、グローバルマッピングを含むすべてのアドレス変換キャッシュエントリを無効にします。
1	0	1	グローバルマッピングを含むエントリを除き、PSCID および GSCID オペランドで識別される VM アドレス空間のすべてのアドレス変換キャッシュエントリを無効にします。
1	1	0	ADDRオペランドのIOVAに対応するグローバルマッピングを含む、第1段階のリーフページテーブルエントリーを含むすべてのアドレス変換キャッシュエントリーを、GSCIDオペランドに関連付けられたすべてのVMアドレス空間に対して無効にします。
1	1	1	PSCID オペランドと GSCID オペランドで識別される VM アドレス空間の、ADDR オペランドの IOVA に対応する第 1 段リーフ・ページ・テーブル・エントリを含むすべてのア

			ドレス変換キャッシュ・エントリを無効にします（グローバル・マッピングを含むエントリを除く）。

IOTINVAL.GVMAは、IOMMUから対応する第2段ページテーブルへの後続のすべての暗黙の読み出しの前に、第2段ページテーブルに行われた以前のストアが観察されることを保証する。IOTINVAL.GVMAでPSCVを1に設定することは違法である。

表10.IOTINVAL.GVMAオペランドとオペレーション

GV	AV	**オペレーション**
0	イグノレッド	すべてのVMアドレス空間について、セカンドステージページテーブルのどのレベルからもキャッシュされた情報を無効にする。
GV	AV	**オペレーション**
1	0	セカンドステージページテーブルのどのレベルからもキャッシュされた情報を無効にするが、GSCIDオペランドで識別されるVMアドレス空間に対してのみ無効にする。
1	1	ADDRオペランドのゲスト物理アドレスに対応するリーフ第2ステージページテーブルエントリからキャッシュされた情報を、GSCIDオペランドで識別されたVMアドレス空間に対してのみ無効にします。

1つはゲストの仮想アドレスをゲストの物理アドレスにマップし、もう1つはゲストの物理 アドレスをスーパバイザ物理アドレスにマップします。IOTINVAL.GVMAは前者のキャッシュを無効にする必要はありませんが、後者のキャッシュからIOTINVAL.GVMAアドレスおよびGSCIDオペランドに一致するエントリを無効にする必要があります。

より一般的には、実装にアドレス変換キャッシュが含まれており、ゲス ト仮想アドレスが直接スーパーバイザ物理アドレスにマップされ、インダイ レクションレベルが削除されます。このような実装では、IOTINVAL.GVMAアドレスとGSCID引数に一致するゲスト物理アドレスにゲスト仮想アドレスがマップされるすべてのエントリを無効にする必要があります。この方法でエントリを選択的に無効にするには、ゲスト物理アドレスでタグ付けする必要があり、コストがかかるため、一般的な手法は、address引数に関係なく、GSCID引数に一致するすべてのエントリを無効にすることである。

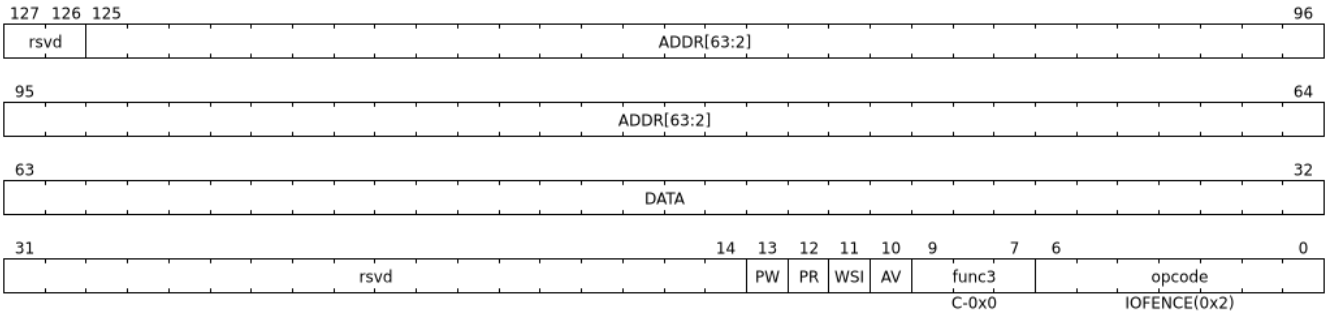
より単純な実装では、IOTINVAL.VMAおよび/またはIOTINVAL.GVMAのオペランドを無視し、常にすべてのアドレス変換エントリーのグローバル無効化を実行することができる。

この仕様の結果として、実装は、そのアドレスを包含する最新の IOTINVAL 以降の任意の時点で有効であったアドレスに対する任意の翻訳を使用することができる。特に、リーフPTEが変更されたが、それを包含するIOTINVALが実行されな かった場合、古い訳語または新しい訳語のいずれかが使用されるが、その選択は 予測不可能である。それ以外の動作は明確に定義されている。

従来のTLB設計では、例えば、最初に元のノンリーフPTEの有効ビットをクリアし、AV=0でIOTINVAL.VMAまたはIOTINVAL.GVMAを実行することなく、ページがより大きなページにアップグレードされた場合、複数のエントリが1つのアドレスに一致する可能性があります。この場合、同様の注意が適用されます：古いノンリーフPTEと新しいリーフPTEのどちらが使用されるかは予測できませんが、それ以外の動作は明確に定義されています。

この仕様のもう1つの帰結は、PTEの幅より小さい幅のストアのセットを使用してPTEを更新することは一般的に安全ではないということである。

3.1.2. IOMMUコマンドキューフェンスコマンド



IOMMUはCQから順番にコマンドをフェッチするが、IOMMUはフェッチしたコマンドを順番通りに実行しないことがある。IOMMU advancing cqhは、IOMMUがフェッチしたコマンドが実行またはコミットされたことを保証するものではない。

IOFENCE.Cコマンドの完了は、cqhがCQ内のIOFENCE.Cコマンドのインデックスを越えて進むことによって決定され、CQからフェッチされた以前のすべてのコマンドが完了し、コミットされたことを保証する。

IOFENCE.C が、タイムアウトが指定された前のコマンドの完了待ちでタイムアウトした場合、 cqcsr セクション 5.15 の cmd_to ビットがこの状態を知らせるためにセットされる。cqh は、タイムアウトした IOFENCE.C のインデックスを保持し、タイムアウトが指定され ていない前のコマンドはすべて完了し、コミットされている。

このバージョンの仕様では、ATS.INVALコマンドだけがタイムアウトを持つよ う に 指定されている。

コマンドは、IOMMU、他のRISC-Vハート、外部デバイスやコプロセッサから見て、IOMMUに接続されたI/Oデバイスからのメモリアクセスを命令するために使用することができる。

PR ビットを 1 に設定すると、IOMMU が既に処理したデバイスからの以前のすべての読み出し要求を、システム内のすべての RISC-V ハー トおよび IOMMU が観察できるように、グローバルな順序付けポイントにコミットするように IOMMU に要求できます。

PW ビットを 1 に設定すると、IOMMU が既に処理したデバイスからの以前のすべての書き込み要求が、システム内のすべての RISC-V ハー トおよび IOMMU が観察できるように、グローバルな順序付けポイントにコミットされるように IOMMU に要求できます。

WSI（wire-signaled-interrupts） ビットを 1 に設定すると、IOFENCE.C の完了時に（cqcsr.fence_w_ip - セクション 5.15 の設定により）コマン ド・キューからのワイヤード割り込みが発生する。このビットは、IOMMU がワイヤード割り込みをサポートしていないか、 ワイヤード割り込みが有効になっていない（すなわち、fctl.WSI == 0）場合に予約される。

ソフトウェアは、以前にデバイスからアクセスできるようにされたメモリを再生する前に、IOMMU によって処理された以前のすべての読み取りと書き込みがグローバル順序ポイントにコミットされていることを確認する必要があります。このようなメモリ再利用の安全な順序は、まずページ・テーブルを更新してデバイスからメモリへのアクセスを禁止し、次に IOTINVAL.VMA または IOTINVAL.GVMA を適切に使用して IOMMU をページ・テーブルの更新と同期させることである。同期化の一環として、再生されたメモリが以前に

そうでなければ、再生されたメモリが以前にデバイスに書き込みアクセス可能であったなら、以前のすべての読み出しと書き込みの順番を要求する。再生されたメモリが、デバイスから見えてはならないデータを保持するために使用される場合、以前の読み出しの順序付けが要求されるかもしれない。

PRおよび/またはPWが1に設定されたIOFENCE.Cは、IOMMUによって既に処理されたリクエストがグローバル・オーダリング・ポイントにコミットされることを保証するだけである。IOMMU によってまだ処理されていないデバイスからのすべてのインフライト・リクエストを確実に観測する必要がある場合、ソフトウェアはインターコネクト固有のフェンス動作を実行しなければならない。例えば、PCIe の場合、デバイス・メモリからの読み出しに応答するデバイスからの完了は、完了が以前のポストされた書き込みを通過しない可能性があるため、以前のポストされた書き込みが IOMMU によって観測されることを保証するという特性を持つ。

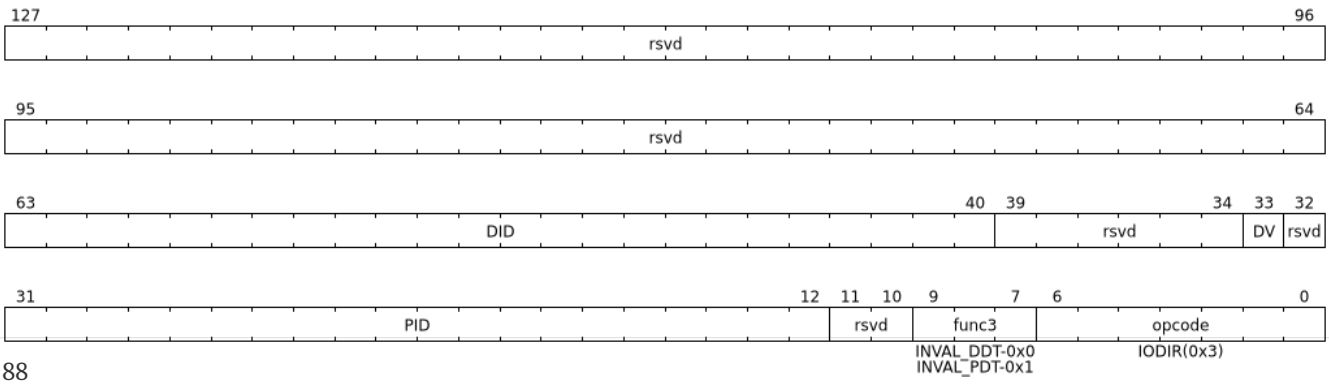
順序保証はメインメモリへのアクセスに対して行われる。I/Oメモリへのアクセスについては、順序保証は実装とI/Oプロトコルで定義される。

より単純な実装では、以前のすべてのメモリアクセスを無条件にグローバルに順序付けることができる。

AV コマンドオペランドは、ADDR[63:2]オペランドと DATA オペランドが有効かどうかを示す。AV=1 の場合、IOMMU はコマンド完了時に 4 バイトストアとして 4 バイトアラインドアドレス ADDR[63:2] * 4 のメモリに DATA を書き込む。AV=0の場合、ADDR[63:2]およびDATAオペランドは無視される。

ソフトウェアは、ADDR[63:2] コマンド・オペランドを構成して、IMSIC 内の seteipnum_le/seteipnum_be レジスタのアドレスを指定し、IOFENCE.C 完了時に外部割り込み通知を発生させることができる。あるいは、ソフトウェアが ADDR[63:2] をメモリ位置にプログラムし、IOFENCE.C を使用してコマンドの完了を示すフラグをメモリに設定することもできる。

3.1.3. IOMMUディレクトリキャッシュ無効化コマンド



IOMMU の操作は、DDT および/または PDT への暗黙の読み込みを引き起こす。このような読み出しの待ち時間を短縮するために、IOMMU は DDT および/または PDT のエントリを IOMMU ディレクトリキャッシュにキャッシュすることがある。これらのキャッシュは、ソフトウェアがメモリ上のこれらのデータ構造に対して実行した変更を観察しないかもしれない。

IOMMUのDDTキャッシュ無効化コマンドIODIR.INVAL_DDTは、DDTの更新をIOMMUの動作と同期させ、一致するキャッシュエントリーをフラッシュする。

IOMMUのPDTキャッシュ無効化コマンドIODIR.INVAL_PDTは、PDTの更新をIOMMUの動作と同期させ、一致するキャッシュエントリーをフラッシュする。

DV オペランドは、デバイス ID(DID)オペランドが有効かどうかを示す。IODIR.INVAL_PDT の場合、DV オペランドは 1 でなければならない。DVオペランドが1の場合、DIDオペランドの値はddtp.iommu_modeでサポートされる値より広くてはならない。

IODIR.INVAL_DDTは、IOMMUからDDTへの後続のすべての暗黙の読み出しの前に、RISC-Vハートによって DDTに行われた以前のストアが観察されることを保証する。DV が 0 の場合、コマンドはすべてのデバイスに対してキャッシュされたすべての DDT および PDT エントリを無効にします。DV が 1 の場合、コマンドは DID オペランドで識別されるデバイスのキャッシュされたリーフレベルの DDT エントリと、関連するすべての PDT エントリを無効にする。PIDオペランドはIODIR.INVAL_DDTコマンドのために予約されている。

IODIR.INVAL_PDTは、IOMMUからPDTへの後続のすべての暗黙的な読み取りの前に、RISC-VハートによってPDTに行われた以前の保存が観察されることを保証する。このコマンドは、指定されたPIDとDIDに対してキャッシュされたPDTリーフ・エントリーを無効にする。IODIR.INVAL_PDTのPIDオペランドの幅は、IOMMUがサポートする幅を超えてはならない（セクション5.3参照）。

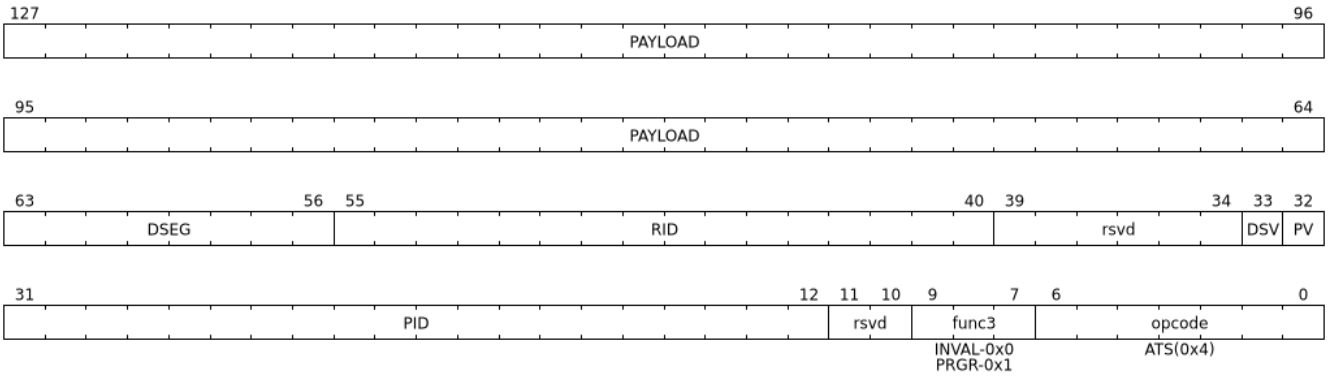
Device-context または Process-context の一部のフィールドは、ゲスト物理アド レスである可能性がある。デバイスコンテキストまたはプロセスコンテキストをキャッシュする実装は、これらのフィールドをスーパバイザ物理アドレスに変換した後にキャッシュしてもよい。他の実装では、これらのフィールドをゲスト物理アドレスとしてキャッシュし、これらのアドレスによって参照されるメモリにアクセスする直前に、セカンドステージページテーブルを使用してスーパーバイザ物理アドレスに変換することができる。

これらのトランスレーションに使用されるセカンドステージページテーブルが変更された場合、実装によってはトランスレーションされたスーパバイザ物理アドレスポインタをIOMMUディレクトリキャッシュにキャッシュするため、ソフトウェアは適切なIODIRコマンドを発行する必要があります。

IOTINVAL コマンドは、IOMMU ディレクトリキャッシュには影響しない。

3.1.4. IOMMU PCIe ATSコマンド

このコマンドは、capabilities.ATSが1に設定されている場合にサポートされる。



ATS.INVAL コマンドは、RID で識別される PCIe デバイス・ファンクションに「無効化要求」メッ セージを送信するよう IOMMU に指示します。無効化要求」メッセージは、デバイス機能のアドレス変換キャッシュからアドレス範囲の特定のサブセットをクリアするために使用されます。ATS.INVAL

コマンドは、デバイスから "無効化完了 "応答メッセージを受信するか、応答を待っている間にプロトコルで定義されたタイムアウトが発生すると完了する。IOMMU は応答を待っている間、cqh を進め、CQ からさらにコマンドをフェッチすることができる。タイムアウトが発生した場合は、後続の IOFENCE.C コマンドが実行されたときに報告される。

デバイス上で無効化操作が完了したかどうかを知る必要があるソフトウェアは、IOMMUコマンド・キュー・フェンス・コマンド（IOFENCE.C）を使用して、以前のすべての「無効化要求」メッセージに対する応答を待つことができる。IOFENCE.Cは、以前にフェッチされたすべてのコマンドが実行され完了する前に完了しないことが保証されている。デバイスのATCを無効にするために以前にフェッチされたATSコマンドは、要求がタイムアウトするか、デバイスから有効な応答を受信するまで完了しません。

IOFENCE.Cの前にある1つ以上のATS無効化コマンドがタイムアウトした場合、ソフトウェアはCQを再び動作可能にし、タイムアウトした可能性のある無効化コマンドを再送信することができる。IOFENCE.Cの前にキューに入れられたATS.INVALコマンドが複数のデバイスに向けられていた場合、ソフトウェアは、タイムアウトの原因となったデバイスを識別するために、これらのコマンドをATS.INVALとIOFENCE.Cのペアとして再送信することができる。

ATS.PRGR コマンドは、IOMMU に対して、RID で識別される PCIe デバイス・ファンクションに 「ページ要求グループ応答」メッセージを送信するように指示します。ページ・リクエスト・グループ・レスポンス "メッセージは、"ページ・リクエスト "の完了、またはインターフェイスの致命的な故障を通知するために、デバイス機能のページ・リクエスト・インターフェイスと通信するために、システム・ハードウェアおよび/またはソフトウェアによって使用される。

PV オペランドが 1 に設定されている場合、メッセージは PASID フィールドが PID オペランドに設定された PASID とともに生成される。PV オペランドが 0 に設定されている場合、PID オペランドは無視され、メッセージは PASID なしで生成される。

コマンドのPAYLOADオペランドはメッセージ・ボディを形成するために使用され、そのフィールドはPCIe仕様[1]で規定されている通りである。PAYLOAD フィールドは以下のようにフォーマットされる：

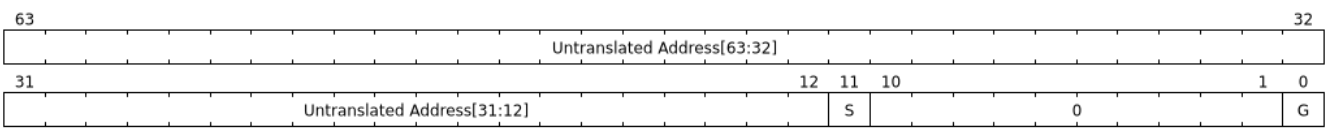


図30.ATS.INVALコマンドのPAYLOAD

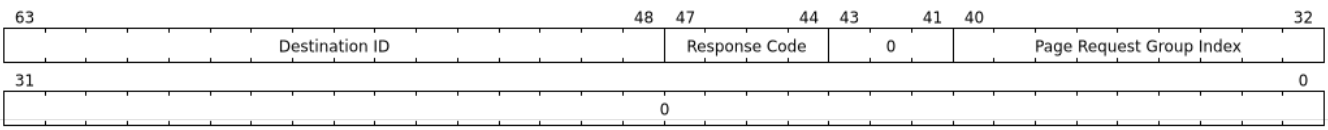


図31 ATS.PRGRコマンドのPAYLOADATS.PRGRコマンドのPAYLOAD

DSVオペランドが1の場合、有効な宛先セグメント番号がDSEGオペランドで指定される。DSVオペランドが0の場合、DSEGオペランドは無視される。

階層はPCI ExpressのI/O相互接続トポロジで、バス/デバイス/ファンクション番号のタプルと呼ばれる構成空間アドレスが一意である。一部のコンテキストでは、階層はセグメントとも呼ばれます、

セグメント番号がファンクションのIDに含まれることがある。

3.2. 障害/イベントキュー（FQ）

フォールト/イベント・キューは、トランザクション処理時に発生するイベントやフォールトを報告するために使用されるメモリ内キューのデータ構造である。各フォールトレコードは32バイトである。

このメモリ内キューのベースのPPNとキューのサイズは、フォールトキューベース（fqb）と呼ばれるメモリマップレジスタに設定される。

フォルトキューの末尾は、fqt と呼ばれる IOMMU 制御の読み出し専用メモリマップレジス タに存在する。 fqt は、IOMMU がフォルトキューに書き込む次のフォルトレコードのイン デックスである。フォルトキューの先頭は、fqh と呼ばれる読み取り/書き込みメモリマップド・ソフトウェア制御レジスタにあります。fqhは、SWが次に処理すべきフォルトレコードのインデックスである。フォルトレコードを処理した後、ソフトウェアは処理したフォルトレコードの数だけ fqh を進める。fqh == fqt の場合、フォルトキューは空である。fqt == (fqh - 1) の場合、フォルトキューは満杯である。

フォルトレコードは4つの64ビットダブルワードとして解釈される。メモリ上の各ダブルワードのバイトオーダー、リトルエンディアンまたはビッグエンディアンは、fctl.BE（セクション5.4）で決定されるエンディアンとなる。

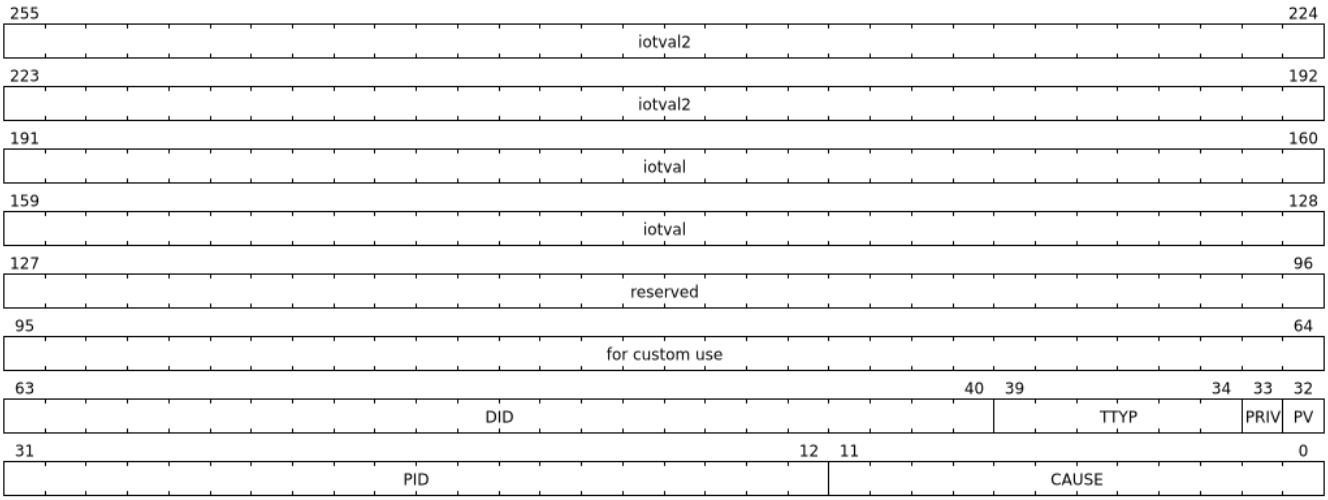


図32.障害キューの記録

CAUSE は、フォルト/イベントの原因を示すコードである。

表11.障害レコードの CAUSE フィールドのエンコーディング

原因	説明	以下の場合に報告 DTFが1?

1	命令アクセス障害	いいえ
4	読み出しアドレスがずれている	いいえ
5	読み取りアクセス障害	いいえ
6	書き込み/AMOアドレスがずれている	いいえ
7	書き込み/AMOアクセス障害	いいえ
原因	**説明**	**以下の場合に報告 DTFが1?**
12	インストラクションページの不具合	いいえ
13	ページフォルトを読む	いいえ
15	書き込み/AMOページフォルト	いいえ
20	インストラクションゲストページ障害	いいえ
21	ゲストページの障害を読む	いいえ
23	書き込み/AMOゲスト・ページ・フォールト	いいえ
256	すべてのインバウンド取引を拒否	はい
257	DDT エントリ・ロード・アクセス・フォールト	はい
258	DDTエントリーが無効	はい
259	DDTエントリーの設定ミス	はい
260	許可されないトランザクション・タイプ	いいえ
261	MSI PTEロードアクセスフォールト	いいえ
262	MSI PTEが有効でない	いいえ
263	MSI PTEの設定ミス	いいえ
264	MRIFアクセス障害	いいえ
265	PDTエントリ・ロード・アクセス・フォールト	いいえ
266	PDTエントリーが無効	いいえ
267	PDTエントリーの設定ミス	いいえ
268	DDTデータ破損	はい
269	PDTデータ破損	いいえ
270	MSI PT データ破損	いいえ
271	MSI MRIFデータ破損	いいえ
272	内部データパスのエラー	はい
273	IOMMU MSIライト・アクセス・フォールト	はい
274	第1/第2ステージPTデータ破損	いいえ

275 から 2047 までの CAUSE エンコーディングは将来の標準使用のために予約されており、 2048 から 4095 までのエンコーディングはカスタム使用のために指定されている。表 11 に指定されていない 0 から 275 までのエンコーディングは、将来の標準使用のために予約されている。

フォルト状態によって有効なデバイスコンテキストが見つからない場合、そのようなフォルトを報告するために想定されるDTF値は0である。

TTYPフィールドは、受信トランザクションのタイプを報告する。

表12.故障記録 TTYP フィールドのエンコーディング

TTYP	説明
0	なし。インバウンドトランザクションに起因しない障害。
1	実行トランザクションの未翻訳読み取り
2	未翻訳の読み取りトランザクション
3	未翻訳の書き込み/AMOトランザクション
4	予約
5	トランザクション実行のための翻訳された読み取り
6	翻訳された読み取りトランザクション
7	翻訳された書き込み/AMOトランザクション
8	PCIe ATS翻訳リクエスト
9	PCIeメッセージリクエスト
10 - 31	予約
31 - 63	カスタム仕様

TTYP が IOVA を持つトランザクションの場合、iotval に報告される。TTYP が PCIe メッセージ・リクエストの場合、メッセージ・コードが iotval に報告される。TTYP が 0 の場合、iotval と iotval2 フィールドに報告される値は、CAUSE で定義されているとおりです。

IOVAは仮想ページ番号（VPN）とページ・オフセットに分割される。VPN はアドレス変換プロセスによって物理ページ番号（PPN）に変換されますが、ページ・オフセットはこのプロセスには必要ありません。同様に、IOMMU は iotval2 内の GPA のページオフセットを 0 と報告することがある。

DIDはトランザクションのdevice_idを保持する。PVが0の場合、PIDとPRIVは0である。PVが1の場合、PIDはトランザクションのprocess_idを保持し、トランザク ションの特権がSupervisorの場合、PRIVビットは1である。

CAUSE がゲストページフォールトの場合、ゼロ拡張ゲスト物理アドレスのビット 63:2 が iotval2[63:2] で報告されます。iotval2 のビット 0 が 1 の場合、ゲストページフォールトは第 1 段階アドレス変換の暗黙のメモリアクセスによって発生しました。iotval2 のビット 0 が 1 で、暗黙のアクセスが書き込みだった場合、iotval2 のビット 1 が 1 に設定され、それ以外の場合は 0 に設定されます。

iotval2のビット1は、実装がA/Dビットのハードウェア更新をサポートし、暗黙的メモリアクセスが第1段ページテーブルのAおよび/またはDを自動的に更新しようとした場合に設定される。第1段アドレス変換のための他の全ての暗黙的メモリ アクセスは読み出されます。A/Dビットのハードウェア更新が実装されていない場合、*書き込み*

ケースは決して発生しない。

セカンドステージが Bare でない場合、プロセスコンテキストを見つけるための PDT エントリの読み出しのためのメモリアクセスは、ファーストステージのアドレス変換のための暗黙のメモリアクセスである。ゲスト・ページ・フォールトが PDT エントリを読み込むための暗黙のメモリ・アクセスによって発生した場合、そのメモリ・アクセスは第一段階のアドレス変換のための暗黙のメモリ・アクセスになる。

> エントリーの場合、iotval2のビット0は1と報告され、ビット1は0と報告される。

フォールト・キューが満杯であったり、IOMMUがキュー・メモリにアク セスしようとしたときにアクセス・フォールトが発生したりといったエラー状 態のために、IOMMUはフォールト・キューを通じてフォールトを報告できな い場合がある。メモリにマップされたフォルト・コントロール・ステータス・レジスタ（fqcsr）は、このようなフォルトに関する情報を保持する。フォールト・キュー・フル状態が検出されると、IOMMUはfqcsrにフォールト・キュー・オーバーフロー（fqof）ビットをセットする。IOMMUがフォールト・キュー・メモリへのアクセスでフォールトに遭遇した場合、IOMMUはfqcsrにフォールト・キュー・メモリ・アクセス・フォールト（fqmf）ビットをセットする。fqcsrにいずれかのエラービットがセットされている間、IOMMUはフォルトの原因となったレコードと、それ以降のすべてのフォルトレコードを破棄する。fqcsr のエラー・ビットが 1 のとき、またはフォールト・キューに新しいフォールト・レコードが生成されたとき、 フォールト・キューからの割り込みが有効になっていれば、ipsr にフォールト割り込みペンディング（fip）ビットが設定される（fqcsr.fie が 1 ）。

IOMMUは、同一のフォルトを検出したものとして、複数のリクエストを識別すること ができる。そのような場合、IOMMUはそれらのフォルトのそれぞれを個別に報告してもよいし、 一つを含むリクエストのサブセットに対してフォルトを報告してもよい。

3.3. ページ・リクエスト・キュー（PQ）

ページ・リクエスト・キューは、PCIe ATS の "ページ・リクエスト "および "ストップ・マーカ "メッセージ [1]をソフトウェアに報告するために使用されるメモリ内キューのデータ構造である。このメモリ内キューのベースPPNとキューのサイズは、ページ・リクエスト・キュー・ベース（pqb）と呼ばれるメモリ・マップド・レジスタに設定される。各ページ・リクエスト・レコードは16バイトである。

pqtは、IOMMUによって次のページ要求メッセージが書き込まれるキューへのインデックスを保持する。メッセージを書き込んだ後、IOMMUはpqtを1進める。

キューの先頭は、pqhと呼ばれるソフトウェア制御の読み書き可能なメモリマップレジスタに存在する。pqhは、次のページ要求メッセージがソフトウェアによって受信されるキューのインデックスを保持する。メッセージを処理した後、ソフトウェアは、処理されたメッセージの数だけpqhを進める。

pqh == pqtならば、ページ要求キューは空である

。pqt == (pqh - 1)の場合、ページ要求キューは満

杯である。

IOMMUは、キューが無効になっていたり、キューが一杯になっていたり、キュー・ メモリにアクセスしようとしたときにIOMMUがアクセス・フォールトに遭遇 したりといったエラー状態のために、キューを通して "ページ要求 "メッセージを 報告できないことがある。このようなフォールトに関する情報を保持するために、メモリ・マップされたページ・リクエスト・キュー・コントロール・アンド・ステータス・レジスタ（pqcsr）が使用される。ページ・キューが一杯になると、ページ・リクエスト・キュー・オーバーフロー（pqof）ビットがpqcsrにセットされる。IOMMUがキュー・メモリにアクセスする際にエラーが発生した場合、 pqcsrにページ・リクエスト・キュー・メモリ・アクセス・フォールト(pqmf)ビットがセットされる。pqcsr にエラービットがセットされている間、IOMMU はエラービットがセットされる原因となったメッセー ジを含め、それ以降のすべての「ページ要求」メッセージを破棄する。応答を必要としない "ページ要求 "メッセージ、すなわち "PRG 内最後の要求 "フィール ドが 0 のメッセージは、静かに破棄される。応答が必要な「ページ要求」メッセージ、すなわち、「PRG 内最後の要求」フィー ルドが 1 に設定され、「ストップマーカー」メッセージではないものは、セクション 2.7 で規定されているように、IOMMU が生成した「ページ要求グループ応答」メッセージによって自動完了することができる。

pqcsrのエラービットが1であるとき、または新しいメッセージがキューに生成されたとき、ページ・リクエスト・キューからの割り込みが有効になっていれば、ページ・リクエスト・キュー割り込み保留（pip）ビットがipsrに設定される（すなわち、pqcsr.pieが1である）。

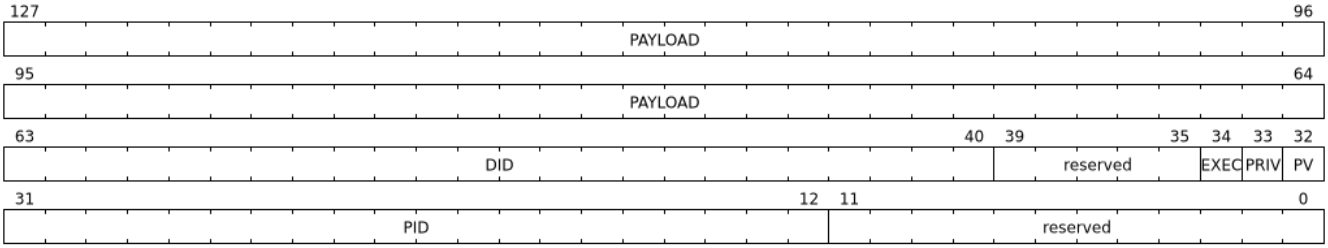


図33.ページ・リクエスト・キュー・レコード

DID フィールドはメッセージの要求者 ID を保持する。PID フィールドは PV が 1 の場合に有効であり、メッセージの PASID を報告する。メッセージに PASID がなかった場合、PRIV は 0 に設定され、そうでなければ TLP の「Privilege Mode Requested」ビットを保持する。EXEC ビットは、メッセージに PASID がなかった場合は 0 にセットされ、そうでない場合は TLP からの "Execute Requested" ビットを報告する。ページ要求」メッセージのペイロード（メッセージのバイト 0x08 から 0x0F）は PAYLOAD フィールドに保持される。R と W が共に 0 で L が 1 の場合、メッセージは "Stop Marker "である。

ページ・リクエスト・キューのレコードは2つの64ビット・ダブルワードとして解釈される。メモリ上の各ダブルワードのバイトオーダー、リトルエンディアンまたはビッグエンディアンは、fctl.BE（セクション5.4）によって決定されるエンディアンになる。

PAYLOADはメッセージ・ボディを保持し、そのフィールドはPCIe仕様[1]で規定されている通りである。そのフィールドは
PAYLOADフィールドの書式は以下の通り：

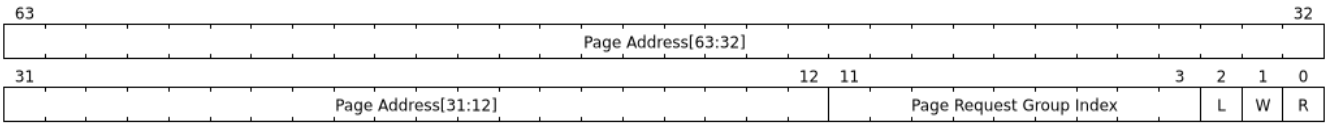


図34.ページ要求」メッセージのPAYLOAD

第4章.デバッグ・サポート

ソフトウェア・デバッグをサポートするために、IOMMU は、ソフトウェアが IOMMU にアドレス変換を要求するた め の オプションのレジスタ・インタフェースを提供することができる。このインタフェースは、ソフトウェアが IOVA をプログラムするために使用する変換要求レジスタと、 IOVA を未変換要求として変換するために必要なその他の入力レジスタから構成される。処理が正常に完了した場合、翻訳結果は翻訳応答レジスタを通して報告される。障害が発生して処理が停止した場合、その障害は通常通り障害キューで報告され、翻訳応答レ ジスタは障害インジケータで更新される。IOVA が仮想割込みファイル(セクション 2.1.3.6)であると判断され、対応する MSI PTE が MRIF モードである場合、プロセスは停止し、"Transaction type disallowed" (cause = 260)フォルトを報告する。

この目的のために IOVA を翻訳するプロセスが起動された場合、IOMMU は、第一段階の PTE、第二段階の PTE、DDT エントリ、PDT エントリ、または翻訳プロセスのためにアクセスされた MSI PTE を IOATC にキャッシュしてもしなくてもよい。IOMMU は、IOATC に既にキャッシュされているかもしれない PTE またはディレクトリ構造エントリを使用することが許される。IOMMU がサポートしている場合、IOMMU は、変換処理に使用される PTE の Accessed (A) および/または Dirty (D) ビットを更新することができる。IOMMU が HPM を実装している場合、HPM カウンタは IOMMU によって通常通り更新される。HPM でのカウントのために、これらのリクエストは未翻訳リクエストとして扱われる。

翻訳要求インターフェースは、以下の64ビットWARLレジスタで構成される：

- tr_req_iova (5.24節)
- tr_req_ctl (5.25節)

トランスレーション・レスポンス・インターフェースは、64ビットのROレジスタ tr_response（セクション5.26）で構成される。
次にtr_req_ctlレジスタが書き込まれる。Go/Busy` ビットは、レジスタに有効なリクエストがあることを示すために tr_req_ctl にセットされる。Go/Busyビットはリード・ライト・スティッキー(RWS)ビットであり、一度セットされるとレジスタの書き込みによってクリアされることはない。Go/Busyビットは、プロセスが完了したとき（成功したとき、またはフォルトが発生したとき）、IOMMUによって0にクリアされる。Go/Busy ビットが 1 から 0 になると、tr_response レジスタに応答が有効である。

以下の場合、IOMMUの動作はUNSPECIFIEDである：

- Go/Busyビットが1のとき、tr_req_iovaまたはtr_req_ctlが変更される。
- ddtp.iommu_mode などの IOMMU コンフィギュレーションが変更される。

このデバッグインターフェイスを介した変換要求を完了するまでの時間は UNSPECIFIED であるが、有限で

あることが要求される。もしIOMMUが、このレジスタインターフェイスを通して要求がなされたとき、IOブリッジからの変換要求に対応しているなら、要求を完了するまでの時間は、IOMMUがアイドルであるときよりも長くなるかもしれない。

デバッグ・インターフェースはオプションであるが、ソフトウェア・デバッグを支援し、アーキテクチャー・コンプライアンス・テストを実施するために実装することが推奨される。

第5章.メモリー・マップド・レジスタ・インターフェイス

IOMMUは、メモリマップド・プログラミング・インターフェースを提供する。各IOMMUのメモリマップド・レジスタは、物理アドレス空間のナチュラル・アラインド4KiB領域（1ページ）内に配置される。

アドレスがアクセスサイズにアライメントされていない、または複数のレジスタにまたがる、またはアクセスサイズが4バイトまたは8バイトでないレジスタアクセスに対するIOMMUの動作はUNSPECIFIEDである。IOMMUレジスタへの4バイト・アクセスは、シングル・コピー・アトミックでなければならない。IOMMU レジスタへの 8 バイト・アクセスがシングルコピー・アトミックであるかどうかは UNSPECIFIED であり、そのようなアクセスは IOMMU 内部では、2 つの別々の 4 バイト・アクセスが実行されたかのように見えるかもしれない。

8バイトIOMMUレジスタは、2つのソフトウェア・アクセスまたは2つのハードウェア・トランザクションの間で、それぞれ副作用に関するレジスタ・セマンティクスが尊重される限り、ソフトウェアがレジスタのハイハーフとローハーフに対して、2つの個別の4バイト・アクセスを実行できるように、またはハードウェアがレジスタのハイハーフとローハーフに対して、8バイト・アクセスから生じる2つの独立した4バイト・トランザクションを実行できるように定義されている。

IOMMUレジスタはリトルエンディアン・バイトオーダーである（すべてのハートがビッグエンディアンのみのシステムでも）。

IOMMUを使用するビッグエンディアン・コンフィギュレーション・ハートは、Zbb拡張で定義されたREV8バイト反転命令を実装することが期待される。REV8が実装されていない場合は、一連の命令を使ってエンディアン変換を実装することができる。

対応するケイパビリティ・レジスタ・ビットが0であることによって決定されるように、レジスタがオプショナルである場合、そのレジスタのメモリ・マップド・レジスタ・オフセットからの読み出しは0を返し、そのオフセットへの書き込みは無視される。

5.1. レジスタ・レイアウト

表13.IOMMU メモリーマップド・レジスタ・レイアウト

オフセット	名称	サイズ	説明	オプションですか?
0	能力	8	IOMMUの能力	いいえ
8	fctl	4	特徴	いいえ
12	*習慣*	4	*指定 カスタム用*	
16	ddtp	8	デバイス・ディレクトリ・テーブル・ポインタ	いいえ
24	cqb	8	コマンド・キュー・ベース	いいえ
32	cqh	4	コマンド・キュー・ヘッド	いいえ
36	cqt	4	コマンドキューテイル	いいえ
オフセット	**名称**	**サイズ**	**説明**	**オプションですか?**
40	fqb	8	故障キュー・ベース	いいえ
48	fqh	4	障害キュー・ヘッド	いいえ
52	fqt	4	障害キューの最後尾	いいえ
56	pqb	8	ページ・リクエスト・キュー・ベース	if capabilities.ATS==0
64	pqh	4	ページ・リクエスト・キュー・ヘッド	if capabilities.ATS==0
68	pqt	4	ページ・リクエスト・キュー・テール	if capabilities.ATS==0
72	cqcsr	4	コマンドキューCSR	いいえ
76	エフシーエスアール	4	障害キューCSR	いいえ
80	pqcsr	4	ページ・リクエスト・キュー CSR	if capabilities.ATS==0
84	イプサー	4	割り込み保留ステータスレジスタ	いいえ
88	イオクントフ	4	HPMカウンタのオーバーフロー	if capabilities.HPM==0
92	イオクンティン	4	HPMカウンターは以下を抑制する	if capabilities.HPM==0
96	iohpmcyclesさん	8	HPMサイクルカウンター	if capabilities.HPM==0
104	iohpmctr1-31	248	HPMイベントカウンター	if capabilities.HPM==0
352	iohpmevt1-31	248	HPMイベントセレクタ	if capabilities.HPM==0

600	tr_req_iova	8	翻訳依頼 IOVA	capabilities.DBG==0
608	tr_req_ctl	8	翻訳リクエスト制御	capabilities.DBG==0
616	tr_response	8	翻訳リクエスト応答	capabilities.DBG==0
624	予約	64	将来の使用のために予約（WPRI）	
688	*習慣*	72	*カスタム仕様*（WARL）	
760	イクベック	8	割り込み原因をベクタ・レジスタへ	いいえ
768	msi_cfg_tbl	256	MSI構成表	if capabilities.IGS==WSI
1024	予約	3072	標準使用	

5.2. リセット動作

以下のレジスタ・フィールドのリセット値は0である。

- CQcsr-CQEN、CQIE、CQON、ビジー
- fqcsr - fqen、fqie、fqon、busy
- pqcsr - pqen、pqie、pqon、busy
- tr_req_ctl.Go/ビジー
- ddtp.busy

以下のレジスタのリセット値は0である。

- イプサー

ddtp.iommu_mode フィールドのリセット値は、Off または Bare のいずれ

かでなければならない。リセット後、キャッシュ(セクション 2.8)には

有効なエントリがないこと。

iommu_modeのリセット値はOffを推奨する。

他のすべてのレジスタやフィールドのリセット値は UNSPECIFIED である。

5.3. IOMMUの能力（ケイパビリティ）

ケイパビリティ・レジスタは、IOMMU がサポートする機能を報告する読み出し専用レジスタである。各フィールドがクリアされていなければ、IOMMU にその機能があることを示す。リセット時、このレジスタは IOMMU がサポートする機能を含む。

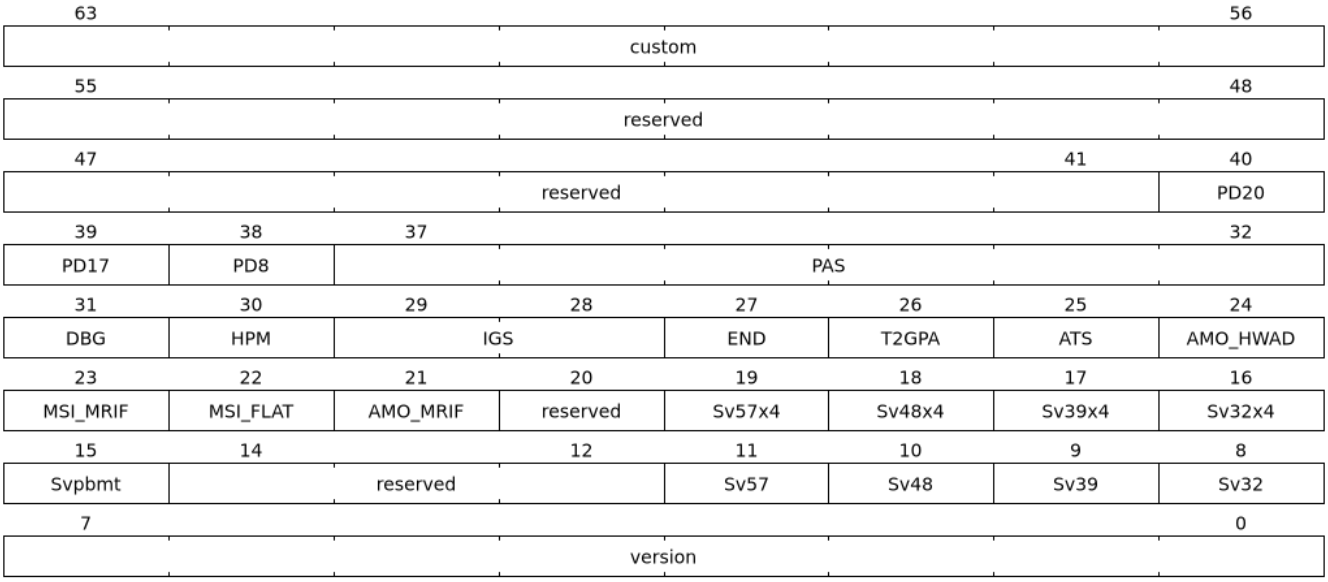


図35.IOMMU 能力レジスタ・フィールド

ビット	フィールド	属性	説明
7:0	バージョン	RO	バージョンフィールドは、IOMMUが実装する仕様のバージョンを保持する。下位ニブルは仕様のマイナーバージョン、上位ニブルは仕様のメジャーバージョンを保持する。例えば、仕様のバージョン1.0をサポートする実装は、0x10を報告する。

8	Sv32	RO	ページベースの32ビット仮想アドレッシングをサポート。
9	Sv39	RO	ページベースの39ビット仮想アドレッシングをサポート。
10	Sv48	RO	ページベースの48ビット仮想アドレッシングがサポートされる。Sv48が設定されている場合、Sv39が設定されていなければならない。
11	Sv57	RO	ページベースの57ビット仮想アドレッシングをサポートSv57が設定されている場合、Sv48が設定されていなければならない。
ビット	**フィールド**	**属性**	**説明**
14:12	控えめ	RO	標準的な使用のために予約されている。
15	スウェーデン	RO	ページベースのメモリタイプ。
16	Sv32x4	RO	第2段アドレス変換のためのページベースの34ビット仮想アドレス指定がサポートされている。
17	Sv39x4	RO	第2段アドレス変換のためのページベースの41ビット仮想アドレス指定がサポートされている。
18	Sv48x4	RO	第2段アドレス変換のためのページベースの50ビット仮想アドレス指定がサポートされている。
19	Sv57x4	RO	第2段アドレス変換のためのページベースの59ビット仮想アドレス指定がサポートされている。
20	控えめ	RO	標準的な使用のために予約されている。
21	AMO_MRIF	RO	MRIFのアトミックアップデートがサポートされる。
22	MSI_FLAT	RO	パススルーモードMSI PTEを使用したMSIアドレス変換がサポートされている。
23	MSI_MRIF	RO	MRIFモードMSI PTEを使用したMSIアドレス変換がサポートされている。
24	AMO_HWAD	RO	アクセスされたPTE（A）およびダーティ（D）ビットの原子更新がサポートされる。
25	ATS	RO	PCIeアドレス変換サービス（ATS）とページ要求インターフェイス（PRI）[1]がサポートされている。

26	T2GPA	RO	ATSの変換完了でguest-physical-addressを返すことがサポートされている。
27	終了	RO	0 の場合、IOMMU は1つのエンディアン（little または big）をサポートする。1 の場合、IOMMU は両方のエンディアンをサポートする。 エンディアンはfctlレジスタで定義される。
29:28	IGS	RO	IOMMU割り込み生成サポート。

ビット	フィールド	属性	説明
30	HPM	RO	IOMMUはハードウェア・パフォーマンス・モニターを実装している。
31	データベースグループ	RO	IOMMUは翻訳要求インターフェースをサポートしている。
37:32	PAS	RO	IOMMUがサポートする物理アドレスサイズ。
38	ピーディーエイト	RO	8ビットprocess_idの1レベルPDTをサポート。
39	PD17	RO	17ビットのprocess_idを持つ2レベルPDTをサポート。
40	PD20	RO	20ビットのprocess_idを持つ3レベルPDTをサポート。
55:41	控えめ	RO	標準使用
63:56	*習慣*	RO	*カスタム仕様*

HPMが1の時、iohpmcyclesとiohpmctr1レジスタが存在し、少なくとも32ビット幅でなければならない。

IOMMUから割り込みを発生させる方法として、MSIまたはWSIの少なくとも1つがサポートされていなければならない。

価値	名称	説明
0	三井住友海上	IOMMUは、メッセージ・シグナル・インタラプト生成のみをサポートする。
1	世界陸上	IOMMUは、ワイヤ・シグナル・インタラプト生成のみをサポートする。
2	両方	IOMMU は MSI と WSI の両方の発生をサポートする。割り込み発生方法は fctl レジスタで定義する必要がある。
3	0	標準使用

IOMMU実装は、NAPOT翻訳連続性のためのSvnapot標準拡張をサポートしなければならない。

ハイパーバイザーはSWエミュレートされたIOMMUを提供し、ゲストが第1ステージのページテーブルを管理できるようにすることで、ゲストが制御するデバイスがアクセスするメモリをきめ細かく制御することができます。

このようなエミュレートされたIOMMUをゲストに提供するハイパーバイザーは、第2段階のアドレス変換の制御を保持し、エミュレートされたケイパビリティレジスタのSvNx4フィールドをクリアすることができる。

このようなエミュレートされたIOMMUをゲストに提供するハイパーバイザーは、IMSIC内のゲスト割り込みファイルまたはメモリ常駐割り込みファイルにMSIを指示するために使用されるMSIページテーブルの制御を保持し、エミュレートされた機能レジスタのMSI_FLATおよびMSI_MRIFフィールドをクリアすることができます。

AMO_HWAD/AMO_MRIF ビットは、デバイス主導のアトミック・メモリ操作のサポートを示さない。デバイス主導のアトミック・メモリ動作のサポートは、他の手段で発見する必要がある。

IOMMU は、高度にモジュール化され、拡張可能な機能セットを提供するように設計されているため、実装者は、アプリケーションに必要な機能セットだけを含めることができる。さらに、実装は IOMMU に独自の拡張機能を追加することができる。

IOMMUは、システム内のどのハーツでもサポートされているすべての仮想メモリ拡張をサポートしなければならない。

RISC-Vプラットフォームの仕様では、以下のようなIOMMUの機能が要求される場合があります。

これらの仕様に準拠するために、実装によって提供されなければならない。

5.4. 機能制御レジスタ（fctl）

このレジスタは、どのような実装でも読み取り可能でなければならない。ある実装では、レジスタの1つ以上のフィールドを書き込み可能にして、そのフィールドで制御される機能の有効化または無効化をサポートすることができる。

IOMMUがOFFでないとき（すなわちddtp.iommu_mode != Offのとき）にソフトウェアが機能を有効または無効にした場合、IOMMUの動作はUNSPECIFIEDとなる。

IOMMUのインメモリキューが有効になっているときに、ソフトウェアがその機能を有効または無効にした場合（すなわち、cqcsr.cqon/cqen == 1、fqcsr.fqon/cqen == 1、またはpqcsr.pqon/pqen == 1）、IOMMUの動作はUNSPECIFIEDとなる。

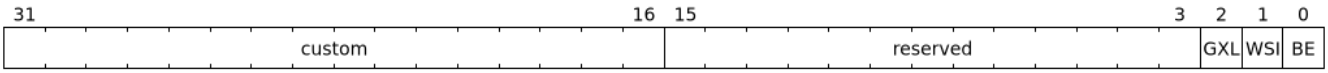


図36.機能制御レジスタ・フィールド

ビット	フィールド	属性	説明
0	BE	ワール	0のとき、表7に規定されるメモリ常駐データ構造へのIOMMUアクセスとメモリ内キューへのアクセスはリトルエンディアン・アクセスとして実行され、1のときはビッグエンディアン・アクセスとして実行される。
1	世界陸上	ワール	1の場合、IOMMU割り込みはワイヤ・シグナル割り込みとしてシグナルされ、それ以外の場合はメッセージ・シグナル割り込みとしてシグナルされる。
2	ジーエックスエル	ワール	表2に定義されているように、ゲストの物理アドレスに使用できるアドレス変換スキームを制御します。
15:3	控えめ	WPRI	標準的な使用のために予約されている。
31:16	*習慣*	WPRI	*カスタム仕様。*

5.5. デバイス・ディレクトリ・テーブル・ポインタ（ddtp）

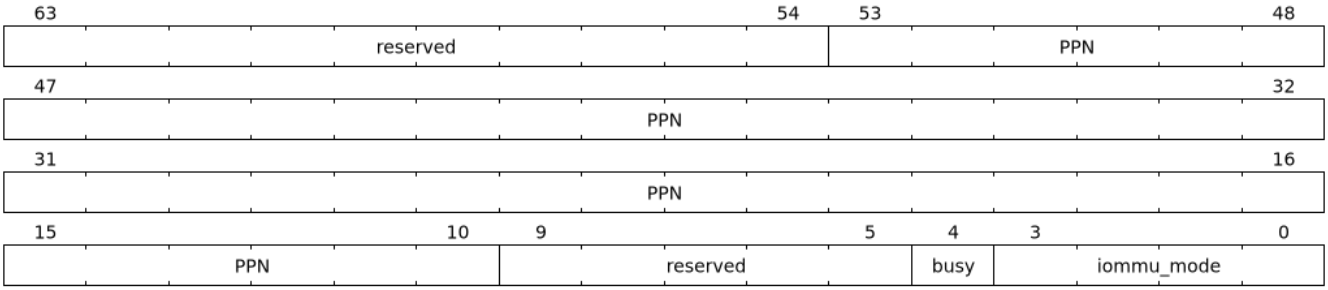


図37.デバイス・ディレクトリ・テーブル・ポインタ・レジスタのフィールド

ビット	フィールド	属性	説明
3:0	アイオムモード	ワール	IOMMUは以下のモードに設定できる：
4	忙しい	RO	ddtpへの書き込みは、IOMMUが書き込みに同期して発生しないかもしれない多くの動作を実行することを要求するかもしれない。ddtpによって書き込みが観測されると、ビジービットは1に設定される。ビジービットが1のとき、ddtpへの追加書き込みの動作はUNSPECIFIEDである。ある実装では2回目の書き込みを無視するかもしれないし、他の実装では2回目の書き込みによって決定される動作を実行するかもしれない。ソフトウェアはddtpに書き込む前にビジービットが0であることを確認しなければならない。 ビジー・ビットが0を読めば、IOMMUはddtpへの前回の書き込みに関連する動作を完了したことになる。 これらの動作を同期的に完了できる IOMMU は、このビットをハードワイヤで 0 に設定することができる。
9:5	控えめ	WPRI	標準使用
53:10	ピーピーエヌ	ワール	device-directory-tableのルートページのPPNを保持する。

63:54	控えめ	WPRI	標準使用

capabilities.MSI_FLATが1の場合、デバイスコンテキストのサイズは64バイト、それ以外の場合は32バイトである。

iommu_mode が Bare または Off の場合、PPN フィールドは don't-care となる。Bare モードの場合は、Untranslated のみ。

価値	名称	説明
0	オフ	IOMMUはインバウンドのメモリトランザクションを許可しない。
1	裸	トランスレーションもプロテクションもない。インバウンドのメモリアクセスはすべて通過する。
2	1LVL	1レベル・デバイス・ディレクトリー・テーブル
3	2LVL	2レベル・デバイス・ディレクトリー・テーブル
4	3LVL	3レベル・デバイス・ディレクトリー・テーブル
5-13	控えめ	標準的な使用のために予約されている。
14-15	カスタム	カスタム仕様。

リクエストは許可されている。翻訳リクエスト、翻訳リクエスト、PCIeメッセージトランザクションはサポートされない。

すべての IOMMU は、Off および Bare モードをサポートしなければならない。IOMMU は、ディレクトリテーブルレベルとデバイスコンテキスト幅のサブセットをサポートすることが許される。最低でもいずれかのモードをサポートしなければならない。

iommu_mode フィールドの値が Off に変更された場合、IOMMU は、IOMMU に接続されたデバイスからのインフライト・トランザクションが iommu_mode フィールドの古い値に適用されるコンフィギュレーションで処理されること、および IOMMU によって既に処理されたデバイスからのすべてのトランザクションと以前の要求が、プラットフォーム内のすべての RISC-V ハー ト、デバイス、および IOMMU によって観察できるように、グローバルな順序付けポイントにコミットされることを保証します。

iommu_mode の直前の値が Off または Bare でない場合に iommu_mode を 1LVL、2LVL、または 3LVL に書き込んだ場合の IOMMU の動作は UNSPECIFIED である。DDT レベルを変更するには、まず IOMMU を Bare または Off 状態に遷移させる必要がある。

IOMMU がベア・オブ・オフ状 態 に 遷移するとき、IOMMU はページテーブル、DDT、PDT などのメモリ内データ構造からキャッシュされた情報を保持することがある。キャッシュされたエントリを無効にするには、適切な無効化コマンドを使用しなければならない。

RV32では、レジスタの下位32ビット（22ビットのPPNと4ビットのiommu_mode）のみを書き込む必要がある。

5.6. コマンドキューベース（cqb）

この64ビットのレジスタ（RW）は、コマンドキューのルートページのPPNとキュー内のエントリ数を保持する。各コマンドは16バイトである。

cqcsr.busyビットまたはcqonビットが1のときにcqbを書き込んだときのIOMMUの動作は未定義である。ソフトウェアが推奨するcqbを変更するシーケンスは、まずcqenをクリアしてコマンドキューを無効にし、cqcsr.busyとcqonの両方が0になるのを待ってからcqbを変更することである。cqbへの書き込み後のcqtのビット31:cqb.LOG2SZのステータスは0であり、cqtのビットcqb.LOG2SZ-1:0は有効だがそれ以外はUNSPECIFIEDの値となる。

図38. コマンド・キュー・ベース・レジスタ・フィールド

ビット	フィールド	属性	説明
4:0	LOG2SZ-1	ワール	LOG2SZ-1フィールドは、コマンド待ち行列のエントリ数を2から1を引いた値で保持する。値 0 は、2 エントリのキューを示す。各IOMMUコマンドは16バイトである。コマンド・キューが256エントリ以下の場合、キューのベース・アドレスは常に4-KiBにアライメントされる。コマンド・キューが256以上のエントリーを持つ場合、コマンド・キューのベース・アドレスは、当然、$2^{LOG2SZ \times 16}$にアラインされていなければならない。
9:5	控えめ	WPRI	標準使用
53:10	ピーピーエヌ	ワール	ソフトウェアが IOMMU にコマンドをキューイングするために使用するメモリ内コマンドキューのルートページの PPN を保持する。PPNによって決定されるベースアドレスが要求されるようにアライメントされていない場合、キュー内のすべてのエントリはIOMMUからUNSPECIFIEDとして見え、IOMMUがキュー内のエントリにアクセスするために計算し使用するかもしれないアドレスもUNSPECIFIEDである。
63:54	控えめ	WPRI	標準使用

RV32では、レジスタの下位32ビット（22ビットPPNと5ビットLOG2SZ-1）のみを書き込む必要がある。

5.7. コマンドキューヘッド（cqh）

この32ビット・レジスタ（RO）は、IOMMUが次のコマンドをフェッチするコマンド・キューのインデックスを保持する。

図39.コマンドキューヘッドレジスタフィールド

ビット	フィールド	属性	説明
31:0	インデックス	RO	IOMMU が次のコマンドを取得するコマンドキューのインデックスを保持する。

5.8. コマンドキューテイル (cqt)

この 32 ビット・レジスタ（RW）は、ソフトウェアが IOMMU に対して次のコマンドをキューイングするコマンド・キューのインデックスを保持する。

図40.コマンド・キュー末尾レジスタ・フィールド

ビット	フィールド	属性	説明
31:0	インデックス	ワール	ソフトウェアが IOMMU の次のコマンドをキューイングするコマンドキューのインデックスを保持する。LOG2SZ-1:0ビットのみ書き込み可能。

5.9. フォールト・キュー・ベース（fqb）

この64ビットレジスタ（RW）は、フォルトキューのルートページのPPNとキュー内のエントリ数を保持する。各故障レコードは 32 バイトである。

fqcsr.busyまたはfqonビットが1のときにfqbを書き込んだときのIOMMUの動作は未定義である。ソフトウェアが推奨する fqb の変更手順は、まず fqen をクリアしてフォールトキューを無効にし、fqcsr.busy と fqon の両方が 0 になるのを待ってから fqb を変更する。fqbへの書き込み後のfqhのビット31:fqb.LOG2SZのステータスは0であり、fqhのビットfqb.LOG2SZ-1:0は有効だがそれ以外はUNSPECIFIEDの値となる。

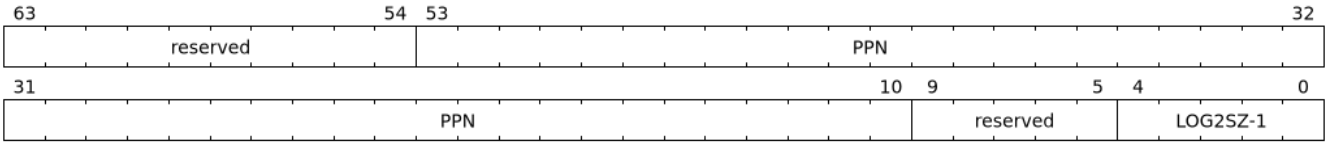


図41. フォールト・キュー・ベース・レジスタのフィールド

ビット	フィールド	属性	説明
4:0	LOG2SZ-1	ワール	LOG2SZ-1フィールドは、フォールトキューのエントリ数をlog-to-base-2から1を引いた値で保持する。値 0 は、2 エントリのキューであることを示す。各 fault レコードは 32 バイトである。fault-queue のエントリ数が 128 以下の場合、キューのベースアドレスは常に 4-KiB にアライメントされる。フォールト・キューに128以上のエントリがある場合、フォールト・キューのベース・アドレスは、当然ながら2^{LOG2SZ} x 32にアラインされていなければならない。.
9:5	控えめ	WPRI	標準使用
53:10	ピーピーエヌ	ワール	IOMMU がフォールトレコードをキューイングするために使用するメモリ内フォールトキューのルートページの PPN を保持する。PPN によって決定されるベースアドレスが要求されるようにアライメントされていない場合、キュー内のすべてのエントリは IOMMU に UNSPECIFIED として表示され、IOMMU がキュー内のエントリにアク

			セスするために計算し使用するアドレスも UNSPECIFIED となる。
63:54	控えめ	WPRI	標準使用

を書く必要がある。
RV32では、レジスタの下位32ビット（22ビットのPPNと5ビットのLOG2SZ-1）のみ。

5.10. フォルトキューヘッド（fqh）

この 32 ビット・レジスタ（RW）は、ソフトウェアが次のフォルト・レコードを取得するフォルト・キューのインデックスを保持する。

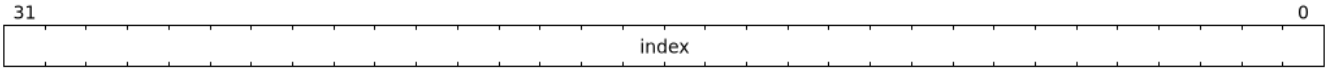


図42. フォルト・キュー・ヘッドのレジスタ・フィールド

ビット	フィールド	属性	説明
31:0	インデックス	ワール	ソフトウェアが次のフォルトレコードを読み出すフォルトキューのインデックスを保持する。LOG2SZ-1:0ビットのみ書き込み可能。

5.11. フォールト・キュー・テール（fqt）

この32ビット・レジスタ（RO）は、IOMMUが次のフォルト・レコードをキューに入れるフォルト・キューのインデックスを保持する。

図43. フォルト・キュー・テール・レジスタのフィールド

ビット	フィールド	属性	説明
31:0	インデックス	RO	IOMMU が次の fault レコードを書き込む fault-queue のインデックスを保持する。

5.12. ページ・リクエスト・キュー・ベース (pqb)

この64ビットのレジスタ（WARL）は、ページ・リクエスト・キュー のルート・ページのPPNとキュー内のエントリ数を保持する。各「ページ・リクエスト」メッセージは16バイトである。

pqcsr.busyまたはpqonビットが1のときにpqbを書き込んだときのIOMMUの動作は不明である。ソフトウェアが推奨するpqbを変更するシーケンスは、まずpqenをクリアしてページ・リクエスト・キューを無効にし、pqcsr.busyとpqonの両方が0になるのを待ってからpqbを変更することである。pqbへの書き込み後のpqhのビット31:pqb.LOG2SZのステータスは0であり、pqhのビットpqb.LOG2SZ-1:0は有効だがそれ以外はUNSPECIFIEDの値となる。

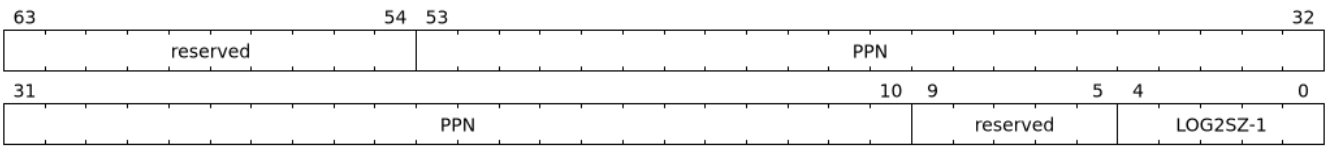


図44. ページ・リクエスト・キュー・ベース・レジスタ・フィールド

ビット	フィールド	属性	説明
4:0	LOG2SZ-1	ワール	LOG2SZ-1フィールドは、log-to-base-2から1を引いた値として、ページ要求 キューのエントリ数を保持する。値0は2エントリーのキューを示す。各ページリクエストは16バイトである。ページリクエストキューのエントリが256以下の場合、キューのベースアドレスは 常に4-KiBにアラインされる。ページリクエストキューが256以上のエントリーを持つ場合、ページリクエストキューのベースアドレスは当然$2^{LOG2SZ \times 16}$にアラインされなければならない。
ビット	フィールド	属性	説明
9:5	控えめ	WPRI	標準使用
53:10	ピーピーエヌ	ワール	IOMMU が "Page Request" メッセージをキューイングするために使用するメモリ内 page-request- queue のルートページの PPN を保持する。PPN によって決定されるベースアドレスが要求されるように整列されていない場合、キュー内のすべてのエントリは IOMMU から UNSPECIFIED として見え、IOMMU がキュー内のエントリにアクセスするために計算し使用するアドレスも UNSPECIFIED となる。
63:54	控えめ	WPRI	標準使用

RV32では、レジスタの下位32ビット（22ビットPPNと5ビットLOG2SZ-1）のみを書き込む必要がある。

5.13. ページ・リクエスト・キュー・ヘッド（pqh）

この32ビット・レジスタ（RW）は、ソフトウェアが次のページ要求をフェッチするページ要求キューのインデックスを保持する。

図45.ページ・リクエスト・キュー・ヘッド・レジスタ・フィールド

ビット	フィールド	属性	説明

31:0	インデックス	ワール	ソフトウェアが次の「ページ要求」メッセージを読み出すページ要求キューのインデックスを保持する。LOG2SZ-1:0ビットのみ書き込み可能。

5.14. ページ・リクエスト・キュー・テール (pqt)

この32ビットレジスタ（RO）は、IOMMUが次のページ要求を書き込むページ要求キューのインデックスを保持する。

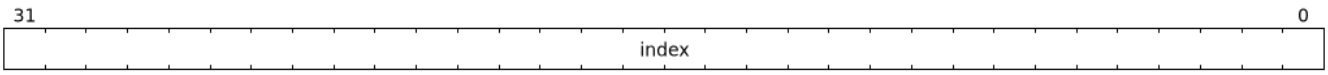


図46. ページ・リクエスト・キュー・テール・レジスタ・フィールド

ビット	フィールド	属性	説明
31:0	インデックス	RO	IOMMUが次の "Page Request "メッセージを書き込むpage-request-queueへのインデックスを保持する。

5.15. コマンドキューCSR (cqcsr)

この32ビット・レジスタ（RW）は、コマンド・キューの動作制御と状態報告に使用される。

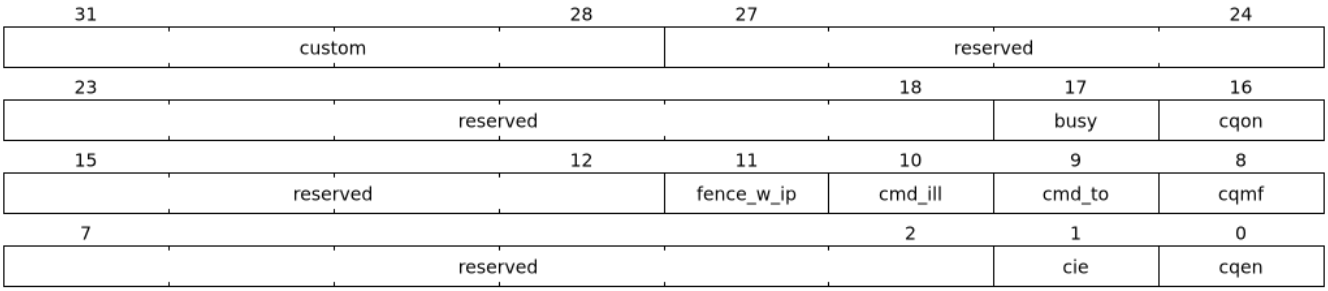


図47. コマンドキューCSRレジスタフィールド

ビット	フィールド	属性	説明
0	クイーン	RW	コマンドキューイネーブルビットは、1にセットされるとコマンドキューを有効にする。 cqenを0から1に変更すると、cqhレジスタとcqcsrビットcmd_ill,cmd_to,cqmf,fence_w_ipが0に設定される。cqenを1に設定してからコマンド・キューがアクティブになるまでに時間がかかることがある。コマンド・キューがアクティブになると、cqonビットは1を読み出す。 cqenが1から0に変更されると、コマンド・キューから既にフェッチされたコマンドが処理され、或いはコマンド・キューからの未処理の暗黙のロードがあるまで、コマンド・キューは（ビジーがアサートされた状態で）アクティブのままとなる可能性がある。コマンド・キューがオフになると、cqonビットは0を読み出す。 cqonビットが0を読み出すと、IOMMUは、コマンド・キューへの暗黙的なメモリ・アクセスがインフライトしていないことを保証し、コマンド・キューは、キュー・メモリへの新たな暗黙的なロードを生成しない。
1	三枝	RW	Command-queue-interrupt-enable ビットは、1に設定するとコマンド・キューからの割り込み発生を有効にする。
7:2	控えめ	WPRI	標準使用
8	cqmf	RW1C	コマンド・キュー・アクセスがメモリー・フォールトにつながる場合、コマンド・キュー・メモリー・フォールト・ビットは1にセットされ、このビットがクリアされるまでコマンド・キューはストールする。コマンド処理を再び有効にするには、ソフトウェアがこの

			ビットに1を書き込んでクリアする必要がある。
9	cmd_to	RW1C	コマンドの実行がタイムアウトにつながった場合（例えば、デバイスATCを無効にするコマンドは、完了を待ってタイムアウトになることがある）、コマンド・キューはcmd_toビットをセットし、コマンド・キューからの処理を停止する。コマンド処理を再び有効にするには、ソフトウェアがこのビットに1を書き込んでクリアする必要がある。
10	cmd_ill	RW1C	不正な、あるいはサポートされていないコマンドがコマンド・キューによってフェッチされ、デコードされた場合、コマンド・キューはcmd_illビットをセットし、コマンド・キューからの処理を停止する。コマンド処理を再び有効にするには、ソフトウェアがこのビットに1を書き込んでクリアする必要がある。
ビット	**フィールド**	**属性**	**説明**
11	フェンス	RW1C	ワイヤシグナルインタラプトのみをサポートする IOMMU は、IOFENCE.C コマンドの完了を示すために fence_w_ip ビットをセットする。IOFENCE.C の完了時に割り込みを再び有効にするには、ソフトウェアがこのビットを 1 を書き込んでクリアする必要がある。このビットは、IOMMU が wire-signaled-interrupts をサポートしていないか、wire-signaled-interrupts が有効になっていない（すなわち、fctl.WSI == 0）場合に予約される。
15:12	控えめ	WPRI	標準使用
16	クコン	RO	cqonが1の場合、コマンド・キューがアクティブになる。

17	忙しい	RO	cqcsr への書き込みは、IOMMU が書き込みに同期していない多くの処理を実行することを必要とする場合がある。cqcsr が書き込みを検出すると、ビジー・ビットは 1 に設定される。 ビジービットが1の場合、cqcsrへの追加書き込みの動作は未定義である。ある実装は2回目の書き込みを無視するかもしれないし、他の実装は2回目の書き込みによって決定される動作を実行するかもしれない。 に書き込む前に、ビジー・ビットが0であることを確認する必要がある。 cqcsr. これらの動作を同期的に完了できる IOMMU は、このビットをハードワイヤで 0 に設定することができる。
27:18	控えめ	WPRI	標準使用
31:28	*習慣*	WPRI	*カスタム仕様。*

cqcsr の cmd_ill または cqmf が 1 の場合、cqh はエラーの原因となった CQ 内のコマンドを参照する。それ以前のコマンドは、完了したか、タイムアウトしたか、IOMMUによって実行が中断された可能性がある。

ソフトウェアが cmd_ill または cqmf エラーの後に CQ を再び動作可能にする場合、ソフトウェアは、正常に完了した最後の IOFENCE.C 以降に送信されたコマンドを再送信する必要がある。

cmd_to ビットは、IOFENCE.C コマンドが、タイムアウトが指定された 1 つ以上の前のコマンドがタイムアウトしたが、 IOFENCE.C より前の他のすべてのコマンドが完了したことを検出したときにセットされる。cmd_to が 1 のとき、 cqh はタイムアウトを検出した IOFENCE.C コマンドを参照する。

コマンド・キューが空であることは、コマンド・キューからフェッチされたすべてのコマンドが完了したことを意味しない。コマンド・キューを無効にするように要求された場合、実装は、すでにフェッチされたコマンドを完了するか、それらのコマンドの実行を中止することができる。ソフトウェアは、IOFENCE.C コマンドを使用して、コマンド・キューをオフにする前に、以前のコマンドがすべてコミットされるのを待つ必要があります。

5.16. フォールトキューCSR（fqcsr）

この 32 ビット・レジスタ（RW）は、フォールト・キューの動作制御と状態報告に使用される。

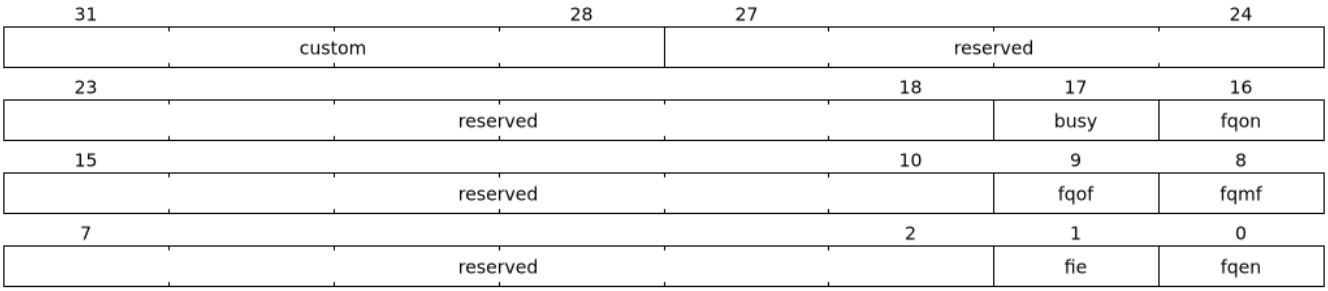


図48. フォルトキュー CSR レジスタ・フィールド

ビット	フィールド	属性	説明
0	れん	RW	フォールトキュー・イネーブルビットは、1にセットされるとフォールトキューを有効にする。 fqenを0から1に変更すると、fqtレジスタとfqcsrビットのfqofとfqmfが0に設定される。 fqenを1に設定してからフォルトキューがアクティブになるまでに時間がかかることがある。フォルトキューがアクティブになると、fqonビットは1を読み出す。 fqen が 1 から 0 に変更されると、フォールト・キューは、インフライト・フォールト・レコ ーディングが完了するまでアクティブのまま（ビジーがアサートされたまま）である。fqonが0を読み出すと、IOMMUは、進行中のフォールト・キューへの暗黙の書き込みがなく、フォールト・キューに新しいフォールト・レコードが書き込まれないことを保証する。
1	フィー	RW	フォールト・キュー割り込みイネーブル・ビットは、1に設定されるとフォールト・キューからの割り込み発生を有効にする。
7:2	控えめ	WPRI	標準使用
8	fqmf	RW1C	フォルトキューにフォルトレコードを格納するときに IOMMU がアクセスフォルトに遭遇すると、fqmf ビットが 1 に設定される。書き込もうとしたフォルトレコードは破棄され、ソフトウェアが fqmf ビットに 1 を書き込んでクリアするまで、フォルトレコードは生成さ

			れません。
9	フ	RW1C	フォルトキューオーバーフロービットは、IOMMU がフォルトレコードをキューに入れる必要があるが、 フォルトキューが満杯（すなわち、fqt == fqh - 1）である場合に 1 に設定される。 フォルトレコードは破棄され、ソフトウェアがfqofビットに1を書き込んでクリアするまで、フォルトレコードは生成されない。
15:10	控えめ	WPRI	標準使用
16	フコン	RO	fqonが1の場合、フォールトキューはアクティブである。
ビット	**フィールド**	**属性**	**説明**
17	忙しい	RO	fqcsrへの書き込みは、IOMMUが書き込みに同期しない多くの動作を実行することを必要とする場合がある。書き込みがfqcsrによって観測されると、ビジー・ビットは1に設定される。ビジービットが 1 の場合、fqcsr への追加書き込みの動作は UNSPECIFIED である。ある実装では2回目の書き込みを無視し、他の実装では2回目の書き込みによって決定される動作を実行する。 に書き込む前に、ビジー・ビットが0であることを確認すべきである。 fqcsr。 同期的に制御を完了できる IOMMU は、このビットをハード的に 0 に配線してもよい。
27:18	控えめ	WPRI	標準使用
31:28	*習慣*	WPRI	*カスタム仕様。*

5.17. ページ・リクエスト・キューCSR (pqcsr)

この32ビット・レジスタ（RW）は、ページ・リクエスト・キューの動作制御とステータス報告に使用される。

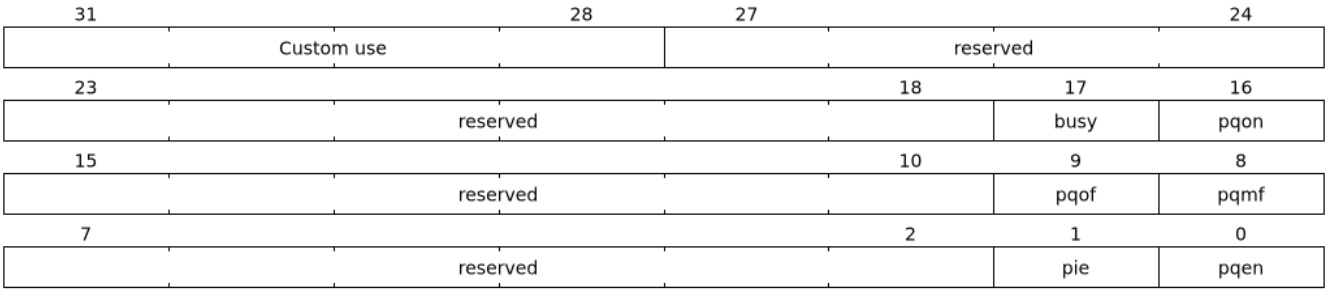


図49. ページ・リクエスト・キューCSRレジスタ・フィールド

ビット	フィールド	属性	説明
0	プーケン	RW	ページ・リクエスト有効ビットは、1にセットされるとページ・リクエスト・キューを有効にする。 pqenを0から1に変更すると、pqhレジスタとpqcsrビットのpqmfとpqofが0に設定される。pqenを1に設定してからページ・リクエスト・キューがアクティブになるまでに時間がかかることがある。ページ・リクエスト・キューがアクティブになると、pqonビットは1を読み出す。 pqenが1から0に変更されると、ページ・リクエスト待ち行列は、飛行中のページ・リクエスト書き込みが完了するまで（ビジーがアサートされた状態で）アクティブなままである可能性がある。ページ・リクエスト・キューがオフになると、pqonビットは0を読み出す。 pqonが0を読み出すと、IOMMUは、キュー・メモリへの古いインフライト暗黙の書 き込みがないことを保証し、それ以上キュー・メモリへの暗黙の書 き込みが発生しないことを保証する。 IOMMUは、セクション2.7で規定されているように、page-request-queueがオフになっているか、オフの過程にあるときに受信した"Page Request "メッセージに応答してもよい。
1	パイ	RW	page-request-queue-interrupt-enable ビットを1に設定すると、page-request-queue からの割り込み発生を有効にする。
7:2	控えめ	WPRI	標準使用
8	pqmf	RW1C	pqmfビットは、IOMMUが「ページ要求」メッセージをページ要求キューに格納する際にアクセス・フォールトに遭遇した場合に1にセットされる。 pqmf または pqof エラーの原因となった "Page Request "メッセー ジとそれ以降の "Page Request "メッセージは、ソフトウェアが pqof または pqmf ビットに 1 を書き込んでクリアするまで破棄される。 IOMMUは、セクション2.7で規定されているように、pqofまたはpqmfビットがセットされた "Page Request "メッセージ、およびこれ

			らのビットが1である間に受信したそれ以降のすべての "Page Request "メッセージに応答することができる。

ビット	フィールド	属性	説明
9	プクフ	RW1C	すなわち、IOMMUは "ページ要求 "メッセージをキューに入れる必要があるが、 ページ要求キューが満杯（すなわち、pqt == pqh - 1）である。 pqmf または pqof エラーの原因となった "Page Request "メッセー ジとそれ以降の "Page Request "メッセージは、ソフトウェアが pqof または pqmf ビットに 1 を書き込んでクリアするまで破棄される。 IOMMUは、セクション2.7で規定されているように、pqofまたはpqmfビットがセットされた "Page Request "メッセージ、およびこれらのビットが1である間に受信したそれ以降のすべての "Page Request "メッセージに応答することができる。
15:10	控えめ	WPRI	標準使用
16	プコン	RO	ページ要求がアクティブになるのは、pqonが1を読み取ったときである。
17	忙しい	RO	pqcsrへの書き込みは、IOMMUが書き込みに同期して発生しない可能性のある多くの処理を実行することを必要とする場合がある。pqcsrに書き込みが観測されると、ビジー・ビットは1にセットされる。 ビジービットが1の場合、pqcsrへの追加書き込みの動作は未定義である。実装によっては、2 回目の書き込みを無視するものもあれば、2 回目の書き込みによって決定される動作を実行するものもある。ソフトウェアはpqcsrに書き込む前にビジー・ビットが0であることを確認する必要がある。 同期的に制御を完了できる IOMMU は、このビットを 0 にハードワイヤすることができます。
27:18	控えめ	WPRI	標準使用
31:28	*習慣*	WPRI	*カスタム仕様。*

5.18. 割り込み保留ステータスレジスタ（ipsr）

この 32 ビット・レジスタ（RW1C）は、ソフトウェア・サービスを必要とする保留中の割り込みを 報告

する。レジスタの各割り込み待ちビットは、IOMMU 内の割り込みソースに対応します。レジスタの割込み保留ビットは、一度1にセットされると、ソフトウェアがその割込み保留ビットをクリアするために1を書き込むまで1のままです。

fctl.WSIが1のとき、割り込み待ちビットは対応するicvecで選択されたワイヤーを駆動する。
フィールドで割り込みを知らせる。

fctl.WSI が 0 のとき、IOMMU はメッセージを使用して割り込みを通知する。メッセージのアドレスとデータは、割り込み待ちビットに対応する icvec フィールドで選択された msi_cfg_tbl エントリから取得されます。

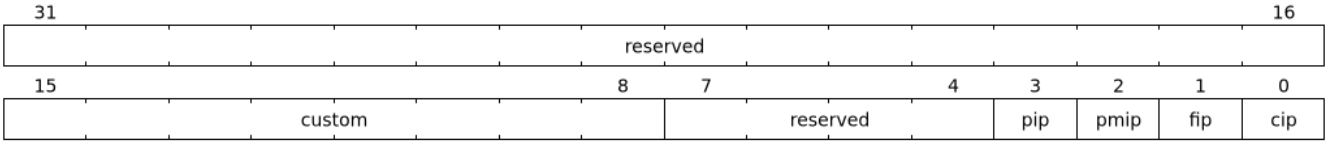


図50.割り込み待ちステータス・レジスタのフィールド

表14.割り込み保留ステータス・レジスタ・フィールド

ビット	フィールド	属性	説明
0	チップ	RW1C	cqcsr.cieが以下の場合、コマンドキュー割り込み待ちビットは1にセットされる。 が1であり、以下のいずれかが真である： • cqcsr.fence_w_ipは1。 • cqcsr.cmd_illは1。 • cqcsr.cmd_toは1。 • cqcsr.cqmf は 1 です。
1	フィップ	RW1C	fqcsr.fieが1で、以下のいずれかが真であれば、fault-queue-interrupt-pendingビットは1に設定される： • fqcsr.fqof は 1 です。 • fqcsr.fqmf は 1 です。 • FQで新記録が生まれる。
2	ピップ	RW1C	iohpmcycles または iohpmctr1-31 レジスタの OF ビットが 0 から 1 に遷移すると、performance-monitoring-interrupt-pending が 1 に設定される。
3	ピップ	RW1C	pqcsr.pieの場合、page-request-queue-interrupt-pendingは1に設定される。 が1であり、以下のいずれかが真である： • pqcsr.pqof は 1 です。 • pqcsr.pqmf は 1 です。 • PQに新しいメッセージが作成される。
7:4	控えめ	WPRI	標準使用
15:8	習慣	WPRI	カスタム仕様
31:16	控えめ	WPRI	標準使用

ipsrのビットが1の場合、そのビットに1を書き込むとビットは1→0に遷移する。そのビットをセットする条件がまだ存在する場合（[IPSR_FIELDS]を参照）、またはビットがクリアされた後にその条件が発生した場合、そのビットは0から1に再び遷移する。

5.19. パフォーマンス・モニタリング・カウンターのオーバーフロー状態（iocountovf）

パフォーマンス・モニター・カウンターのオーバーフロー・ステータスは、32ビットの読み出し専用レジスタで、以下の内容を含む。

iohpmevt1-31レジスタのOFビットのシャドウコピー。
iohpmevtXとビット0はiohpmcyclesのOFビットに対応する。

このレジスタにより、オーバーフロー割り込みハンドラ・ソフトウェアは、どのカウンタがオーバーフローしたかを迅速かつ容易に判断することができる。

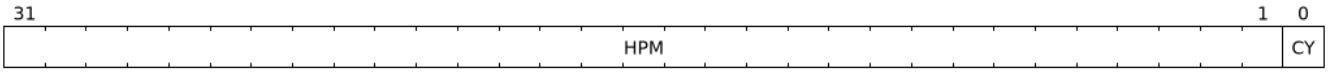


図51.パフォーマンス・モニタ・カウンタ・オーバーフロー・ステータス・レジスタのフィールド

ビット	フィールド	属性	説明
0	CY	RO	iohpmcycles.OFの影
31:1	HPM	RO	iohpmevtの影[1-31].OF

5.20. パフォーマンス・モニタリング・カウンターの抑制（iocountinh）

パフォーマンス・モニタリング・カウンタ禁止は 32 ビットの WARL レジスタで、対応するカウンタのカウントを禁止するビットを含む。ビットXがセットされるとiohpmctrXのカウントを禁止し、ビット0がセットされるとiohpmcyclesのカウントを禁止する。

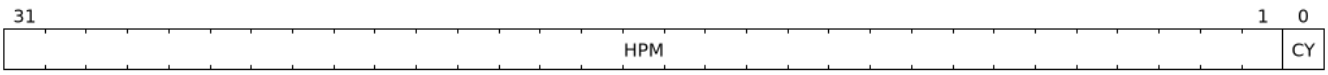


図52.パフォーマンス・モニタリング・カウンターはレジスタ・フィールドを禁止する

ビット	フィールド	属性	説明
0	CY	RW	設定されると、iohpmcyclesカウンタはカウントを禁止する。
31:1	HPM	ワール	ビットXがセットされると、iohpmctrXのイベントのカウントは禁止される。

iohpmcyclesカウンタが不要な場合は、 条件付きで禁止して消費電力を抑えることが望ましい。全てのカウンターを禁止するために1つのレジスタを提供することで、a)1つ以上のカウンターにカウントするイベントをアトミックにプログラムすることができる。
b) アトミックにサンプリングされる1つ以上のカウンタ。

5.21. パフォーマンス・モニタリング・サイクル・

カウンタ（iohpmcycles）

この64ビット・レジスタはフリー・ランのクロック・サイクル・カウンタである。関連する iohpmevt0 はない。

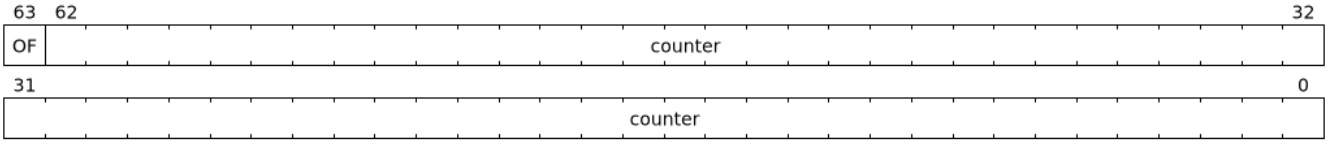


図53.パフォーマンス・モニタリング・サイクル・カウンタ・レジスタのフィールド

ビット	フィールド	属性	説明
62:0	カウンター	ワール	サイクルカウンターの値。
63	オブ	RW	オーバーフロー

OF ビットは、iohpmcycles カウンタがオーバーフローするとセットされ、ソフトウェアによってクリアされるまでセットされたままである。iohpmcycles値は符号なし値なので、オーバーフローは符号なしオーバーフローと定義される。OFビットがセットされている間、カウンタは折り返しカウントを続けるので、オーバーフロー後も情報が失われることはない。

OF ビットが 0 の時に iohpmcycles カウンタがオーバーフローした場合、ipsr.pmip ビットを 1 に設定することで HPM カウンタ・オーバーフロー割り込みが発生します。その結果、OF ビットは iohpmcycles のカウント・オーバーフロー割り込みディセーブルとしても機能します。

5.22. パフォーマンス・モニタリング・イベント・カウンタ（iohpmctr1-31）

これらのレジスタは 64 ビット WARL カウンタ・レジスタである。

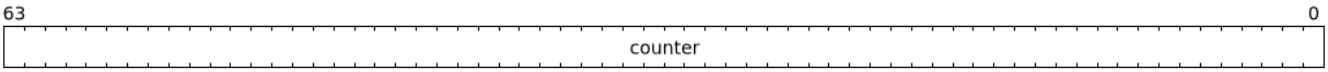


図54.パフォーマンス・モニタリング・イベント・カウンタ・レジスタのフィールド

ビット	フィールド	属性	説明
63:0	カウンター	ワール	イベントカウンタの値。

5.23. パフォーマンス・モニタリング・イベント・セレクタ（iohpmevt1-31）

これらのパフォーマンス・モニタリング・イベント・レジスタは64ビットRWレジスタである。IOMMUによって処理されたトランザクションが、カウンタにカウントするようにプログラムされたイベントを引き起こすと、カウンタはインクリメントされる。イベントのマッチングに加えて、イベントセレクタは、デバイス ID、プロセス ID、GSCID、および PSCID に基づく追加のフィルタリングでプログラムされてもよい。このようなdevice_idに基づくフィルタリングが使用される場合、照合は正確な照合または部分的な照合

になるように構成されるかもしれない。部分一致は、IDの範囲を持つトランザクションがカウンタによってカウントされることを可能にする。

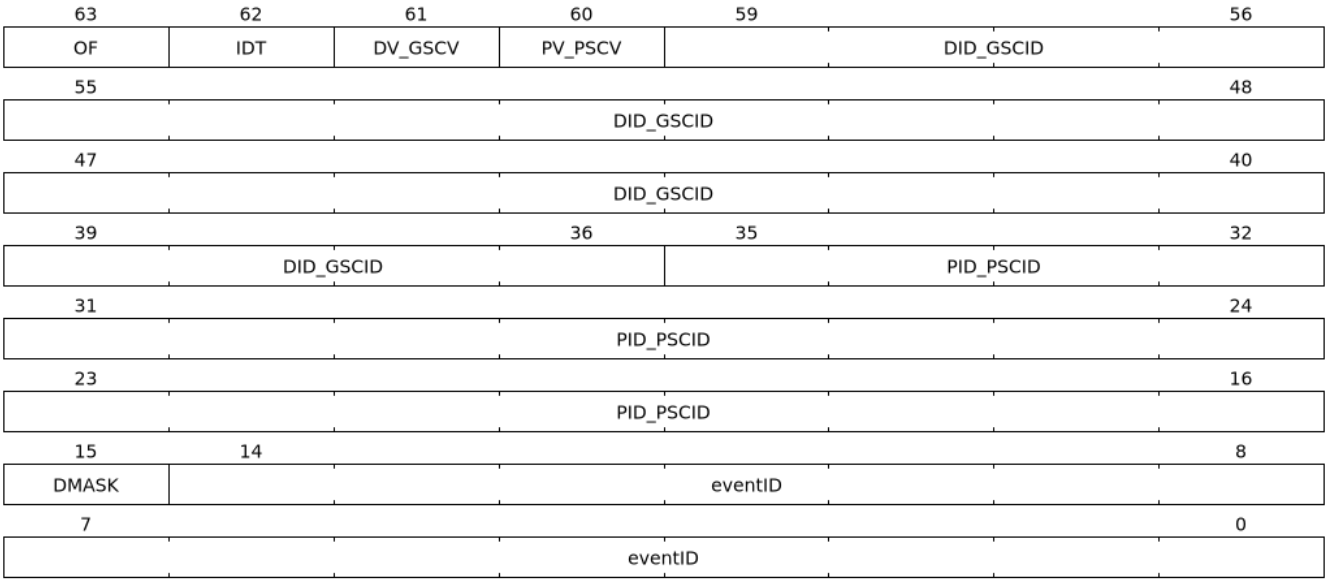


図55.パフォーマンス・モニタリング・イベント・セレクタ・レジスタのフィールド

ビット	フィールド	属性	説明
14:0	イベントID	ワール	カウントするイベントを示す。値が0の場合、イベントはカウントされない。 エンコーディング1から16383は、表17に定義された標準イベント用に予約されている。 エンコーディング16384から32767は、カスタム用に指定されている。 eventIDが変更された場合（0を含む）、カウンターはその値を保持する。
15	ディーエムエーエスケー	RW	1に設定すると、トランザクションに対してDID_GSCIDの部分一致が実行される。DID_GSCIDの下位ビットから最初の下位0ビットまで（0ビット位置そのものを含む）がマスクされる。
35:16	PID_PSCID	RW	IDTが0の場合はprocess_id、IDTが1の場合はPSCID
59:36	DID_GSCID	RW	IDTが0の場合はdevice_id、1の場合はGSCID。
60	PV_PSCV	RW	設定された場合、process_id または PSCID（フィルターIDタイプに基づく）がカウントされる。
61	DV_GSCV	RW	設定された場合、一致する device_id または GSCID（フィルターIDタイプに基づく）がカウントされる。

62	IDT	RW	フィルターIDタイプ：このフィールドはフィルタリングするIDのタイプを示す。0の場合、DID_GSCIDフィールドはdevice_idを、PID_PSCIDフィールドはprocess_idを保持する。1の場合、DID_GSCIDフィールドはGSCIDを、PID_PSCIDフィールドはPSCIDを保持する。
63	オブ	RW	オーバーフロー状態または割り込み禁止

以下の表は、IDによるフィルタリングをサポートするイベントのフィルタリングオプションをまとめたものです。

フィルタリングオプション

IDT	DV_GSCV	PV_PSCV	オペレーション
0/1	0	0	カウンターの増分。IDベースのフィルタリングはありません。
0	0	1	トランザクションが有効なprocess_idを持つ場合、process_idがPID_PSCIDと一致するとカウンタがインクリメントする。
0	1	0	device_idがDID_GSCIDに一致するとカウンタがインクリメントする。
0	1	1	トランザクションが有効なprocess_idを持つ場合、device_idがDID_GSCIDに一致し、process_idがPID_PSCIDに一致すると、カウンターはインクリメントする。
1	0	1	トランザクションが有効なPSCIDを持つ場合、カウンタは以下の場合にインクリメントする。 そのプロセスのPSCIDがPID_PSCIDに一致する。
1	1	0	GSCIDが有効でDID_GSCIDと一致すればカウンタはインクリメントする。
1	1	1	GSCIDが有効でDID_GSCIDと一致すればカウンターはインクリメントする。 PSCIDが有効でPID_PSCIDと一致する場合。

device_id または GSCID によるフィルタリングが選択され、イベントが ID ベースのフィルタリングをサポートしている場合、 DMASK フィールドを使用して部分一致を設定することができる。DMASKを1に設定すると、DID_GSCIDの部分一致がトランザクションに対して実行される。DID_GSCIDの下位ビットから最初の下位0ビットまで(0ビット位置そのものを含む)がマスクされる。

以下の例は、DMASKの使用とdevice_idによるフィルタリングを示している。

表16.IDTがdevice_idベースのフィルタリングに設定されたDMASK

ディーエムエーエスケー	DID_GSCID	コメント
0	イイ	特定のセグ:バス:デバイス:ファンコ
1	YYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYY011	seg:bus:dev - あらゆる機能
1	0111111111	seg:バス - 任意 dev:func
1	yyyyyyyyy 01111111 11111111	セグ - 任意 bus:dev:func

次の表は、カウントできる標準的なイベントの一覧である：

表17.標準イベントリスト

イベントID	イベント数	対応するIDT設定
0	カウントしない	
1	未翻訳のリクエスト	0
2	翻訳依頼	0
3	ATS 翻訳依頼	0
4	TLBミス	0/1
5	デバイス・ディレクトリ・ウォーク	0
6	プロセス・ディレクトリ・ウォーク	0
7	第1ステージのテーブル・ウォーク	0/1
イベントID	**イベント数**	**対応するIDT設定**
8	セカンドステージのテーブル・ウォーク	0/1
9 - 16383	将来の標準のために予約	-

プログラムされた IDT 設定がイベントに対してサポートされていない場合、関連するカウンタはインクリメントしない。

OF ビットは、対応する iohpmctr1-31 カウンタがオーバーフローするとセットされ、ソフトウェアによってクリアされるまでセットされたままである。iohpmctr1-31値は符号なし値なので、オーバーフローは符号なしオーバーフローと定義される。OFビットがセットされたまま、カウンタは折り返し、カウントを続けるので、オーバーフロー後に情報が失われることはないことに注意してください。

iohpmctr1-31カウンタが、関連するOFビットが0の時にオーバーフローした場合、ipsr.pmipビットを 1に設定することでHPMカウンタオーバーフロー割り込みが発生します。その結果、OF ビットは関連する iohpmctr1-31 のカウント・オーバーフロー割り込みディセーブルとしても機能します。

オーバフロー・ステータスとオーバフロー割り込みイネーブル・ビットは別個に存在しない。実際には、（OFビットをクリアすることによって）オーバーフロー割り込みの発生を有効にすることは、カウンタを開始値に初期化することと同時に行われます。一度オーバーフローしたカウンタは、再度オーバーフロー割り込みを発生させる前に、カウンタと OF ビットを再初期化する必要があります。

RV32では、iohpmevt1-31へのメモリ・マップド・ライトはレジスタの32ビット部分のみ

を変更します。以下のシーケンスを使用すると、レジスタの中間値に起因するイベントのスプリアスをカウントすることなく、レジスタを更新することができます:

- eventIDを0に設定するために、下位32ビットを書き込む。
- 上位32ビットに新しい希望値を書き込む。
- 低次の32ビットに、以下の値を含む新しい希望値を書き込む。
  eventID フィールド。

別の方法として、カウンタはまず、更新中にイベントがカウントされないように禁止され、レジスタが所望の値でプログラムされた後に禁止が解除されることもある。

capabilities.HPM が 1 の場合、この仕様に準拠するためには、サイクル・カウンタ以外に最低 1 つのプログラマブル・イベント・カウンタが必要である。イベントをサンプリングするために、時間多重化された方法で 1 つのカウンタを使用することもできるが、そのような解析は完了するまでに時間がかかる可能性がある。IOMMU は CPU MMU と異なり、IO の複数のストリームにサービスを提供し、HPM はパフォーマンス・アナリストがこれらのストリームの 1 つ以上を同時に解析するために使用することができる。通常、パフォーマンス・アナリストは IO ストリームのイベントをカウントするために 4 つのプログラマブル・カウンタを必要とする。少なくとも 2 つの IO ストリームの同時解析をサポートするには、7 つのプログラマブル・カウンタをサポートすることを推奨します。

5.24. 翻訳リクエストIOVA (tr_req_iova)

tr_req_iovaは、デバッグ用のトランスレーション・レクエスト・インターフェー スを実装するために使用される64ビットのレジスタである。このレジスタは、 capabilities.DBG == 1 のときに存在する。

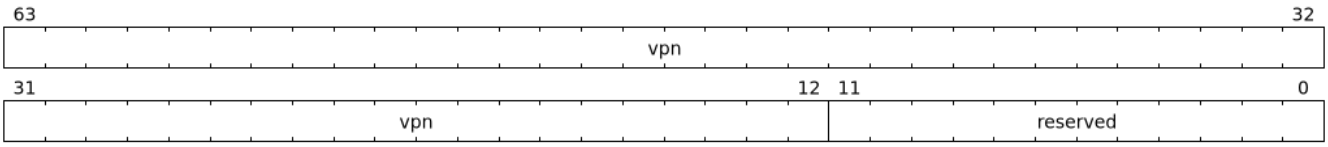


図56.翻訳要求IOVA レジスタ・フィールド

ビット	フィールド	属性	説明
11:0	控えめ	WPRI	標準使用
63:12	ブイピーエヌ	ワール	IOVAのバーチャルページ番号

5.25. 翻訳リクエスト制御 (tr_req_ctl)

tr_req_ctlは、デバッグ用トランスレーション・レクエスト・インターフェースの実装に使用される64ビットWARLレジスタである。このレジスタは、 capabilities.DBG == 1 の時に存在する。

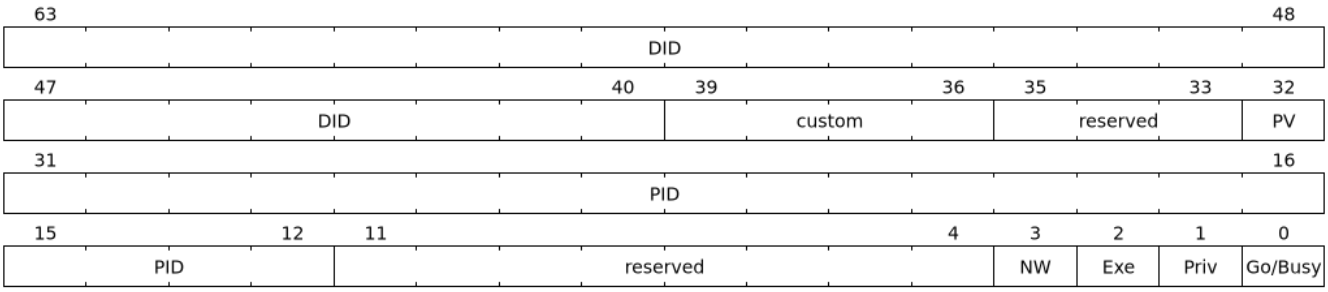


図57.翻訳要求制御レジスタ・フィールド

ビット	フィールド	属性	説明
0	ゴー/ビジー	RW1S	このビットがセットされると、有効なリクエストがtr_req_iova/tr_req_ctlレジスタはIOMMUがトランスレートするためのものである。 IOMMUは、このビットを0にクリアすることで、要求された翻訳が

			完了したことを示す。完了すると、翻訳結果はtr_responseレジスタに格納される。
1	プライベート	ワール	1に設定された場合、特権モードアクセスが要求され、そうでなければ特権モードアクセスは要求されない。
2	エクセ	ワール	1に設定されている場合、実行許可が要求され、1に設定されていない場合、実行許可は要求されない。
3	エヌダブリュー	ワール	1に設定すると、読み取りパーミッションが要求される。0に設定すると、読み取りと書き込みの両方のパーミッションが要求されます。
11:4	控えめ	WPRI	標準使用
ビット	**フィールド**	**属性**	**説明**
31:12	ピッド	ワール	PVが1の場合、このフィールドはこの翻訳リクエストのprocess_id入力を提供する。PVが0の場合、このフィールドは使用されません。
32	PV	ワール	1に設定されている場合、レジスタのPIDフィールドは有効であり、この変換リク エストのprocess_idを提供する。0に設定された場合、PIDフィールドは使用されず、process_idはこの変換要求に対して有効ではない。
35:33	控えめ	WPRI	標準使用
39:36	*習慣*	WPRI	*カスタム仕様*
63:40	DID	ワール	このフィールドは、この翻訳リクエストのdevice_idを提供する。

5.26. 翻訳レスポンス（tr_response）

tr_responseは64ビットのROレジスタで、トランスレーション・レクエスト・インターフェー スを使ってリクエストされたトランスレーションの結果を保持するために使われる。このレジスタは、capabilities.DBG == 1のときに存在する。

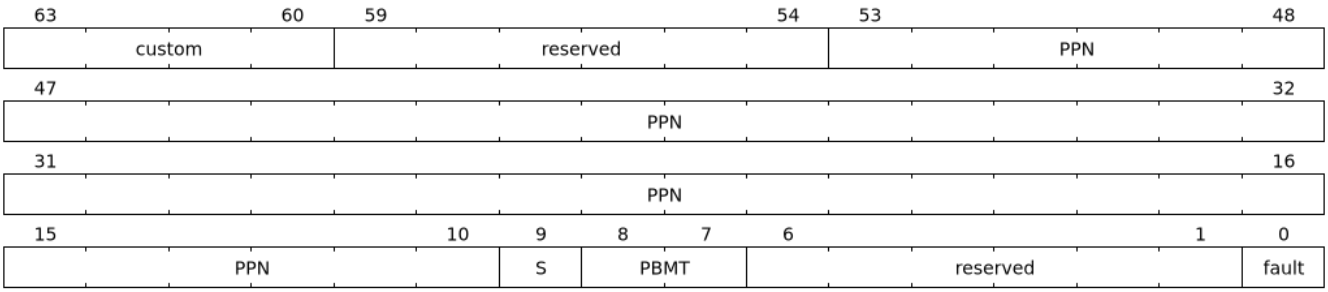


図58.翻訳レスポンス・レジスタのフィールド

ビット	フィールド	属性	説明
0	フォールト	RO	検出されたフォルトは、フォルトキューを通して報告される。
6:1	控えめ	RO	標準使用
8:7	PBMT	RO	翻訳に使用された第1ステージおよび／または第2ステージのページテーブルのPBMTフィールドを使用して、翻訳のために決定されたメモリタイプ。faultフィールドが1の場合、このフィールドの値はUNSPECIFIEDである。
9	S	RO	変換範囲サイズフィールドは、1に設定されている場合、変換が4KiBより大きい範囲に適用され、変換範囲のサイズがPPNフィールドに符号化されていることを示す。faultフィールドが1の場合、このフィールドの値はUNSPECIFIEDである。
ビット	**フィールド**	**属性**	**説明**

53:10	ピーピーエヌ	RO	faultビットが0の場合、このフィールドはtr_req_iovaのvpnを変換した結果決定されたPPNを提供する。 faultビットが1の場合、このフィールドの値はUNSPECIFIEDである 。Sビットが0の場合、変換のサイズは4KiB（1ページ）である。 Sビットが1の場合、変換の結果スーパーページが生成され、スーパーページのサイズはPPN自体にエンコードされる。ビット位置0からビット位置43までスキャンする場合、X位置で値が0である最初のビットは、スーパーページのサイズは$2^{X+1} * 4$ KiBである。 Xが0でない場合、0からX-1までのすべてのビットはそれぞれ1の値でエンコードされる。 *表18. PPNにおけるスーパーページサイズの符号化例* **ピーピーエヌ** / **S** / **サイズ** yyyy・・・yyyy yyyy yyyy yyyy yyyy / 0 / 4 KiB yyyy...yyyy yyyy 0111 / 1 / 64 KiB yyyy...yyy0 1111 1111 / 1 / 2 MiB yyyy...yy01 1111 1111 / 1 / 4 MiB
59:54	控えめ	RO	標準使用
63:60	*習慣*	RO	*カスタム仕様*

IOMMU実装は、スーパーページ変換を報告する必要はなく、可能なすべての スーパーページサイズの報告をサポートする必要もない。実装は、要求されたvpnに対応する4KiB変換を報告するか、ページテーブルで設定されたスーパーページ・サイズより小さい変換サイズを報告することが許される。

5.27. 割り込み要因対ベクトル・レジスタ（icvec）

割り込み原因をベクターにマップする。すべての原因を同じベクターにマッピングすることもできるし、原因に固有のベクターを与えることもできる。

ベクトルが使われる：

1. 割り込みを MSI として生成する IOMMU が、生成する MSI を決定するために MSI コンフィギュレーションテーブル (msi_cfg_tbl) にインデックスを作成する。IOMMU は、capabilities.IGS==MSI または capabilities.IGS==BOTH の場合、割り込みを MSI として生成できる。capabilities.IGS==BOTH の場合、fctl.WSI を 0 に設定することで、IOMMU は MSI として割り込みを生成するように設定

できる。

2. WSI を生成する IOMMU は、割り込みをシグナルするワイヤを決定する。IOMMU は、capabilities.IGS==WSI の場合、または以下の場合に、ワイヤシグナリング割り込みを生成することができる。

capabilities.IGS==BOTH.capabilities.IGS==BOTHの場合、fctl.WSIを1に設定することで、IOMMUがワイヤシグナル割り込みを生成するように設定できる。

もし実装が1つのベクターしかサポートしていない場合、このレジスタの全ビットは0（WARL）にハードワイヤされるかもしれない。同様に、2つのベクターしかサポートしていない場合は、それぞれの原因のビット0だけが書き込み可能です。

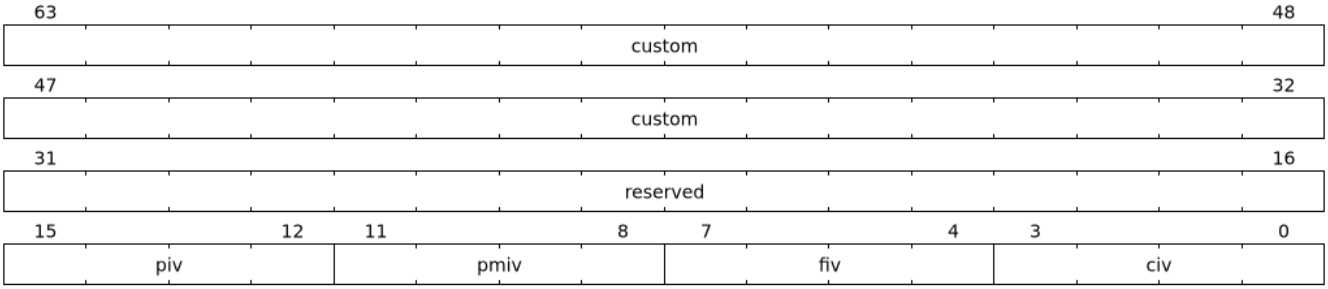


図59.割り込み原因対ベクトル・レジスタ・フィールド

ビット	フィールド	属性	説明
3:0	シヴ	ワール	コマンドキューインタラプトベクター（civ）は、コマンドキューインタラプトに割り当てられたベクター番号である。
7:4	ファイブ	ワール	fault-queue-interrupt-vector（fiv）は、fault-queue-interruptに割り当てられたベクタ番号である。
11:8	pmiv	ワール	performance-monitoring-interrupt-vector (pmiv)は、performance-monitoring-interruptに割り当てられたベクター番号である。
15:12	ピブ	ワール	page-request-queue-interrupt-vector（piv）は、page-request-queue-interruptに割り当てられたベクター番号である。
31:16	控えめ	WPRI	標準使用
63:32	*習慣*	WPRI	*カスタム仕様*

5.28. MSI 設定テーブル (msi_cfg_tbl)

IOMMU 由来の割り込み（capabilities.IGS == MSI または capabilities.IGS == BOTH）を MSI として生成することをサポートする IOMMU は、MSI テーブルエントリを決定するために icvec からのベクタによってインデックス付けされる MSI 設定テーブルを実装します。割り込みベクタxの各MSIテーブル・エントリには、3つのレジスタmsi_addr_x、msi_data_x、およびmsi_vec_ctl_xがあります。これらのレジスタは、capabilities.IGS == WSIの場合、0にハードワイヤされています。

msi_addr_x を使用した MSI 書き込みでアクセスフォールトが検出された場合、IOMMU は、 TTYP を 0 に設定し、iotval を msi_addr_x の値に設定して、「IOMMU MSI write access fault」（原因 273）フォールトを報告する。

表19.MSI設定テーブル構造

ビット63	**ビット0**	**バイト・オフセット**
エントリ 0: メッセージ・アドレス		+000h
エントリー0：ベクター・コントロール	エントリ 0: メッセージ・データ	+008h
エントリ 1: メッセージ・アドレス		+010h
ビット63	**ビット0**	**バイト・オフセット**
エントリー1：ベクター・コントロール	エントリー1：メッセージ・データ	+018h
…		+020h

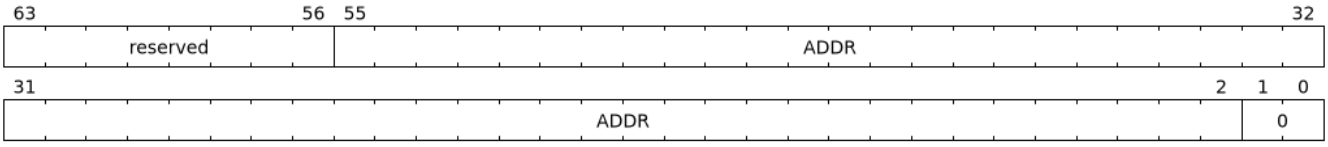


図60.メッセージ・アドレス・レジスタのフィールド

ビット	フィールド	属性	説明
1:0	0	RO	0に固定
55:2	アドレス	ワール	4バイトにアラインされたMSIアドレスを保持する。
63:56	控えめ	WPRI	標準的な使用のために予約されている。

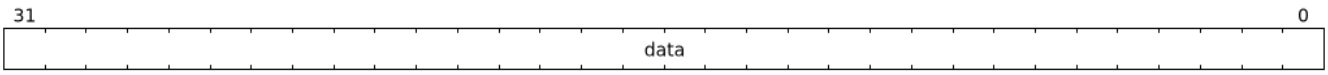


図61.メッセージ・データ・レジスタ・フィールド

ビット	フィールド	属性	説明
31:0	データ	ワール	MSIデータを保持

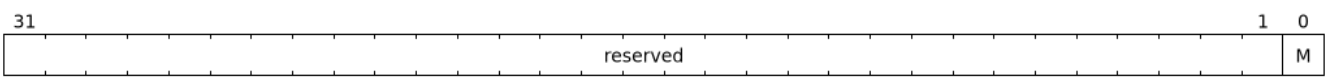


図62.ベクトル制御レジスタ・フィールド

ビット	フィールド	属性	説明
0	M	RW	マスク・ビット M が 1 の場合、対応する割り込みベクタはマスクされ、IOMMU は関連するメッセー ジの送信を禁止されます。対応するマスク・ビットが 0 にクリアされると、そのベクタに対する保留中のメッ セージが後で生成されます。
31:1	控えめ	WPRI	標準的な使用のために予約されている。

第6章.ソフトウェアのガイドライン

本節では、ソフトウェア開発者が IOMMU インターフェースを使用する際の正しい順序と期待される順序について、ガイドラインを示す。これらのガイドラインに従わない場合の IOMMU の動作は実装で定義されている。

6.1. IOMMUレジスタの読み書き

IOMMUレジスタへの読み書きアクセスは、以下の規則に従わなければならない:

- アクセスのアドレスは、アクセスのサイズにアライメントされていなければならない。
- アクセスは複数のレジスタにまたがってはならない。
- 64ビット幅のレジスタは、32ビットアクセスでも64ビットアクセスでもよい。
- 32ビット幅のレジスタは、32ビットアクセスでのみアクセスしなければならない。

6.2. 初期化のガイドライン

IOMMUの初期化のガイドラインは以下の通り:

1. IOMMU の能力を発見するためにケイパビリティ・レジスタを読む。
2. capabilities.versionがサポートされていない場合は停止し、失敗を報告する。
3. 機能制御レジスタ（fctl）を読む。
4. ビッグエンディアンのメモリアクセスが必要で、capabilities.ENDフィールドが0（つまりエンディアンが1つだけ）で、fctl.BEが0（つまりリトルエンディアン）の場合、停止して失敗を報告する。
5. ビッグエンディアンのメモリアクセスが必要で、capabilities.ENDフィールドが1（つまり両方のエンディアンをサポート）である場合、fctl.BEフィールドがまだ1でなければ1（つまりビッグエンディアン）に設定する。
6. IOMMU が開始する割り込みと機能に有線信号による割り込みが必要な場合は、停止して失敗を報告する。
7. IOMMUが開始する割り込みとケイパビリティのために有線信号による割り込みが必要な場合、IGSは次のようになる。
   BOTH、fctl.WSIが1でなければ1に設定。
8. 他の必要な機能（仮想アドレスモード、MSI変換など）がサポートされていない場合は、停止して失敗を報告する。

9. icvec レジスタは、各割り込み要因に対する割り込みベクタをプログラムするために使用される。各フィールドに 0xF を書き込み、書き込み可能なビット数を読み出すことで、IOMMU がサポートするベクタの数を決定します。書き込み可能なビット数が N の場合、サポートされるベクタの数は 2^N です。各原因 C に対して、ベクタ V をその原因に関連付けます。Vは0から$(2^N - 1)$の間の数である。

10. IOMMUがワイヤード割り込みを使用するように構成されている場合、各ベクタVはプラットフォーム・レベル割り込みコントローラ（APLICなど）に接続された割り込みワイヤに対応します。デバイス・ツリーなどのコンフィギュレーション機構によって提供されるコンフィギュレーション情報を使用して、このようなワイヤごとにプログラムすべき割り込みコントローラ・コンフィギュレーション・レジスタを決定し、割り込みコントローラをプログラムします。

11. IOMMU が MSI を使用するように設定されている場合、各ベクタ V は msi_cfg_tbl のインデックスとなる。msi_addr_V レジスタに値 A を、msi_data_V レジスタに値 D を設定し、msi_vec_ctl_V レジスタの割り込みマスク M を適切に設定する。

12. コマンド・キューをプログラムするには、まずコマンド・キューに必要なエントリ数Nを決定する。コマンド・キューのエントリ数は2のべき乗でなければならない。N×16バイトの大きさのメモリ・バッファを割り当てる。このバッファは、4-KiBかN×16バイトのいずれか大きい方に自然にアライメントされる。$k=_{log}2(N)$、B を 割り当てられたメモリ・バッファの物理ページ番号(PPN)とする。コマンド・キュー・レジスタを以下のようにプログラムする：
    - temp_cqb_var.PPN = B
    - temp_cqb_var.LOG2SZ-1 = (k - 1)
    - cqb = temp_cqb_var
    - cqt = 0
    - cqcsr.cqen = 1
    - cqcsr.cqonを「1」になるまでポーリングする。

13. フォルトキューをプログラムするには、まず、フォルトキューに必要なエントリ数 N を決定する。フォルトキューのエントリ数は常に2の累乗である。N×32バイトの大きさのメモリ・バッファを割り当てる。このバッファは、4-KiBまたはN×32バイトのいずれか大きい方に自然にアラインされる。$k=log_2(N)$、Bを割り当てられたメモリバッファのPPNとする。フォールト・キュー・レジスタを以下のようにプログラムする：
    - temp_fqb_var.PPN = B
    - temp_fqb_var.LOG2SZ-1 = (k - 1)
    - fqb = temp_fqb_var
    - fqh = 0
    - fqcsr.fqen = 1
    - fqcsr.fqonを「1」になるまでポーリングする。

14. ページ要求キューをプログラムするには、まずページ要求キューに必要な エントリーの数Nを決定する。ページ・リクエスト・キューのエントリ数は常に2のべき乗である。N×16バイトの大きさのバッファを割り当てる。このバッファは、4-KiBまたはN×16バイトのいずれか大きい方に自然にアラインされる。$k=_{log}2(N)$、Bを割り当てられたメモリバッファのPPNとする。ページ要求キュー・レジスタを以下のようにプログラムする：
    - temp_pqb_var.PPN = B
    - temp_pqb_var.LOG2SZ-1 = (k - 1)
    - pqb = temp_pqb_var
    - pqh = 0

- pqcsr.pqen = 1
- pqcsr.pqonを「1」になるまでポーリングする。

15. DDT ポインタをプログラムするには、まず、サポートされる device_id 幅 Dw とデバイスコンテキストデータ構造のフォーマットを決定する。capabilities.MSIが0の場合、IOMMUはベースフォーマットのデバイスコンテキストを使用し、それ以外の場合は拡張フォーマットのデバイスコンテキストを使用する。のページ（4 KiB）を割り当てる。

DDTのルート・テーブルとして使用するメモリを確保する。割り当てられたメモリをすべて 0 に初期化する。割り当てられたメモリの PPN を B とする。DwとIOMMUデバイス・コンテキスト・フォーマットに基づいて、DDTのモードMを以下のように決定する：

- ddtp.iommu_modeがサポートする値を、正当な値を書き込み、その値が保持されたかどうかを読み取ることで判断する。サポートされているモードが必要なDwをサポートしていない場合は、停止して失敗を報告する。
- 拡張フォーマットのデバイス・コンテキストが使用される場合、次のようになる。
  a. Dwが6ビット以下で1LVLがサポートされている場合、M = 1LVL
  b. Dwが15ビット以下で2LVLがサポートされている場合、M = 2LVL
  c. Dwが24ビット以下で3LVLがサポートされている場合、M = 3LVL
- もしベースフォーマットのデバイスコンテキストが使われるなら、次のようになる。
  a. Dwが7ビット以下で1LVLがサポートされている場合、M = 1LVL
  b. Dwが16ビット以下で、2LVLがサポートされている場合、M = 2LVL
  c. Dwが24ビット以下で3LVLがサポートされている場合、M = 3LVL

ddtpレジスタを以下のようにプログラムする：

- temp_ddtp_var.iommu_mode = M
- temp_ddtp_var.PPN = B
- ddtp = temp_ddtp_var

IOMMUは初期化され、IOMMUの範囲内にあるデバイスのデバイスコンテキストを設定することができる。

6.3. 無効のガイドライン

このセクションでは、IOMMU のメモリ内データ構造を変更する際に、CQ を介して IOMMU に送信する無効化コマンドに関するガイドラインを示します。ソフトウェアは、更新がグローバルに可視化された後に無効化を実行しなければならない。FENCE 命令およびアトミック命令のアクイジション/リリース・ビットによるストアの順序付けは、IOMMU が観察するように、これらのストアに関連するデータ構造の更新も順序付けます。

IOFENCE.Cコマンドは、CQからフェッチされた以前のすべてのコマンドが完了し、コミットされたことを確認するためにソフトウェアが使用することができる。IOFENCE.C コマンドの PR および/または PW ビットを 1 に設定することで、IOMMU によって既に処理された以前のすべての読み出し/書き込み要求を、

IOFENCE.C コマンドの一部としてグローバル・オーダリング・ポイントにコミットするよう要求することができる。

6.3.1. デバイス・ディレクトリ・テーブル・エントリの変更

ソフトウェアがリーフ・レベルのDDTエントリ（つまりdevice_id = Dのデバイスのデバイス・コンテキスト（DC））を変更した場合、以下の無効化を実行しなければならない：

- DV=1、DID=DのIODIR.INVAL_DDT
- DC.tc.PDTV==1なら、IODIR.INVAL_PDTでDV=1、PV=0、DID=D

- もしDC.iohgatp.MODE != Bareなら
  - GV=1、AV=PSCV=0、GSCID=DC.iohgatp.GSCIDのIOTINVAL.VMA
  - GV=1、AV=0、GSCID=DC.iohgatp.GSCIDのIOTINVAL.GVMA
- その他
  - If DC.tc.PDTV==1 || DC.tc.PDTV == 0 && DC.fsc.MODE == Bare
    - GV=AV=PSCV=0のIOTINVAL.VMA
  - その他
    - GV=AV=0、PSCV=1、PSCID=DC.ta.PSCIDのIOTINVAL.VMA。

ソフトウェアがリーフレベルでないDDTエントリーを変更した場合、以下の無効化を実行しなければならない：

- DV=0のIODIR.INVAL_DDT

DDTエントリの変更から、キャッシュされたエントリを無効にする無効化コマンドがIOMMUによって処理されるまでの間、IOMMUはエントリの古い値または新しい値を使用することができる。

6.3.2. プロセス・ディレクトリ・テーブル・エントリの変更

ソフトウェアがリーフレベルのPDTエントリ（すなわちプロセスコンテキスト（PC）、device_id=Dかつprocess_id=P）の場合、以下の無効化を実行しなければならない：

- IODIR.INVAL_PDT（DV=1、PV=1、DID=D、PID=Pの場合
- もしDC.iohgatp.MODE != Bareなら
  - GV=1、AV=0、PV=1、GSCID=DC.iohgatp.GSCID、PSCID=PC.PSCIDのIOTINVAL.VMA
- その他
  - GV=0、AV=0、PV=1、PSCID=PC.PSCIDのIOTINVAL.VMA

PDTエントリの変更から、キャッシュされたエントリを無効にする無効化コマンドがIOMMUによって処理されるまでの間、IOMMUはエントリの古い値または新しい値を使用することができる。

6.3.3. MSIページテーブルエントリの変更

ソフトウェアが、翻訳されていないMSIアドレスAに対応する割込みファイル番号Iで識別されるMSIページテーブルエントリを変更した場合、以下の無効化を実行しなければならない：

- GV=AV=1、ADDR[63:12]=A[63:12]、GSCID=DC.iohgatp.GSCIDのIOTINVAL.GVMA

MSIページ・テーブルからすべてのキャッシュ・エントリーを無効にするには、以下の無効化を実行しなければならない：

- GV=1、AV=0、GSCID=DC.iohgatp.GSCIDのIOTINVAL.GVMA

MSI PTE が変更されてから、キャッシュされた PTE を無効にする無効化コマンドが IOMMU によって処理されるまでの間、IOMMU は古い PTE 値または新しい PTE 値を使用することができる。PW=1 の IOFENCE.C コマンドは、IOMMU によって以前に処理された MSI 書込みを含むすべての書込みがグローバル順序にコミットされることを保証するために使用できる。

システム内のすべてのRISC-Vハー トとIOMMUが観測できるようにする。

6.3.4. 第2ステージのページテーブルエントリーを変更する

ソフトウェアがVMの第2ステージのページテーブルのリーフエントリを変更し、その変更がゲストPPN Gの変換に影響する場合、以下の無効化を実行しなければならない：

- GV=AV=1、GSCID=DC.iohgatp.GSCID、ADDR[63:12]=GのIOTINVAL.GVMA

ソフトウェアがVMの非リーフ第2ステージ・ページテーブル・エントリーを変更した場合、以下の無効化を実行しなければならない：

- GV=1、AV=0、GSCID=DC.iohgatp.GSCIDのIOTINVAL.GVMA

DCにはゲストPPNを保持するフィールドがある。実装は、DCをキャッシュする一部として、そのようなフィールドをスーパーバイザPPNに変換することができる。第2段階のページテーブル更新がDCに保持されているゲストPPNの変換に影響する場合、ソフトウェアはDV=1でDIDを対応するdevice_idに設定したIODIR.INVAL_DDTを使用して、そのようなキャッシュされたDCをすべて無効にしなければならない。あるいは、DV=0のIODIR.INVAL_DDTを使用して、すべてのキャッシュされたDCを無効にしてもよい。

セカンドステージのPTEが変更されてから、キャッシュされたPTEを無効にする無効化コマンドがIOMMUによって処理されるまでの間、IOMMUは古いPTE値または新しいPTE値を使用することができる。

6.3.5. 第1ステージのページテーブルエントリーの変更

DCは、（DC.tc.PDTV=0のとき）第一段階のページテーブル、または（DC.tc.PDTV=1のとき）process-directory-tableからprocess_idを使用して選択された第一段階のページテーブルのディレクトリで構成することができる。

第 1 段のページテーブルに変更が加えられ、第 2 段がベアされる場合、ソフトウェアは、表 9 に規定されるように、GV=0、AV および PSCV オペランドが変更に適した IOTINVAL.VMA を使用して無効化を実行しなければならない。

第 1 段のページテーブルに変更が加えられ、第 2 段が Bare でない場合、ソフトウエアは、表 9 に規定されるように、 GV=1、GSCID=DC.iohgatp.GSCID、および変更に適した AV および PSCV オペランドを持つ IOTINVAL.VMA を使用して無効化を実行しなければならない。

ファーストステージの PTE が変更されてから、キャッシュされた PTE を無効にする無効化コマンドが IOMMU によって処理されるまでの間、IOMMU は古い PTE 値または新しい PTE 値を使用することができる。

6.3.6. アクセス(A)/ダーティ(D)ビットの更新とページプロモーション

IOMMUがハードウェア管理されたAビットとDビットの更新をサポートする場合、ソフトウェアがファーストステージおよび/またはセカンドステージのPTEのAビットおよび/またはDビットをクリアする場合、ソフトウェアはIOMMUによってキャッシュされる可能性のある対応するPTEエントリを無効にしなければならない。そのような無効化が実行されない場合、IOMMUは、そのようなエントリを使用する後続のトランザクショ ンを処理するときに、これらのビットを設定しないかもしれない。

ソフトウェアが、最初に元の非リーフPTEの有効ビットをクリアせず、かつ、PTE内のキャッシュされたトランスレーションを無効にすることなく、ファーストステージPTおよび/またはセカンドステージPTのページをスーパーページにアップグレードする場合。

IOMMUは、1つのアドレスに一致する複数のエントリをキャッシュすることが可能である。IOMMUは古いノンリーフPTEまたは新しいノンリーフPTEのどちらかを使用することができるが、それ以外の動作は明確に定義されている。

ページ・サイズを昇格または降格させる場合、ソフトウェアは、元の PTE と新しい PTE が同一のパーミッション属性とメモリ・タイプ属性を持ち、元の PTE または新しい PTE を使用して変換された結果決定される物理アドレスが、任意の入力に対して同一であることを確認する必要があります。元のPTE内のVビットをクリアし、適切なIOTINVALコマンドを実行することなく、IOMMUがサポートする唯一のPTE更新は、ページサイズの昇格または降格を行うことである。他の属性がこの方法で変更された場合のIOMMUの動作は、実装で定義される。

6.3.7. デバイスアドレス変換キャッシュの無効化

ファーストステージおよび/またはセカンドステージページテーブルが変更されると、変更されたページテーブルからの翻訳をキャッシュしている可能性のあるデバイスのDevATCに対して無効化が必要になる場合があります。このようなページ・テーブルの無効化には、ATS.INVALコマンドを使用してATS無効化を生成する必要がある。ソフトウェアは PCIe ATS 仕様 [1]に定義された規則に従って PAYLOAD を指定しなければならない。

ソフトウェアが平均DevATCサービス・レートを超えるレートでATS無効化リクエストを生成した場合、デバイスによってフロー制御メカニズムがトリガーされ、レートがスロットルされる可能性がある。この副作用として、輻輳が他のチャネルやリンクに広がり、性能劣化につながる可能性があります。ATSが可能なデバイスは、ATSケイパビリティ構造のQueue Depthフィールドを通じて、バックプレッシャーを引き起こす前にバッファできる無効化の最大数を公表します。デバイスが PCIe SR-IOV を使用して仮想化されている場合、このキューの深さはデバイスのすべての VF 間で共有されます。ソフトウエアは、キューに入れられる未処理の ATS 無効化の数を、デバイスがアドバタイズする制限に制限しなければならない。

RID フィールドは、ATS 無効化要求メッセージ宛先のルーティング ID を指定するために使用される。PV=1 を設定し、PIDにPASIDを指定することで、PASID固有の無効化を実行することができる。IOMMU が複数のセグメントをサポートしている場合は、DSV=1 を設定し、DSEG にセグメント番号を指定することで、RID を宛先セグメント番号で修飾する必要がある。

デバイスで ATS プロトコルが有効になっている場合、IOMMU は、DevATC に翻訳を提供するだけでなく、IOATC に翻訳をキャッシュすることができる。デバイスのコンテキストで ATS が有効になっている場合でも、ソフトウェアは IOMMU の翻訳キャッシュの無効化をスキップしてはならない。DevATCからの翻訳要求がIOATCからIOMMUによって満たされることがあるため、正しい動作を保証するために、ソフトウェアはDevATCに無効化を送信する前に、まずIOATCを無効化しなければならない。

6.3.8. 無効なエントリーのキャッシュ

本仕様では、v（有効）ビットがクリアされたファーストステージ/セカンドステージ PTE、v（有効）ビットがクリアされたノンリーフ DDT エントリ、v（有効）ビットがクリアされたデバイスコンテキスト、v（有効）ビットがクリアされたノンリーフ PDT エントリ、v（有効）ビットがクリアされたプロセスコンテキスト、または v ビットがクリアされた MSI PTE のキャッシュを許可しない。

これらのエントリーのvビットを0から1に変更する際、ソフトウェアが無効化を実行する必要はない。

6.4. PMAの再構成

プラットフォームがPMAのダイナミック・リコンフィギュレーションをサポートする場合、通常、プラットフォームを正しくコンフィギュレーションできるマシンモード・ドライバが提供されます。IOMMUがこれらのオペレーションに参加しなければならない場合、IOMMUのプラットフォーム固有のオペレーションがマシンモード・ドライバによって使用され、このようなリコンフィギュレーションが実行されます。

6.5. IOMMUからの割り込み処理のガイドライン

IOMMU は、CQ、FQ、PQ、または HPM から割り込みを発生させることができる。各割り込みソースは、固有のベクタで構成されてもよいし、1 つ以上の割り込みソース間でベクタが共有されてもよい。割り込みは MSI またはワイヤ・シグナル・インターラプトとして配信される。割り込みハンドラは以下の動作を行う：

1. ipsrレジスタを読み出して、保留中の割り込みのソースを決定する。
2. ipsr.cipビットがセットされている場合は、CQからの割り込みが保留されている。
    a. cqcsr レジスタを読む。
    b. cmd_to ビット、cmd_ill ビット、および cqmf ビットの状態を調べることで、割り込みの原因がエラーであるかどうかを判断し、エラーであればその原因を特定する。これらのビットのいずれかが設定されている場合、CQ はエラーに遭遇し、コマンド処理は一時的に無効となる。
    c. エラーが発生した場合は、エラーの原因を修正し、cqcsrの修正されたエラーに対応するビットに1を書き込んでクリアする。
        i. cqcsr のエラー表示ビットをすべてクリアすると、コマンド処理が再度有効になる。
    d. ワイヤード割り込みをサポートする IOMMU は、IOFENCE.C コマンドの完了時にコマンドキューから割り込みを生成するよう要求されることがある。この原因は、fence_w_ip ビットによって示される。ソフトウェアハンドラは、このビットに 1 を書き込んでクリアすることで、IOFENCE.C の完了時に CQ からの割り込みを再び有効にすることができる。
    e. ビットに1を書き込んでipsr.cipをクリアする。
3. ipsr.fipビットがセットされている場合、FQからの割り込みが保留されている。
    a. fqcsr レジスタを読む。
    b. fqmf ビットと fqof ビットの状態を調べることで、エラーが割り込みを発生させたかどうか、発生させ た場合はその原因を特定する。これらのビットのいずれかが設定されている場合は、FQ がエ

ラーに遭遇したことになり、フォルト/イ ベント報告が一時的に無効になります。

c. エラーが発生した場合は、エラーの原因を修正し、fqcsrの修正されたエラーに対応するビットに1を書き込んでクリアする。

   i. cqcsr のエラー表示ビットをすべてクリアすると、フォルト/イベント報告が再度有効になる。

d. ビットに1を書き込んでipsr.fipをクリアする。

e. fqtとfqhレジスタを読む。

f. fqtの値がfqhの値と等しくない場合、FQは空ではなく、処理が必要な障害/イベント報告が含まれている。

g. 処理が必要な保留中の故障/イベント・レポートを処理し、処理されたレコードの数だけfqhを進めることによって、それらをFQから削除する。

4. ipsr.pipビットがセットされていれば、PQからの割り込みが保留されている。

   a. pqcsr`レジスタを読む。

   b. pqmf ビットと pqof ビットの状態を調べることにより、エラーが割り込みを発生させたかどうか、発生させ た場合はその原因を特定する。これらのビットのいずれかが設定されている場合、PQはエラーに遭遇し、「ページ要求」報告は一時的に無効となる。

   c. エラーが発生した場合は、エラーの原因を修正し、pqcsrの修正されたエラーに対応するビットに1を書き込んでクリアする。

      i. pqcsrのエラー表示ビットをすべてクリアすると、「ページ要求」報告が再び有効になる。

   d. ビットに1を書き込んでipsr.pipをクリアする。

   e. pqtとpqhレジスタを読む。

   f. もしpqtの値がpqhの値と等しくなければ、PQは空ではなく、処理が必要な 「ページ要求」メッセージが含まれている。

   g. 処理が必要な保留中の "Page Request "メッセージを処理し、"Page Request "メッセージから削除する。
   PQは、pqhを処理されたレコードの数だけ進める。

      i. PQオーバーフロー状態が発生すると、「ページ要求」メッセージがドロップされたために、ソフトウェアが不完全なページ要求グループを観察することがある。IOMMUは、「PRG内の最後のリクエスト」フラグがメッセージ内で1に設定されていれば、そのようなグループ内でドロップされた「ページリクエスト」に対して自動的に応答（セクション2.7を参照）したかもしれない。ソフトウェアは、そのような不完全なグループを無視し、サービスを行わないべきである。

      ii. PQオーバーフロー時に「PRG内の最後の要求」が1に設定された「ページ要求」に対する自動応答は、デバイスがATS変換要求を再試行する原因となることが予想される。しかし、IOMMUが生成した応答は、デバイスが「ページ要求」を最初に送 信した原因となった状態を実際には解決していないため、デバイスが「ページ要求」メッセー ジを再度送信する可能性が高い。これらの再試行されたメッセージは、PQにスペースを作ることによってオーバーフロー状態が修正された場合、PQに格納される可能性がある。

5. ipsr.pmip ビットが設定されている場合、HPM からの割り込みが保留されている。

   a. ビットに1を書き込んでipsr.pmipをクリアする。

   b. パフォーマンス監視カウンターのオーバーフローを処理する。

6.6. ATSおよび/またはPRIの有効化と無効化のガイドライン

ATSおよび/またはPRIを有効にする：

1. デバイスによってトランザクションが生成されないように、デバイスをアイドル状態にする。
2. デバイスのデバイスコンテキストが既に有効である場合、まずデバイスコンテキストを無効としてマークし、IOMMUへのコマンドをキューに入れ、キャッシュされたすべての第1/第2ステージページテーブルエントリ、DDTエントリ、MSI PTエントリ（必要な場合）、PDTエントリ（必要な場合）を無効にする。
3. EN_ATS を 1 に設定し、必要に応じて T2GPA フィールドを 1 に設定して、デバイスコンテキストをプログラムする。

を1に設定する。EN_PRI が 1 に設定されている場合、必要に応じて PRPR を 1 に設定する。

4. デバイスコンテキストを有効なものとしてマークする。
5. デバイスが ATS を使用できるようにし、必要に応じて PRI

を有効にする。ATS および/または PRI を無効にする：

1. デバイスによってトランザクションが生成されないように、デバイスをアイドル状態にする。
2. デバイスで ATS および/または PRI を無効にする。
3. デバイスコンテキストで EN_ATS および/または EN_PRI を 0 に設定する。EN_ATS が 0 に設定されている場合は、EN_PRI と T2GPA を設定する。
   EN_PRI が 0 に設定されている場合は、PRPR を 0 に設定する。
4. IOMMUにコマンドをキューイングして、キャッシュされたすべての第1/第2ステージページテーブルエントリ、DDTエントリ、MSI PTエントリ（必要な場合）、PDTエントリ（必要な場合）を無効にする。
5. IOMMUにコマンドをキューイングし、無効化要求メッセージを生成してDevATCを無効にする。
6. デバイスのDMAオペレーションを有効にする。

第7章.ハードウェア・ガイドライン

このセクションは、プラットフォームに IOMMU を搭載するシステム/ハードウェアインテグレーターに対す るガイドラインである。

7.1. PCIeデバイスとしてのIOMMUの統合

IOMMU は、PCIe デバイスとして構築され、PCIe で定義されたベースクラス 08h、サブクラス 06h、およびプログラ ミング・インターフェイス 00h [4]を持つ専用の PCIe 機能として検出可能である。

このような IOMMU は、この仕様で定義されている IOMMU レジスタを PCIe BAR マップされたレジスタとしてマッピングしなければならない。

IOMMU は MSI または MSI-X、あるいはその両方をサポートする。MSI-X をサポートする場合、MSI-X 能力ブロックは、システム・ソフトウェアが IOMMU がサポートする各メッセージの MSI アドレスとデータのペアを設定できるように、BAR マップされたレジスタの msi_cfg_tbl を指す必要がある。MSI-X PBA は、IOMMU の同じ BAR または別の BAR に配置することができる。IOMMU は MSI-X 機能をサポートすることを推奨する。

7.2. PMAとPMPによる故障

IO ブリッジは、メモリ内データ構造にアクセスするために、IO デバイスからのメモリアクセス、 または IOMMU によって暗黙的に生成されたメモリアクセスに対して、PMA および/または PMP チェッ クを呼び出してもよい。メモリアクセスが PMA チェックに違反した場合、または PMP チェックに違反した場合、IO ブリッジはセクション 7.3 に規定されるようにメモリアクセスを中止することができる。

7.3. トランザクションの中止

中断されたトランザクションがIOMMUによって開始された暗黙のメモリアクセスである場合、IOブリッジはIOMMU自体にそのようなアクセスフォールトをシグナリングする。このようなシグナリングの詳細は、実装で定義される。

中断されたトランザクションが書き込みである場合、IO ブリッジは書き込みを破棄することができる。IO プロトコルが書き込みトランザクションに対する応答を必要とする場合（例えば、AXI）、IO プロトコルで定義された応答が IO ブリッジによって生成されることがある（例えば、BRESP 上の SLVERR - 書き込み応答チャネル）。例えば PCIe の場合、書き込みトランザクションはポストされ、書き込みトランザクションが破棄されたときに応答は返されない。

フォールトトランザクションが読み出しの場合、デバイスは完了を期待する。IO ブリッジはデバイスに完了を提供することができる。このような完了でデータが返される場合は、実装で定義される。この状態を示すために、完了の中でステータスコードがデバイスに返されることがある。例えば AXI の場合、完了ステータスは RRESP（データ読み出しチャネル）の SLVERR によって提供される。例えば PCIe の場合、完了ステータス・フィールドは "Unsupported Request"（UR）または "Completer Abort"（CA）に設定される。

7.4. 信頼性、可用性、サービス性（RAS）

IOMMU は、エラー検出を有効にする方法、検出されたエラー（その重大度、性質、場所を含む）を記録する方法、 および以下の方法を規定する RAS アーキテクチャをサポートすることができる。

エラーハンドラにエラーを報告する手段を設定する。

IOATC のようないくつかのエラーは、エラーが検出されたときにキャッシュされたインメモリデータ構造を再ロードすることで修正可能な場合がある。このようなエラーは、IOMMU の機能には影響しないと考えられる。

エラーによっては、IOMMU の重要な内部状態が破壊され、IOMMU が故障状態に陥ることがある。そのような状態の例には、ddtp、cqb などのレジスタが含まれる。そのような故障状態に入ると、IOMMU は IO ブリッジにすべての着信トランザクションを中止するよう要求することがある。

IOMMUの内部データパス内で発生した破損のようないくつかのエラーは修正できないかもしれないが、そのようなエラーの影響はIOMMUによって処理されているトランザクションに含まれるかもしれない。

トランザクション処理の一環として、IOMMUは、DDT、PDT、第1/第2ステージ・ページ・テーブルなどのメモリ内データ構造からデータを読み出す必要がある場合がある。データの提供者（メモリコントローラまたはキャッシュ）は、要求されたデータに訂正不可能なエラーがあることを検出し、データが破損していることを通知し、エラーをIOMMUに延期することができる。破損データの処理をデータの消費者に先送りするこのような手法は、一般にデータポイズニングとしても知られている。このようなエラーの影響は、破損したデータにアクセスする原因となったトランザクショ ンにまで及ぶ可能性がある。

エラーが処理中のトランザクションに影響するが、そうでなければ IOMMU がサービスを提供し続けることができる場合、 IOMMU はトランザクションを中止し（セクション 7.3 参照）、FQ にフォールトレコードをキューイングしてフォールトを報告することができる。例えば、PCIe の場合、トランザクションを中止するには、"Completer Abort (CA) "応答が適切である。このようなフォールトトランザクションを報告するために、以下の原因コードが使用される：

- DDT データ破損（原因 = 268）
- PDTデータ破損（原因=269）
- MSI PT データ破損（原因 = 270）
- MSI MRIF データ破損（原因 = 271）
- 内部データパス・エラー（原因 = 272）
- 第1/第2段階のPTデータ破損（原因=274）

もし IO ブリッジがこのような遅延エラーを、IOMMU がメモリ内データ構造にアクセスするのを妨げる他のエラーと区別してシグナリングする能力がない場合、IOMMU はこのようなエラーを、区別されたデータ破損の原因コードを使用する代わりに、アクセスフォルトとして報告することができる。

参考文献

[1] "PCI Express® Base Specification Revision 6.0".[Online].利用可能: pcisig.com/pci-express-6.0-specification.

[2] "RISC-V先進割り込みアーキテクチャ".[Online].利用可能: github.com/riscv/riscv-aia.

[3] "RISC-V命令セットマニュアル第2巻：特権アーキテクチャ".[Online].利用可能: github.com/riscv/riscv-isa-manual.

[4] "PCIコードおよびID割り当て仕様書改訂1.1".[Online].Available: pcisig.com/sites/default/files/files/PCI_Code-ID_r_1_11 v24_Jan_2019.pdf.