TEXAS INSTRUMENTS
www.tij.co.jp

参 考 資 料

TMS320C64x+ DSP
メガモジュール

リファレンス・ガイド

2008

Digital Signal Processing Solutions

JAJU058
SPRU871H 翻訳版

最新の英語版資料
http://www-s.ti.com/sc/techlit/spru871.pdf

TMS320C64x+ DSP
メガモジュール
リファレンス・ガイド

TEXAS INSTRUMENTS

ご注意

日本テキサス・インスツルメンツ株式会社（以下TIJといいます）及びTexas Instruments Incorporated（TIJの親会社、以下TIJないしTexas Instruments Incorporatedを総称してTIといいます）は、その製品及びサービスを任意に修正し、改善、改良、その他の変更をし、もしくは製品の製造中止またはサービスの提供を中止する権利を留保します。従いまして、お客様は、発注される前に、関連する最新の情報を取得して頂き、その情報が現在有効かつ完全なものであるかどうかご確認下さい。全ての製品は、お客様とTIJとの間に取引契約が締結されている場合は、当該契約条件に基づき、また当該取引契約が締結されていない場合は、ご注文の受諾の際に提示されるTIJの標準販売契約約款に従って販売されます。

TIは、そのハードウェア製品が、TIの標準保証条件に従い販売時の仕様に対応した性能を有していること、またはお客様とTIJとの間で合意された保証条件に従い合意された仕様に対応した性能を有していることを保証します。検査およびその他の品質管理技法は、TIが当該保証を支援するのに必要とみなす範囲で行なわれております。各デバイスの全てのパラメーターに関する固有の検査は、政府がそれ等の実行を義務づけている場合を除き、必ずしも行なわれておりません。

TIは、製品のアプリケーションに関する支援もしくはお客様の製品の設計について責任を負うことはありません。TI製部品を使用しているお客様の製品及びそのアプリケーションについての責任はお客様にあります。TI製部品を使用したお客様の製品及びアプリケーションについて想定されうる危険を最小のものとするため、適切な設計上および操作上の安全対策は、必ずお客様にてお取り下さい。

TIは、TIの製品もしくはサービスが使用されている組み合せ、機械装置、もしくは方法に関連しているTIの特許権、著作権、回路配置利用権、その他のTIの知的財産権に基づいて何らかのライセンスを許諾するということは明示的にも黙示的にも保証も表明もしておりません。TIが第三者の製品もしくはサービスについて情報を提供することは、TIが当該製品もしくはサービスを使用することについてライセンスを与えるとか、保証もしくは是認するということを意味しません。そのような情報を使用するには第三者の特許その他の知的財産権に基づき当該第三者からライセンスを得なければならない場合もあり、またTIの特許その他の知的財産権に基づきTI からライセンスを得て頂かなければならない場合もあります。

TIのデータ・ブックもしくはデータ・シートの中にある情報を複製することは、その情報に一切の変更を加えること無く、かつその情報と結び付られた全ての保証、条件、制限及び通知と共に複製がなされる限りにおいて許されるものとします。当該情報に変更を加えて複製することは不公正で誤認を生じさせる行為です。TIは、そのような変更された情報や複製については何の義務も責任も負いません。

TIの製品もしくはサービスについてTIにより示された数値、特性、条件その他のパラメーターと異なる、あるいは、それを超えてなされた説明で当該TI製品もしくはサービスを再販売することは、当該TI製品もしくはサービスに対する全ての明示的保証、及び何らかの黙示的保証を無効にし、かつ不公正で誤認を生じさせる行為です。TIは、そのような説明については何の義務も責任もありません。

TIは、TIの製品が、安全でないことが致命的となる用途ないしアプリケーション（例えば、生命維持装置のように、TI製品に不良があった場合に、その不良により相当な確率で死傷等の重篤な事故が発生するようなもの）に使用されることを認めておりません。但し、お客様とTIの双方の権限有る役員が書面でそのような使用について明確に合意した場合は除きます。たとえTIがアプリケーションに関連した情報やサポートを提供したとしても、お客様は、そのようなアプリケーションの安全面及び規制面から見た諸問題を解決するために必要とされる専門的知識及び技術を持ち、かつ、お客様の製品について、またTI製品をそのような安全でないことが致命的となる用途に使用することについて、お客様が全ての法的責任、規制を遵守する責任、及び安全に関する要求事項を満足させる責任を負っていることを認め、かつそのことに同意します。さらに、もし万一、TIの製品がそのような安全でないことが致命的となる用途に使用されたことによって損害が発生し、TIないしその代表者がその損害を賠償した場合は、お客様がTIないしその代表者にその全額の補償をするものとします。

TI製品は、軍事的用途もしくは宇宙航空アプリケーションないし軍事的環境、航空宇宙環境にて使用されるようには設計もされていませんし、使用されることを意図されておりません。但し、当該TI製品が、軍需対応グレード品、若しくは「強化プラスティック」製品としてTIが特別に指定した製品である場合は除きます。TIが軍需対応グレード品として指定した製品のみが軍需品の仕様書に合致いたします。お客様は、TIが軍需対応グレード品として指定していない製品を、軍事的用途もしくは軍事的環境下で使用することは、もっぱらお客様の危険負担においてなされるということ、及び、お客様がもっぱら責任をもって、そのような使用に関して必要とされる全ての法的要求事項及び規制上の要求事項を満足させなければならないことを認め、かつ同意します。

TI製品は、自動車用アプリケーションないし自動車の環境において使用されるようには設計されていませんし、また使用されることを意図されておりません。但し、TIがISO/TS 16949の要求事項を満たしていると特別に指定したTI製品は除きます。お客様は、お客様が当該TI指定品以外のTI製品を自動車用アプリケーションに使用しても、TIは当該要求事項を満たしていなかったことについて、いかなる責任も負わないことを認め、かつ同意します。

弊社半導体製品の取り扱い・保管について

半導体製品は、取り扱い、保管・輸送環境、基板実装条件によっては、お客様での実装前後に破壊/劣化、または故障を起こすことがあります。

弊社半導体製品のお取り扱い、ご使用にあたっては下記の点を遵守して下さい。

1. 静電気
   - 素手で半導体製品単体を触らないこと。どうしても触る必要がある場合は、リストストラップ等で人体からアースをとり、導電性手袋等をして取り扱うこと。
   - 弊社出荷梱包単位（外装から取り出された内装及び個装）又は製品単品で取り扱いを行う場合は、接地された導電性のテーブル上で（導電性マットにアースをとったもの等）、アースをした作業者が行うこと。また、コンテナ等も、導電性のものを使うこと。
   - マウンタやはんだ付け設備等、半導体の実装に関わる全ての装置類は、静電気の帯電を防止する措置を施すこと。
   - 前記のリストストラップ・導電性手袋・テーブル表面及び実装装置類の接地等の静電気帯電防止措置は、常に管理されその機能が確認されていること。
2. 温・湿度環境
   - 温度：0～40℃、相対湿度：40～85%で保管・輸送及び取り扱いを行うこと。（但し、結露しないこと。）
   - 直射日光があたる状態で保管・輸送しないこと。
3. 防湿梱包
   - 防湿梱包品は、開封後は個別推奨保管環境及び期間に従い基板実装すること。
4. 機械的衝撃
   - 梱包品（外装、内装、個装）及び製品単品を落下させたり、衝撃を与えないこと。
5. 熱衝撃
   - はんだ付け時は、最低限260℃以上の高温状態に、10秒以上さらさないこと。（個別推奨条件がある時はそれに従うこと。）
6. 汚染
   - はんだ付け性を損なう、又はアルミ配線腐食の原因となるような汚染物質（硫黄、塩素等ハロゲン）のある環境で保管・輸送しないこと。
   - はんだ付け後は十分にフラックスの洗浄を行うこと。（不純物含有率が一定以下に保証された無洗浄タイプのフラックスは除く。）

以上

目次

図一覧

表一覧

最初にお読みください

本書について

本書では、C64x+ メガモジュールのペリフェラルについて説明します。

表記規則

本書では、次の表記規則を使用します。

- 16 進数は末尾に h を付けて表されています。たとえば、16 進数の 40（10 進数 64）は、40h と表されています。
- 本書ではレジスタは図で表され、表形式で説明されます。
  - レジスタの図は、複数のフィールドで構成される長方形で示されます。各フィールドには、ビット名が付けられています。フィールドの始まりと終わりを示すビットがその上に、リード / ライト属性がその下に書かれています。凡例は、その属性を表すために使用される表記を示しています。
  - レジスタの図に示されている予約ビットは、将来的なデバイスの拡張を考慮しているビットを表しています。

Texas Instruments 社からの関連資料

C6000™ デバイスおよびそのサポート・ツールを解説した関連文献は、次のとおりです。関連資料は、www.ti.com から入手可能です。www.ti.com にアクセスして、検索ボックスに資料番号を入力してください。

C64x+ DSP、関連ペリフェラル、およびその他の技術資料は、C6000 DSP 製品フォルダ www.ti.com/c6000 から入手できます。

SPRU732 - **TMS320C64x/C64x+ DSP CPU and Instruction Set Reference Guide** では、TMS320C6000 DSP ファミリーの TMS320C64x と TMS320C64x+ のデジタル・シグナル・プロセッサ（DSP）の CPU アーキテクチャ、パイプライン、命令セット、および割り込みについて説明しています。C64x/C64x+ DSP 世代は、C6000 DSP プラットフォームの固定小数点デバイスを構成しています。C64x+ DSP は C64x DSP の機能性を高め、命令セットを拡張した機能強化版です。

SPRAA84 - **TMS320C64x to TMS320C64x+ CPU Migration Guide** では、Texas Instruments TMS320C64x デジタル・シグナル・プロセッサ（DSP）から TMS320C64x+ DSP への移行方法について説明しています。本書の目的は、2 つのコア間の相違点を的確に示すことです。2 つのデバイスの機能が等価な場合には、説明は省略されています。

商標

C6000、TMS320C64x+、TMS320C64x、C64x は、Texas Instruments の商標です。

商標

TEXAS INSTRUMENTS

第 1 章

JAJU058

概要

TMS320C64x+™ DSP は、TMS320C64x™ DSP アーキテクチャをベースにした新世代の DSP です。この DSP では、従来の機能が強化されただけでなく、C64x™ DSP アーキテクチャには実装されていなかった新たな機能を提供します。

C64x+™ メガモジュールは、ハードウェアが提供するメモリ、帯域管理、割り込み、メモリ保護、およびパワーダウン・サポートとともに CPU を指定するために使われる名前です。本章では、C64x+ メガモジュールのメイン・コンポーネントとその機能について概要を説明します。本書では、C64x+ CPU について説明しません。『TMS320C64x/C64x+ DSP CPU and Instruction Set Reference Guide』（資料番号 SPRU732）をご覧ください。

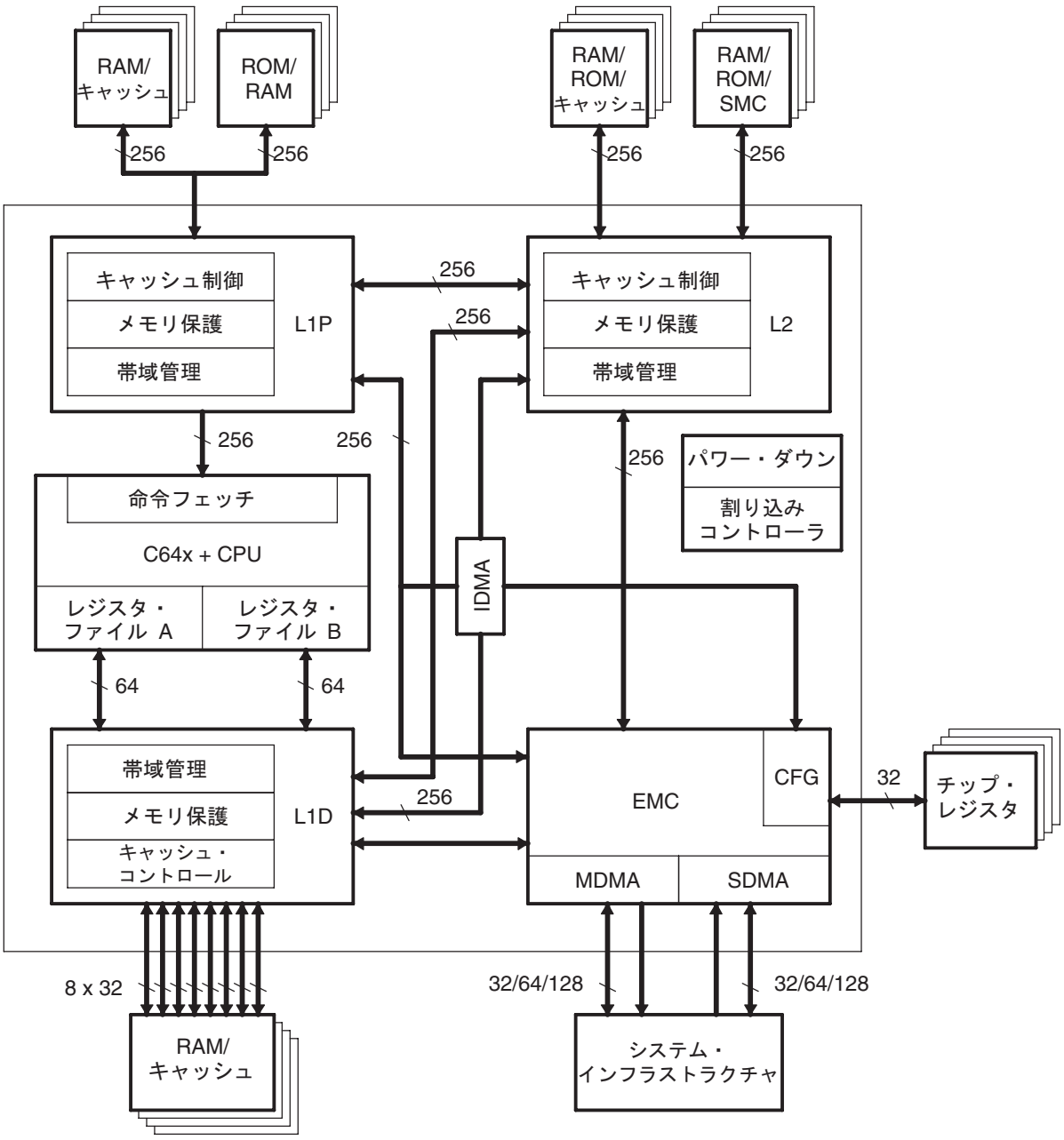
項目 **ページ**

1.1 はじめに

C64x+ メガモジュールに含まれているコンポーネントは、C64x+ CPU、レベル 1 プログラム（L1P）メモリ・コントローラ、レベル 1 データ（L1D）メモリ・コントローラ、レベル 2 （L2）メモリ・コントローラ、内部 DMA（IDMA）、帯域管理（BWM）、割り込みコントローラ（INTC）、パワーダウン・コントローラ（PDC）および拡張メモリ・コントローラ（EMC）です。

メガモジュールのブロック図を図 1-1 に示します。

図 1-1. TMS320C64x+ メガモジュールのブロック図



1.2 C64x+ メガモジュールの概要

これ以降では、C64x+ メガモジュールのメイン・コンポーネントとその機能について概要を説明します。

1.2.1 C64x+ CPU

C64x+ CPU は、C64x CPU の機能拡張版です。C64x+ CPU には次のような新機能があります。

- 実行性能を高める新命令
- コードのコンパクト性向上
- デバッグ機能を高めるソフトウェアおよびハードウェア例外

C64x+ デバイスは、C64x とオブジェクト・コード・レベルで互換性があります。

本書では、C64x+ CPU についてはこれ以上説明しません。C64x+ CPU の詳細については、『TMS320C64x/C64x+ DSP CPU and Instruction Set Reference Guide』（資料番号 SPRU732）を参照してください。

1.2.2 レベル 1 プログラム（L1P）メモリ・コントローラ

L1P メモリ・コントローラにより、CPU フェッチ・パイプラインは L1P メモリに接続されています。L1P メモリの一部を 1 ウェイ・セット・アソシアティブ・キャッシュとして構成できます。サポートされているキャッシュ・サイズは、4KB、8KB、16KB、32KB です。

L1P は帯域管理、メモリ保護、およびパワーダウン・サポートを行います。L1P メモリは、リセット後に、すべての SRAM または最大キャッシュのいずれかに常に設定されます。この動作は C64x+ デバイスごとに異なります。

L1P キャッシュ / メモリの詳細については、第 2 章を参照してください。

1.2.3 レベル 1 データ（L1D）メモリ・コントローラ

L1D メモリ・コントローラにより、CPU データ・パスは L1D メモリに接続されています。L1D メモリの一部を 2 ウェイ・セット・アソシアティブ・キャッシュとして構成できます。サポートされているキャッシュ・サイズは、4KB、8KB、16KB、32KB です。

L1D は帯域管理、メモリ保護、およびパワーダウン・サポートを行います。L1D メモリは、リセット後に、すべての SRAM または最大キャッシュのいずれかに常に設定されます。この動作はデバイスごとに異なります。

L1D キャッシュ / メモリの詳細については、第 3 章を参照してください。

1.2.4 レベル 2（L2）メモリ・コントローラ

L2 メモリ・コントローラにより、レベル 1 メモリは上位レベルのメモリに接続されています。L2 メモリの一部を 4 ウェイ・セット・アソシアティブ・キャッシュとして構成できます。サポートされているキャッシュ・サイズは、32KB、64KB、128KB、256KB です。

L2 は帯域管理、メモリ保護、およびパワーダウン・サポートを行います。L2 メモリは、リセット後に、常にすべて SRAM として設定されます。キャッシュ・モードを開始する場合、デバイス実行時に設定する必要があります。

内部メモリの一部をキャッシュとして構成する場合、L2 コントローラはメモリ内容に対して加えた変更をライトバックする手段、またはキャッシュ内容を完全にインバリデートする手段を提供します。これは、ブロック単位またはブロック全体で行われます。これらの動作によって指定したコヒーレンス操作が可能です。つまり、キャッシュされた情報を元のメモリ・ロケーションの内容とコヒーレントにすることを目的としています。またキャッシュ・アーキテクチャの動作方法に基づいて、キャッシュのライトバックおよびインバリデートも自動的に行われます。本書では、このような動作を一般にコヒーレンス動作と呼んでいます。コヒーレンス動作の詳細については、第 2 章、第 3 章、第 4 章をそれぞれ参照してください。

L2 キャッシュ / メモリの詳細については、第 4 章を参照してください。

1.2.5 内部 DMA（IDMA）

内部 DMA（IDMA）は、メガモジュールにローカルな DMA です。つまり、メガモジュール（L1P、L1D、L2、CFG）内でのみデータ転送サービスを行います。

2 つの IDMA チャネル（0 および 1）があります。

- チャネル 0 を使用すると、データはペリフェラル・コンフィギュレーション空間（CFG）とすべてのローカル・メモリ（L1P、L1D、L2）間で転送可能です。
- チャネル 1 を使用すると、データはローカル・メモリ（L1P、L1D、L2）間で転送可能です。

IDMA データ転送は、CPU が動作しているバックグラウンドで行われます。つまり、チャネル転送がプログラムされると、他の CPU 動作と同時に転送が行われます。しかも、これ以上 CPU が介在することはありません。

IDMA の詳細については、第 5 章を参照してください。

1.2.6 帯域管理（BWM）

C64x+ メガモジュールには、一連のリソース（L1P、L1D、L2、コンフィギュレーション・バス）、およびそのリソースを使用する必要がある一連のリクエスタ（CPU、SDMA、IDMA、コヒーレンス動作）が組み込まれています。リクエスタによるリソースへのアクセスを長時間ブロックさせないようにするために、C64x+ メガモジュールは、帯域管理方式を実装しています。それによって、すべてのリクエスタに対して、特定の帯域幅を保証できます。

BWM の詳細については、第 6 章を参照してください。

1.2.7 割り込みコントローラ（INTC）

C64x+ CPU は、2 つのタイプの非同期信号サービスを行います。

- 割り込み
- 例外

割り込みは、外部または内部のハードウェア信号が存在するため、通常のプログラム・フローをリダイレクトする手段を提供します。例外も、プログラム・フローをリダイレクトするという点でよく似ていますが、通常、システム内のエラー状態に関連付けられています。

C64x+ CPU は、12 個のマスカブル / 構成可能な割り込み、1 個のマスカブル例外、および 1 個のアンマスカブル割り込み / 例外を受け取ることができます。また、この CPU がさまざまな内部例外状態に応答するのは、そのような状態が CPU 内部に完全に組み込まれているためです（ただし、『TMS320C64x/C64x+ DSP CPU and Instruction Set Reference Guide』（資料番号 SPRU732）にはその記述はありません）。

メガモジュールには、割り込みコントローラが内蔵されていて、最大で 124 個のシステム・イベントを CPU 割り込み / 例外入力に送ることができます。これらの 124 個のシステム・イベントは、マスカブル割り込みに直接接続することも、割り込みまたは例外として一緒にグループにまとめることもできます。このようなさまざまなルーティング方法が選択できるため、イベント処理に大幅な柔軟性がもたらされます。

割り込み信号が CPU に送られ、かつ、この割り込みを保留していることを示すフラグがセットされている場合、エラー・イベント信号が送られます。割り込みコントローラは、ルーティング・イベントに加え、CPU が割り込みをミスした状態を検出します。このエラー・イベントを使用すると、CPU がリアルタイム・イベントを処理できないことを、CPU に対して通知することができます。INTC ハードウェアは、ミスした割り込み数をレジスタに保存します。その結果により、是正措置を取ることができます。

INTC の詳細については、第 7 章を参照してください。

1.2.8 メモリ保護アーキテクチャ（MPA）

C64x+ メガモジュールは、自身のローカル・メモリ（L1P、L1D、L2）に対応したメモリ保護サポートを行います。システム・レベルでのメモリ保護は、デバイス固有のもので、すべてのデバイスで使用できるとは限りません。詳細については、各デバイスのデータシートを参照してください。

メモリ保護は、グローバルに定義されていますが、ローカルに実装されます。そのため、全体の保護方式は、C64x+ メガモジュール全体で規定されますが、各リソースは自身の保護ハードウェアを実装しています。メモリ保護の分散方法は、メモリ保護インターフェイスを 1 つだけ理解する必要があることを意味します。それでも C64x+ は将来的なペリフェラルやメモリをサポートできるほどの高い柔軟性を保持しています。

メモリ保護方式を実装するために、メモリ・マップは「ページ」に分割されており、各ページには対応するアクセス権限のセットがあります。無効なアクセスを示す信号が例外とともに送られ、メモリ・フォールト・レジスタでシステムへレポートされます。また、MPA は特権モード（スーパーバイザとユーザ）およびメモリ・ロックをサポートします。

メモリ保護方式の分散実装のため、全体の定義は第 8 章を参照してください。MPA 実装上の固有の詳細については、それぞれのリソースについて説明している章を参照してください。

1.2.9 パワーダウン・コントローラ（PDC）

パワーダウン・コントローラを使用すると、すべての C64x+ メガモジュール・コンポーネントに対応したソフトウェア駆動によるパワーダウン管理を行うことができます。CPU は、実行スレッドに基づき、またはホストやグローバルに動作するコントローラによる外部からの入力に対応して、パワーダウン・コントローラを通じて C64x+ メガモジュールの全部または一部をパワーダウンすることができます。

PDC の詳細については、第 9 章を参照してください。

1.2.10 拡張メモリ・コントローラ（EMC）

拡張メモリ・コントローラ（EMC）は、メガモジュールから他の残りのデバイスへの橋渡しを行います。これには、3 つのポートがあります。

- コンフィギュレーション・レジスタ（CFG）：このポートは、C64x+ デバイス上のさまざまなペリフェラルおよびリソースを制御するメモリマップド・レジスタへアクセスできます。

  注： このポートは、CPU またはメガモジュール内にあるこれらのコントロール・レジスタへアクセスできません。

- マスタ DMA（MDMA）：マスタ DMA は、メガモジュール内で開始されたトランザクションを処理するメガモジュール外部のリソースへアクセスできます（つまり、メガモジュールがトランザクションを処理するマスタになる場所でのアクセス）。通常、マスタ DMA は、L2 レベルを超えるメモリへの CPU / キャッシュによるアクセスに使用されます。これらのアクセスは、キャッシュ・ライン・アロケート、ライトバック、およびシステム・メモリ間でのキャッシュ不可能なロード / ストアの形式で行われます。
- スレーブ DMA（SDMA）：スレーブ DMA は、メガモジュール内部のリソースへアクセスを DMA コントローラ、HPI などのメガモジュール外部にあるシステム・マスタに提供します。つまり、転送はメガモジュールがトランザクション処理時にスレーブになっているメガモジュール外部から開始されます。

CFG バスは、常に 32 ビット幅です。したがって、32 ビットのロード / ストア命令または IDMA を使用して、常に 32 ビット値としてアクセスしてください。MDMA および SDMA ポートの幅は 32、64、128 ビットのいずれかです。ただし、実際の幅は、デバイスごとに異なります。そのため、ご使用になる C64x+ デバイスに対応したデータ・マニュアルを参照してください。

レベル 1 プログラム・メモリ / キャッシュ

項目 **ページ**

2.1 はじめに

2.1.1 レベル 1 プログラム（L1P）メモリ / キャッシュの目的

レベル 1 プログラム（L1P）メモリ / キャッシュの目的は、コードの実行性能を最大限に高めることです。L1P キャッシュを容易に設定できるため、多くのシステムで求められる柔軟性がもたらされます。

2.1.2 機能

L1P メモリ / キャッシュは、C64x+ メガモジュールを使用するデバイスでメモリ転送時に要求される柔軟性を提供します。

- 設定可能な L1P キャッシュ・サイズ：0K、4K、8K、16K、32K
- メモリ保護
- キャッシュ・ブロックおよびグローバルなコヒーレンス動作

2.2 用語と定義

本章で使用する用語の詳細な定義については、本書の付録 A および付録 B を参照してください。付録 A では、本書全体で使用している一般的な用語について説明し、付録 B ではメモリ / キャッシュ・アーキテクチャ関連の用語を定義しています。

2.3 L1 プログラム・メモリ・アーキテクチャ

2.3.1 L1P メモリ

L1P メモリは、RAM と ROM を合わせて最大で 1 MB までサポートします。このメモリは、2 つの領域に分割されます。それぞれの領域は、512KB より大きくすることはできません。L1P メモリは、同一メガモジュール内のレベル 1 データ（L1D）キャッシュ、レベル 1 プログラム（L1P）キャッシュ、レベル 2（L1P）キャッシュのいずれにもキャッシュされません。

L1P メモリのベース・アドレスには、1MB 境界の制約があります。L1P メモリの合計サイズは、16KB の倍数にする必要があります。

2.3.1.1 L1P 領域

L1P メモリは、2 つの領域に分割されています（L1P 領域 0 および L1P 領域 1）。

領域は、L1P に 2 つの主要な影響を与えます。

1. それぞれの領域では、ウェイト・ステート数が異なる
2. それぞれの領域には、別々のメモリ保護エントリがある

この 2 つの領域は、メモリ内では連続しています。領域 0 のサイズは、0KB（ディスエーブル）または 16K ～ 512KB の範囲の 2 の累乗で表される数値になります。領域 1 は、領域 0 の最後から始まります。そのサイズは、16K の倍数で 16K ～ 512KB の範囲にあります。領域 1 のサイズは、領域 0 のサイズ以下にする必要があります（領域 0 がイネーブルの場合）。

2 つの L1P 領域では、メモリ保護エントリは 2 つのセットに分割されています。L1P は 32 ページのメモリを保護できます（2.9.2 項を参照）。先頭の 16 ページは領域 0 を、次の 16 ページは領域 1 をそれぞれ対象とします。領域 0 のサイズが 0KB の場合、メモリ保護ページは使われません。

実際のメモリ構成は、デバイスごとに異なります。詳細については、各デバイスのデータシートを参照してください。

2.3.1.2 L1P アクセス

L1P 領域には、EDMA または IDMA アクセスを使用して、ライトのみ可能です。L1P 領域には、CPU ストアを使用して、ライトすることはできません。L1P 領域をリードするには、EDMA、IDMA、CPU のアクセスのいずれかを使用します。

2.3.1.3 L1P ウェイト・ステート

2 つの領域では、ウェイト・ステート数が異なります。最大のウェイト・ステート数は、3 です。ウェイト・ステート数は、ソフトウェアでは設定できません。チップ製造時に規定されているためです。L1P SRAM は通常、0 ウェイト・ステートのメモリです。L1 レベルの ROM は、0 より大きいウェイト・ステートでメモリを転送します。詳細については、各デバイスのデータシートを参照してください。

2.4 L1P キャッシュ

L1P キャッシュは、大容量のシステム・メモリを取り扱うために、高速なクロック・レートでプログラム・コードのフェッチを容易にする上で必要です。このキャッシュは、低速のシステム・メモリ間でリード / ライト時に発生するレイテンシを意識させないようにします。

L1P の一部またはすべてをキャッシュに変換することができます。L1P がサポートしているキャッシュ・サイズは、4K、8K、16K、および 32K です。

L1P キャッシュは、L1P メモリ・マップの一番上から下方向へ進みながら、メモリを RAM からキャッシュに変換します。わかりやすく説明すると、L1P 領域 1 の最上位アドレスが先頭のキャッシュになります。L1P キャッシュは、領域 1 のみを使用します。

キャッシュ・コントローラは、リセット後に「すべての RAM」または「最大のキャッシュ」に初期化します。デバイス固有の動作については、各デバイスのデータシートを参照してください。

L1P キャッシュの動作は、いくつかのレジスタによって制御されています。表 2-1 に、これらのレジスタの概要を示します。ここではレジスタを簡単に説明しています。詳細については 2.6 節を参照してください。

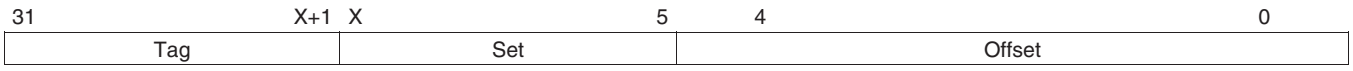
表 2-1. L1P キャッシュ・レジスタの概要

アドレス	略称	レジスタの説明	参照先
0184 0020h	L1PCFG	L1P コンフィギュレーション・レジスタ	2.6.3.1 項
0184 0024h	L1PCC	L1P キャッシュ・コントロール・レジスタ	2.6.3.2 項
0184 4020h	L1PIBAR	L1P インバリデート・ベース・アドレス・レジスタ	2.6.3.3 項
0184 4024h	L1PIWC	L1P インバリデート・ワード・カウント・レジスタ	2.6.3.4 項
0184 5028h	L1PINV	L1P インバリデート・レジスタ	2.6.3.5 項

2.4.1 L1P キャッシュ・アーキテクチャ

L1P キャッシュは、ダイレクト・マップド・キャッシュです。つまり、システムの物理メモリ・ロケーションはいずれも、キャッシュの中に対応したロケーションを 1 つもつということです。CPU がコードをフェッチしようとした場合、L1P はリクエストされたアドレスが L1P キャッシュに存在するかチェックします。これを行うために、CPU が提供する 32 ビット・アドレスを 3 つのフィールド（Tag（タグ）、Set（セット）、Offset（オフセット））に分割します（図 2-1 を参照）。

図 2-1. データ・アクセス・アドレス構成



5 ビットからなる Offset は、L1P ラインのサイズが 32 バイトであるということを示します。キャッシュ制御ロジックは、このアドレスのビット 0 ～ 4 を無視します。Set フィールドは、データが存在する L1P キャッシュ・ライン・アドレスを示します（データがキャッシュされている場合）。Set フィールドの幅は、キャッシュとして設定された L1P の量によって異なります。L1P は Set フィールドを使用して、有効なビットだけでなく、すでにキャッシュされているデータのタグをそのアドレスから調べチェックします。Tag のアドレスがキャッシュに保持されている有効なアドレスを実際に表しているかどうかがわかります。

Tag フィールドは、データ・エレメントの正しい物理ロケーションを指すアドレスの上位部分です。プログラム・フェッチ時に、Tag が一致し、かつ対応する有効ビットがセットされると、フェッチは「ヒット」になり、データは L1P キャッシュ・ロケーションから直接リードされ、CPU に返されます。それ以外の場合、フェッチは「ミス」になり、リクエストはシステムの L1P キャッシュ・ロケーションからデータをフェッチする L2 コントローラに送られます。ミスは直接的あるいは間接的に CPU をストールさせる原因になる場合もあります。

CPU は通常の状況では L1P にデータをライトできません。

L1P のキャッシュ設定は、Set および Tag フィールドのサイズを指定します。

2.4.2 リプレースメントおよびアロケーション処理

L1P キャッシュは、すべてのキャッシュ設定でダイレクト・マップド・キャッシュとして動作します。これは、システム・メモリの各ロケーションは L1P キャッシュのただ 1 つのロケーションにのみ対応するということです。L1P がダイレクト・マップドなため、そのリプレースメント処理はわかりやすいものになります。新たにキャッシュされる各ラインは、以前キャッシュされたラインを置き換えます。

L1P コントローラは、リード・アロケート・キャッシュです。これは、L1P がリード・ミス時に 32 バイトのライン全体をフェッチすることになるということです。

2.4.3 L1P モード変更動作

C64x+ L1P アーキテクチャにより、実行時に L1P のキャッシュ・サイズを選択できます。プログラムから L1PCFG レジスタの L1PMODE フィールドにリクエストされたモードをライトすることにより、L1P のキャッシュ・サイズを選択します。

表 2-2. L1PCFG レジスタの L1PMODE ビットで指定されるキャッシュ・サイズ

L1PCFG レジスタの L1PMODE 設定	L1P キャッシュの量
000b	0K
001b	4K
010b	8K
011b	16K
100b	32K
101b	「最大のキャッシュ」。32K にマップ。
110b	
111b	「最大のキャッシュ」。32K にマップ。

注 : 一般に、L1PMODE の値に大きい値を指定したら、キャッシュ・サイズも大きくなります（実装されている L1P メモリの最大サイズまで）。L1P の最大キャッシュ・サイズは、「L1P RAM サイズに収まる最大の 2 の累乗」と 32K の小さい方になります。

L1P キャッシュ・モードの実際の範囲は、L1P 領域 1 のサイズの制約を受けます。L1P 領域 1 のサイズがわずか 16K しかない場合、L1P キャッシュは、16K より大きくすることはできません。そのため、011b ～ 111b の場合、L1P 領域 1 のサイズが 16K しかないデバイスの 16K キャッシュにマップされます。

このようなデバイスでは、L1PMODE の設定が 100b ～ 111b の場合、32K キャッシュ・モードではなく、16K キャッシュ・モードが選択されます。そのため、モード 000b ～ 011b によりリクエストされたサイズ 0K ～ 16K を常に選択できます。モード 100b ～ 111b では、L1P のメモリ・サイズによって指定された最大サイズが選択されます（16K または 32K）。

このポリシーに基づいて、一定量のキャッシュしか必要としないプログラムでは、上位境界に対応する値をプログラムしてください。「できるだけ多くのキャッシュ」を必要とするプログラムでは、L1PMODE に 111b をプログラムしてください。

プログラムでキャッシュ・モードの変更を行った場合、L1P キャッシュ自体は現在保持している内容をインバリデートします。これにより、キャッシュ・タグの長さが変わっても、間違ったヒットが起こらないことが保証されます。

正しいキャッシュ動作を確実にする上でキャッシュをインバリデートすることは必要ですが、L1P RAM 部分がキャッシュになるため、データ損失を防ぐには十分とはいえません。そのため、L1P キャッシュ・モードを安全に変更するために、アプリケーションは表 2-3 に示す手順に従う必要があります。

表 2-3. L1P モードの切り換え

変更前のモード	変更後のモード	プログラムで、次のステップを実行する必要がある
L1P キャッシュをまったく使用しないか、一部使用するモード	L1P キャッシュをより多く使用するモード	1. DMA、IDMA を使用して、L1P RAM の影響を受ける範囲から必要なデータをコピーします（保存する必要がなければ、DMA は不必要）。 2. 目的のキャッシュ・モードを L1PCFG レジスタの L1PMODE フィールドにライトします。 3. L1PCFG をリードバックします。これにより、モード変更が完了するまで CPU はストールします。
L1P キャッシュを一部使用するモード	L1P キャッシュをより少なく、あるいはまったく使用しないモード	1. 目的のキャッシュ・モードを L1PCFG レジスタの L1PMODE フィールドにライトします。 2. L1PCFG をリードバックします。これにより、モード変更が完了するまで CPU はストールします。

2.4.4 L1P フリーズ・モード

L1P キャッシュは、フリーズ・モードを直接サポートします。このモードを使用すると、アプリケーションは CPU データ・アクセスによってキャッシュからプログラム・コードが追い出されないようにすることができます。この機能は、割り込みのコンテキスト時に便利です。L1P フリーズ・モードは L1P キャッシュにのみ影響を与えます。L1P RAM は、フリーズ・モードの影響を受けません。

フリーズ・モードのとき、L1P キャッシュはリード・ヒットを通常処理します。リード・ヒットは、キャッシュからデータを戻します。フリーズ・モードでは、L1P キャッシュはリード・ミス時に新たなキャッシュ・ラインをアロケートしません。またそれによって、既存のキャッシュ内容が無効とマークされることはありません。

L1PCC レジスタの OPER フィールドは、L1P がフリーズしているか、正常に動作しているかを制御します（例 2-1 を参照）。

L1PCC レジスタの OPER フィールドに 001b をライトすることにより、CPU は L1P をフリーズ・モードに設定します。L1PCC レジスタの OPER フィールドに 0b をライトすることにより、CPU は L1P を通常動作に戻します。

L1PCC レジスタの POPER フィールドは、OPER フィールドの 1 つ前の値を保持しています。OPER フィールドの値は、L1PCC レジスタへライトすると、L1PCC レジスタの POPER フィールドにコピーされます。（OPER フィールドの 1 つ前の値を保存するために）OPER フィールドの値を POPER フィールドにライトする前にコピーすることは、L1PCC レジスタのリードにかかるサイクル・コストを軽減します。もし仮に POPER フィールドが L1PCC レジスタにない場合、プログラムで 1 つ前の動作モードを記録するために OPER フィールドの値のリード、ライト、さらにリードしてキャッシュを完全にフリーズする必要があります。POPER フィールドが L1PCC レジスタにある場合、この動作により 1 回ライトしてからリードするだけになります。

L1PCC レジスタにライトした場合、次の処理が行われます。

1. OPER フィールドの内容を L1PCC レジスタの POPER フィールドにコピーします。
2. POPER フィールドの以前の内容が失われます。
3. OPER フィールドは、CPU が L1PCC レジスタのビット 0 にライトした値に従って更新されます。そのため、L1PCC レジスタにライトすると、このレジスタの OPER フィールドのみが変更されます。

プログラムでは、1 回のライトで POPER フィールドを直接変更することができません。これはそれほど問題にはなりません。POPER フィールドに保持されている値は、L1P キャッシュの動作を変更するわけではなく、OPER フィールドに最近ライトしたプログラムにだけ関係するためです。

L1PCC レジスタが確実に更新されるようにするために、ソフトウェアで L1PCC レジスタへライトしてから、L1PCC レジスタをリードする必要があります。L1PCC レジスタへライトしてから、リードすると、リクエストしたモードが有効になることが保証されます。

L1PCC レジスタの OPER フィールドを使用する目的は、使用しなければ必要となる CPU サイクルの多大なペナルティおよびリード - ライト - 再リードの順番に関わるコード・サイズの増大を回避することです。このように、アプリケーションは例 2-1 に示すような短い手順のコードで L1P を迅速にフリーズし、L1P の以前「フリーズしていた」ステートを記録することができます。

例 2-1. L1P を迅速にフリーズさせるコード・シーケンスの例

```
     MVKL     L1PCC,      A0      ;  Point to L1PCC
     MVKH     L1PCC,      A0      ;
||   MVK      1b,       B0      ;  OPER encoding for 'freeze'

     STW      B0,        *A0[0]  ;  Write 1b to L1PCC.OPER
     LDW     *A0[0],     A1      ;  Read L1PCC to get L1PCC.POPER
     NOP      4
     ;  At this point, L1P is frozen, and the CPU has the old OPER value
     ;  in bit 16 of A1.
```

L1PCC レジスタを使用すると、キャッシュをフリーズしたときと同じ方法で、キャッシュのフリーズを解除できます。例 2-2 に、これを行う方法を示します。

例 2-2. L1PCC の OPER ビットを復元するコード・シーケンスの例

```
; Assume A1 holds value read in at the end of the L1P Quick Freeze
         Example Code Sequence above.
    MVKL    L1PCC,      A0      ; Point to L1PCC
    MVKH    L1PCC,      A0      ;
||  SHRU    A1,   16,   A1      ; Shift POPER field into OPER's position

    STW     A1,         *A0[0]  ; Write to L1PCC, restoring old value of OPER
    LDW     *A0[0],     A1      ; Read back L1PCC to ensure change is complete
    NOP     4
    ; At this point, L1P is in its previous state (frozen or unfrozen)
```

また L1D キャッシュも、フリーズ・モードをサポートします。L1D のキャッシュ・フリーズ・モードの詳細については、第 3 章を参照してください。

多くの場合において、2 つのキャッシュを同時にフリーズさせることが望ましいです。したがって、L1DCC と L1PCC に連続してライトしてから L1DCC と L1PCC をリードするのは、L1D と L1P の両方がフリーズしたことを確認できる確実な方法です。例 2-3 に、L1D と L1P の両方をフリーズさせる手順を示します。

例 2-3. L1D と L1P の両方を同時にフリーズさせるコード・シーケンスの例

```
      MVKL    L1DCC,              A0      ; \
||    MVKL    L1PCC,      B0      ;  |__ Generate L1DCC pointer in A0
      MVKH    L1DCC,      A0      ;  |   and L1PCC pointer in B0
||    MVKH    L1PCC,      B0      ; /
||    MVK     1b,        A1      ; \___ OPER encoding for 'freeze'
||    MVK     1b,        B1      ; /    in both A1 and B1.
      STW     A1,          *A0     ; Write to L1DCC.OPER
||    STW     B1,          *B0     ; Write to L1PCC.OPER
      LDW     *A0,         A1      ; Get old freeze state into A1 from L1DCC
||    LDW     *B0,         B1      ; Get old freeze state into B1 from L1PCC
      NOP     4
      ; At this point, L1D and L1P are frozen.
      ; The old value of L1DCC.OPER is in bit 16 of A1.
      ; The old value of L1PCC.OPER is in bit 16 of B1.
```

2.5 プログラムによって開始されるキャッシュ・コヒーレンス動作

C64x+ L1P キャッシュ・アーキテクチャは、プログラムによって開始されるキャッシュ・コヒーレンス動作をサポートします。これらの動作は、アドレス・ブロックまたは L1P キャッシュ全体に作用します。

2.5.1 グローバル・コヒーレンス動作

グローバル・キャッシュ動作は、タスク・スイッチ、L1P のモード変更またはメモリ保護設定の変更などの重要なイベント間で L1P と「該当システム」の同期をとります。そのため、グローバル・キャッシュ動作は、他のプログラム動作に関して「同期」をとる動作と見なされます。

ソフトウェア制御の下で、L1P キャッシュをグローバルにインバリデートすることができます。グローバル・インバリデート動作を開始するために、プログラムで L1PINV レジスタの I ビットに 1 をライトする必要があります。

L1P キャッシュをグローバルにインバリデートしている間、コードが変更されないためライトバック動作は行われません。

L1PINV レジスタの I ビットは、グローバル・コヒーレンス動作が完了すると 0 にリセットされます。プログラムでこのフィールドをポーリングすると、動作の完了を検出できます。ポーリングするコードは、L1P キャッシュの外部に格納する必要があります。

表 2-4 に、L1P グローバル・コヒーレンス動作の概要について示します。

表 2-4. L1P グローバル・コヒーレンス動作

キャッシュ動作	使用レジスタ	L1P への影響
L1P インバリデート	L1PINV	L1P のすべてのラインがインバリデートされます。

また L2CFG レジスタの IP ビットを 1 にセットすると、L1P キャッシュをグローバルにインバリデートすることもできます。IP ビットは、C64x デバイスとの下位互換性を確保するためのものです。新規に開発するアプリケーションでは、IP ビットではなく、L1PINV レジスタを使用してください。

2.5.2 ブロック・コヒーレンス動作

ブロック・コヒーレンス動作は、グローバル・コヒーレンス動作と機能的にはほぼ同じです。ただし、指定されたコード・ブロックにのみ適用される点が異なります。このブロックは、対応する L1PIBAR および L1PIWC のベース・アドレスおよびワード・サイズ（32 ビット）でそれぞれ指定されます。

ブロック・コヒーレンス動作は、CPU 上で同時に動作しているタスクに与える影響を最小限に抑制しながら、できるだけ効率的に動作するように設計されています。ブロック・キャッシュ動作は、CPU が動作している「バックグラウンド」で動作しています。

L1PIWC の L1P インバリデート・ワード・カウント・フィールドは、ブロック・コヒーレンス動作が完了すると 0 にセットされます。プログラムで、このフィールドをポーリングすると、動作の完了を検出できます。ポーリングするコードは、L1P キャッシュの影響を受けるブロックの外部に格納する必要があります。

表 2-5 は、L1P ブロック・キャッシュ動作の概要について示します。

表 2-5. L1P ブロック・キャッシュ動作

キャッシュ動作	使用レジスタ	L1P への影響
L1P インバリデート	L1PIBAR L1PIWC	範囲内のすべての L1P 内のラインがインバリデートされます。

2.6 L1P キャッシュ・コントロール・レジスタ

2.6.1 メモリ・マップド・キャッシュ・コントロール・レジスタの概要

C64x+ メガモジュールのメモリ・システムには、L1P キャッシュの動作を管理するレジスタ・セットがあります。これらのレジスタを使用すると、キャッシュ・モードの変更が可能になり、キャッシュ・コヒーレンス操作を手動で開始できます。

表 2-6 に、L1P 固有のキャッシュ・コントロール動作を行うレジスタを示します。これらのレジスタのメモリ・アドレスについては、各デバイスのデータ・マニュアルを参照してください。

使用可能なキャッシュ・コントロールの操作についての詳細は、第 4 章を参照してください。

表 2-6. L1P 固有のキャッシュ・コントロール操作を行うレジスタ

アドレス	略称	レジスタの説明	参照先
0184 0020h	L1PCFG	L1P のキャッシュ・サイズを設定します。	2.6.3.1 項
0184 0024h	L1PCC	L1P の動作モードを制御します（フリーズ / 正常）。	2.6.3.2 項
0184 4020h	L1PIBAR	L1P の指定範囲はライトバックされずにインバリデートされます。	2.6.3.3 項
0184 4024h	L1PIWC		2.6.3.4 項
0184 5028h	L1PINV	L1P の内容全体はライトバックされずにインバリデートされます。	2.6.3.5 項

前述の L1P 固有のレジスタだけでなく、L1P キャッシュは L2 固有のコントロールへライトすると同様に直接影響を受けます。キャッシュ・コントロール動作および L1P キャッシュに与える影響の詳細については、第 4 章を参照してください。

2.6.2 CPU キャッシュ・コントロール・レジスタ

CPU には内部コントロール・レジスタである、コントロール・ステート・レジスタ（CSR）が 1 つあります。このレジスタにはキャッシュ・コントロール動作専用のフィールド（CSR レジスタの PCC フィールド）が用意されています。PCC フィールドは、C64x/C62x/C67x デバイスと下位互換性を確保するためのものです。新規に開発するアプリケーションでは、このフィールドを使用してはいけません。その代わりに、L1PCFG および L1PCC レジスタ（2.6.3 項を参照）を使用してください。

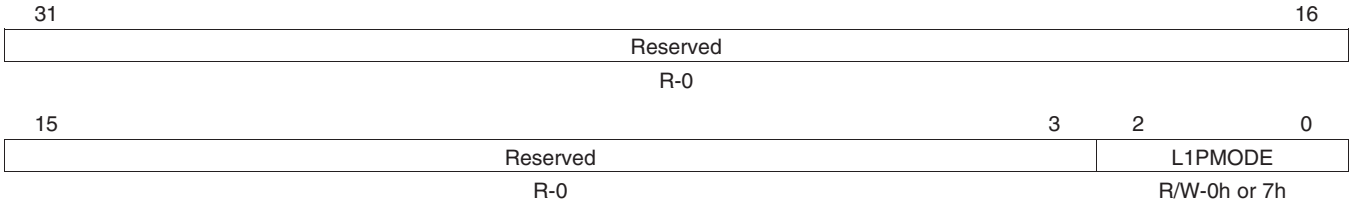
2.6.3 L1P キャッシュ・コンフィギュレーション・レジスタ

L1P コンフィギュレーション・レジスタ（L1PCFG）および L1P キャッシュ・コントロール・レジスタ（L1PCC）は、L1P の動作を制御します。

2.6.3.1 L1P コンフィギュレーション・レジスタ（L1PCFG）

L1P コンフィギュレーション・レジスタ（L1PCFG）は、L1P のキャッシュ・サイズを制御します。L1P コンフィギュレーション・レジスタ（L1PCFG）を図 2-2 に示し、表 2-7 で説明します。

図 2-2. L1P コンフィギュレーション・レジスタ（L1PCFG）



凡例：R/W = リード / ライト。R = リード専用。*-n* = リセット後の値。

表 2-7. L1P コンフィギュレーション・レジスタ（L1PCFG）フィールドの説明

ビット	フィールド	値	説明
31-3	Reserved	0	予約。
2-0	L1PMODE	0 ～ 7h	L1P のキャッシュ・サイズを指定します。L1PMODE フィールドのデフォルトは、0h または 7h です。詳細については、各デバイスのデータ・マニュアルを参照してください。
		0	L1P キャッシュはディスエーブル。
		1h	4K
		2h	8K
		3h	16K
		4h	32K
		5h	最大のキャッシュ。
		6h	最大のキャッシュ。
		7h	最大のキャッシュ。

2.6.3.2 L1P キャッシュ・コントロール・レジスタ（L1PCC）

L1P キャッシュ・コントロール・レジスタ（L1PCC）は、L1P をフリーズするか、フリーズ解除するかを制御します。

L1P キャッシュ・コントロール・レジスタ（L1PCC）を図 2-3 に示し、表 2-8 で説明します。

図 2-3. L1P キャッシュ・コントロール・レジスタ（L1PCC）

31	17	16
Reserved		POPER
R-0		R-0

15	1	0
Reserved		OPER
R-0		R/W-0

凡例：R/W = リード / ライト。R = リード専用。-*n* = リセット後の値。

表 2-8. L1P キャッシュ・コントロール・レジスタ（L1PCC）フィールドの説明

ビット	フィールド	値	説明
31-17	Reserved	0	予約。
16	POPER	0-1	OPER ビットの以前の値を保持します。
15-1	Reserved	0	予約。
0	OPER		L1P フリーズ・モードを制御します。
		0	フリーズ・モードはディスエーブル。
		1	フリーズ・モードはイネーブル。

2.6.3.3 L1P インバリデート・ベース・アドレス・レジスタ（L1PIBAR）

L1P インバリデート・ベース・レジスタ（L1PIBAR）は、ブロック・インバリデートを行うベース・アドレスを指定します。コヒーレンス動作はそれを基準とします。

L1P インバリデート・ベース・アドレス・レジスタ（L1PIBAR）を図 2-4 に示し、表 2-9 で説明します。

図 2-4. L1P インバリデート・ベース・アドレス・レジスタ（L1PIBAR）

L1P Invalidate Base Address
W-x

凡例：W = ライト専用。-*x* = 値は不定。詳細については、各デバイスのデータ・マニュアルを参照してください。

表 2-9. L1P インバリデート・ベース・アドレス・レジスタ（L1PIBAR）フィールドの説明

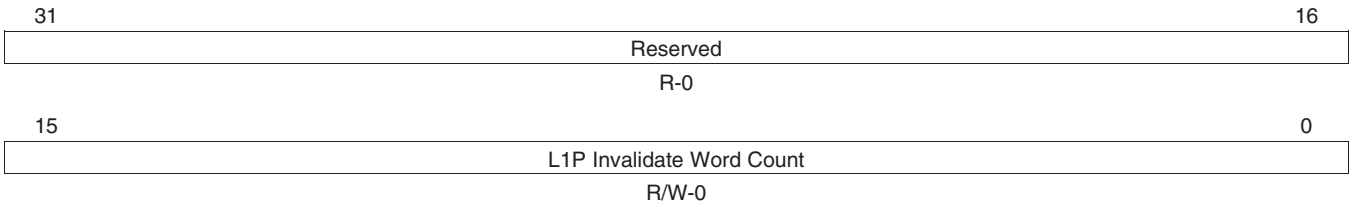
ビット	フィールド	値	説明
31-0	L1PIBAR	0 ～ FFFF FFFFh	ブロック・インバリデートを行う 32 ビット・ベース・アドレス。

2.6.3.4 L1P インバリデート・ワード・カウント・レジスタ（L1PIWC）

L1P インバリデート・ワード・カウント・レジスタ（L1PIWC）は、ブロック・インバリデートを行うサイズを指定します。コヒーレンス動作はそれを基準とします。サイズは 32 ビット・ワードで指定されています。

L1P インバリデート・ワード・カウント・レジスタ（L1PIWC）を図 2-5 に示し、表 2-10 で説明します。

図 2-5. L1P インバリデート・ワード・カウント・レジスタ（L1PIWC）



凡例：R/W = リード / ライト。R = リード専用。*-n* = リセット後の値。

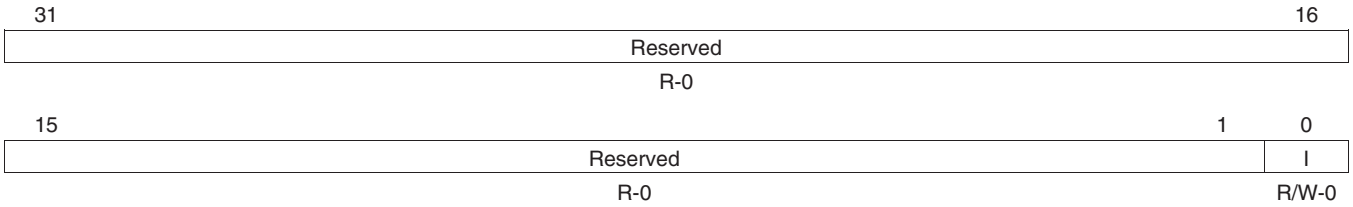
表 2-10. L1P インバリデート・ワード・カウント・レジスタ（L1PIWC）フィールドの説明

ビット	フィールド	値	説明
31-16	Reserved	0	予約。
15-0	L1PIWC	0 ～ FFFFh	ブロック・インバリデートを行うワード・カウント。

2.6.3.5 L1P インバリデート・レジスタ（L1PINV）

L1P インバリデート・レジスタ（L1PINV）は、L1P キャッシュのグローバル・インバリデートを制御します。L1P インバリデート・レジスタ（L1PINV）を図 2-6 に示し、表 2-11 で説明します。

図 2-6. L1P インバリデート・レジスタ（L1PINV）



凡例：R/W = リード / ライト。R = リード専用。*-n* = リセット後の値。

表 2-11. L1P インバリデート・レジスタ（L1PINV）フィールドの説明

ビット	フィールド	値	説明
31-1	Reserved	0	予約。
0	I		L1P キャッシュのグローバル・インバリデートを制御します。
		0	通常動作。
		1	すべての L1P キャッシュ・ラインはインバリデートされます。

2.6.4 権限およびキャッシュ・コントロール動作

キャッシュ・コントロール動作が受ける権限の影響は、次のように要約できます。

- スーパーバイザ・コードは、L1P のキャッシュ・サイズを変更できる
- ユーザ・モード・コードは、L1P のキャッシュ・サイズを変更できない
- スーパーバイザおよびユーザ・モードのコードは両方とも L1P へインバリデートを発行できる
- スーパーバイザおよびユーザ・モードは両方ともいつでも L1P のフリーズまたはフリーズ解除できる

表 2-12 に、L1P キャッシュ・コントロール・レジスタのアクセス権限の概要について示します。

表 2-12. L1P キャッシュ・コントロール・レジスタのアクセス権限

レジスタ	スーパーバイザ	ユーザ
L1PCFG	R/W	R
L1PCC	R/W	R/W
L1PINV	R/W	R/W
L1PIBAR	W	W
L1PIWC	R/W	R/W

2.7 L1P パフォーマンス

2.7.1 L1P ミス・ペナルティ

L1P にヒットするプログラム・フェッチは、CPU をストールさせることなく、1 サイクルで完了します。L2 内でヒットする L1P ミスは、そのミスを起こした近傍の実行パケットの並列度に応じて、最大で X サイクル間 CPU をストールさせることがあります。詳細については、2.7.2 項を参照してください。

L2 キャッシュ内でもミスを起こした L1P ミスは、L2 が外部メモリからそのデータを取り出して L1P にデータを転送するまで、CPU をストールさせます。その後、L1P はデータを CPU に戻します。この遅延は、システムに負荷がかかる他の状況と同様に、プログラムを保持する上で使用される外部メモリのタイプによって異なります。

C64x+ DSP では、実行パケットが 2 つのフェッチ・パケットにまたがることが可能です。このまたがることによっても、1 回のミスに対するペナルティは変わりありません。しかし、フェッチ・パケットが両方とも L1P には存在しない場合、2 回のキャッシュ・ミスが起こります。

2.7.2 L1P ミス・パイプライン化

ミス・パイプライン化は、いくつかのキャッシュ・ミスに対する処理をオーバーラップすることで、オーバーヘッドの多くを見かけ上なくすことができます。さらに、少量のキャッシュ・ミスは、フェッチ・パイプライン内で発生するディスパッチ・ストールにオーバーラップさせることができます。

L1P のミス・パイプライン化が効率的に働くのは、複数のキャッシュ・ミスが未処理のときです。C64x+ DSP のフェッチ・パイプラインは、フェッチ・パイプラインに空きがある限り、新たなフェッチ・パケットのフェッチを毎サイクル試みることで、ミス・パイプライン化を実現します。この動作内容を理解するには、フェッチ・パイプライン自体の本質を理解する必要があります。

フェッチおよびデコード・パイプラインは、全部で 6 つのステージに分割されていますが、最初の実行ステージである E1 は含まれていません。それぞれのステージは、次のとおりです。

- PG - プログラム生成
- PS - プログラム送信
- PW - プログラム・ウェイト
- PR - プログラム・リード
- DP - ディスパッチ
- DC - デコード

C6000 DSP 命令は、フェッチ・パケットと実行パケットの 2 つにグループに分類されます。CPU は、フェッチ・パケットという、8 命令の固定サイズの組でメモリから命令をフェッチします。それらの命令はデコードされ、実行パケットという、並列に発行される命令の組みに分割されます。1 つの実行パケットは、1 ～ 8 個の命令で構成されています。そのため、単一のフェッチ・パケットは、複数の実行パケットで構成されることがあります。また、C64x+ DSP では、実行パケットが 2 つのフェッチ・パケットにまたがることもあります。パイプラインのプログラム・リード（PR）ステージには、フェッチ・パケットの並びの中にある実行パケットの並びを識別する役割があります。ディスパッチ（DP）ステージは、実行パケットを取り出して、各機能ユニットに送り出します。

フェッチ・パケットと実行パケットには相違があるため、フェッチ・パイプライン全体が毎サイクル進む必要はありません。むしろ、DP ステージが PR にあるフェッチ・パケットを完全に使い尽くした時点で、PR パイプライン・ステージは、プログラム・ウェイト（PW）ステージがその内容を PR ステージに進めることを許可します。PR 前のステージは、ギャップを埋めるために必要に応じて進みます。そのため、キャッシュ・ミスがない場合、DP ステージが現在のフェッチ・パケットから個々の実行パケットを取り出している間、フェッチ・パイプラインの初期ステージはストールされます。このようなストールを、ディスパッチ・ストールといいます。

C64x+ DSP は、パイプラインの初期ステージで、キャッシュ・ミスが保留されたままになっている間に、それらのステージが DP に向かって進むのを許すことで、このディスパッチ・ストールを上手く利用しています。キャッシュ・ミスは、PR、PW、PS の各パイプライン・ステージでペンディングされている場合があります。これらのキャッシュ・ストールを CPU に知らせる必要はありません。これは、DP ステージでパイプラインの PR ステージ内のフェッチ・パケットを処理している間に、ディスパッチ・ストールによって PR ステージをストールさせるためです。しかし、フェッチ・パケットが完全に使い尽くされると、PW ステージの内容を PR ステージに進める必要があります。この時点で、DP が PR から実行パケットをリクエストし、そこにまだ未処理のキャッシュ・ミスがある場合、CPU はストールします。

分岐が行われると、分岐先を含むフェッチ・パケットは分岐先が E1 パイプライン・ステージに到達するまで、フェッチ・パイプラインを通してサイクルごとに進みます。分岐先は、前述のディスパッチ・ストールを無効にします。結果的に、分岐先はミス・パイプライン化からは他の命令と同じような恩恵を得ることはありません。ただし、分岐先の直後に続くフェッチ・パケットは恩恵を受けます。分岐先に続くフェッチ・パケット内のコードはただちに実行されなくとも、分岐によってこのコードに対していくつかの連続したフェッチがトリガされます。そのため、該当コードのすべてのミスはパイプライン化されます。さらに、分岐が行われる前にリクエストされていたが、DP パイプライン・ステージに進めなかったフェッチ・パケットに対して、ストールが発生することはありません。

ミス・パフォーマンスは、コードの平均的な並列度に基づいて平均的なミス・ペナルティを招く、直線的な（分岐のない）コードにおいて連続したミスで測定されます。直線的なコードにおける長時間にわたるミスが持続する場合の平均的なミス・ペナルティについて、表 2-13 にまとめています。このコードは、L2SRAM からフェッチしたものです。2 種類の異なる設定が存在します。最初の設定の特長は、L2SRAM の、2 x 128 ビット・バンクの場合 、0 ウェイト・ステートということです。この設定は、DM644x デバイスで利用できます。2 番目の設定の特長は、L2SRAM の、4 x 128 ビット・バンクの場合、1 ウェイト・ステートということです。この設定は、C645x デバイスで利用できます。

表 2-13. L1P ミス・パイプライン化パフォーマンス（実行パケットごとの平均ストール数）

L2 タイプ	0 ウェイト・ステート、2 x 128 ビット・バンク		1 ウェイト・ステート、4 x 128 ビット・バンク	
実行パケットごとの命令数	L2 SRAM	L2 キャッシュ	L2 SRAM	L2 キャッシュ
1	0.000	0.000	0.000	0.000
2	0.001	0.497	0.167	0.499
3	0.501	1.247	0.751	1.249
4	0.997	1.997	1.329	1.999
5	1.499	2.747	1.915	2.749
6	2.001	3.497	2.501	3.499
7	2.497	4.247	3.079	4.249
8	2.999	4.997	3.665	4.999

2.8 パワーダウン・サポート

L1P メモリは、電力を節約するために、いくつかの方法でパワーダウンを行うことができます。

2.8.1 静的パワーダウン

L1P メモリは、メガモジュール全体がパワーダウンした場合にパワーダウンします。C64x+ メガモジュールをパワーダウンさせるのに必要なソフトウェア・シーケンスは、次のとおりです。

1. PDCCMD レジスタの MEGPD フィールドを 1 にセットして、パワーダウンをイネーブルします。
2. メガモジュールをウェイクアップする CPU 割り込みをイネーブルします。他の割り込みをすべてディスエーブルします。
3. IDLE 命令を実行します。

メガモジュールは、上記ステップ 2 でイネーブルされた割り込みが発生するまでパワーダウン・モードになったままです。

パワーダウン・コントローラ・コマンド・レジスタ（PDCCMD）については、第 9 章を参照してください。メガモジュールがパワーダウンしている間に、DMA アクセスが L1D、L1P、L2 のいずれかのメモリに対して行われた場合、PDC は 3 つのメモリ・コントローラをすべてウェイクアップします。PDC は DMA アクセスが処理されると、メモリ・コントローラを再度パワーダウンします。

注： ここで説明したように、メガモジュールをパワーダウンすることは、多くの場合、静的パワーダウンといいます。この用語は、このモードを説明するために使われます。これはこの機能が長時間にわたり使われていることが多いためです。動的パワーダウンという用語が本章で使われている場合は、パワーダウン・モードが限られた時間だけ使われていることを示します。

2.8.2 動的パワーダウン

L1P メモリは、CPU が SPLOOP バッファからコードを実行している間、自動的にパワーダウンします。SPLOOP バッファの詳細については、『TMS320C64x/C64x+ DSP CPU and Instruction Set Reference Guide』（資料番号 SPRU732）を参照してください。

2.8.3 機能に基づいたパワーダウン

L1P キャッシュがディスエーブル（000b を L1PCFG レジスタの MODE フィールドにライト）の場合、エネルギーをさらに節約するためにパワーダウン状態になります。

2.9 L1P メモリ保護

L1P メモリはメモリ保護をサポートし、多数のシステムで求められる堅牢性をもたらします。メモリ保護を行うレベルがいくつか用意されています。すべてのレベルがすべてのデバイスで利用できるとは限りません。詳細については、各デバイスのデータ・マニュアルを参照してください。C64x+ のメモリ保護の詳細については、第 8 章を参照してください。

2.9.1 L1P アクセス時に行われる保護チェック

L1P は L1D とは異なり、L1P メモリからの CPU によるプログラム・フェッチと CPU/DMA/IDMA による L1P メモリへのアクセスに関する各種メモリ保護ルールを実装しています。これ以降では、これらの違いについて詳細に説明します。

3 つのメモリ・コントローラにはすべて、メガモジュールの割り込みコントローラへ送られる 2 つの例外出力を備えているという特長があります。この例外出力の 1 つは、CPU によってトリガされる（「ローカルな」）メモリ例外が発生したことを示します。もう一方は、DMA によってトリガされる（「リモートの」）例外が発生したことを示します。ほとんどのプログラムは、CPU によってトリガされる例外入力を CPU の例外入力へ送り、DMA によってトリガされる入力を割り込み入力へ送る可能性があります。

2.9.1.1 CPU によるプログラム・フェッチ時に行われる保護チェック

L1P は、すべてのフェッチ時にメモリ保護チェックを行います。各フェッチ・パケットには、そのパケットに対応する 2 つのアクセス権限ビットがあります（表 2-14 を参照）。この 2 ビットで、キャッシュ・ライン内に、または L1P RAM/L1P ROM の対応する領域内に含まれるコードに対応する実行権限を決定します。L1P は、フェッチされたデータが最終的にどこから来たのかに関係なく、すべてのフェッチに対してアクセス権限をチェックした結果を出します。

表 2-14. フェッチごとにチェックされた許可ビット

ビット	説明
UX	ユーザ・モードで実行可能。
SX	スーパーバイザ・モードで実行可能。

2.9.1.2 CPU によるデータ・アクセス時に行われる保護チェック

CPU は、ロード / ストア命令を使用して L1P RAM/ROM に直接アクセスすることはできません。ただし、ロード / ストア命令を使用して L1P のコントロール・レジスタをアクセスすることはできます。

レジスタに対応するアクセス権限は、そのレジスタのリードをチェックします。L1P は、このレジスタへのライトをチェックしています。許可されていないライトが行われると、CPU によってトリガされたメモリ例外の信号を送ります。例外の詳細は L1PMPFAR/L1PMPFSR に記録されていて、L1P は CPU メモリ保護例外イベントの信号を割り込みコントローラへ送ります。

2.9.1.3 DMA/IDMA アクセス時に行われる保護チェック

L1P メモリへの DMA / IDMA アクセスは、L1P RAM / ROM に制限されています。DMA/IDMA は L1P キャッシュにアクセスできません。L1P キャッシュ下での L1P RAM へのアクセスは、L1P RAM に対応した保護エントリで管理されています。

DMA/IDMA アクセスはそれぞれ、対応するメモリ保護ページに対して SR/SW/UR/UW およびアクセス・メカニズム ID の各フィールドでチェックされます。DMA/IDMA は先頭の 16 ページのメモリを保護するために領域 0 インデックスへアクセスします。DMA/IDMA は 2 番目の 16 ページのメモリを保護するために領域 1 インデックスへアクセスします。

DMA または IDMA から L1P メモリへの無効なアクセスが行われると、L1P は例外が発生したことを示す信号を送ります。この例外の詳細は、L1PMPFAR/L1PMPFSR に記録されます。L1P は DMA メモリ保護例外イベントが発生したことを示す信号を割り込みコントローラに送ります。

2.9.2 メモリ・プロテクション・レジスタ

L1P 内でメモリ保護動作を管理するレジスタを示します。MMR は 3 つの主要なカテゴリに分類されます。

- メモリ・プロテクション・ページ・アトリビュート・レジスタ（MPPA）：ページ・アトリビュート・レジスタは、保護されたページごとに対応するアクセス権限を格納します。
- メモリ・プロテクション・ロック・レジスタ（MPLK）：ペリフェラルは、ハードウェアによるメモリ保護ロック機能を実現するために選択できます。ペリフェラルで他のリクエストを処理できない場合、ロックをかけてそのペリフェラルのメモリ保護エントリに対するすべての更新をディスエーブルします。
- メモリ・プロテクション・フォールト・レジスタ（MPFxR）：メモリ保護障害を生成する各ペリフェラルには、障害の詳細を記録するために MPFAR、MPFSR、MPFCR の各レジスタが用意されています。

表 2-15. メモリ・プロテクション・レジスタ

アドレス	略称	レジスタの説明	参照先
0184 A6xxh	L1PMPPA*xx*	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ	2.9.2.1 項
0184 A500h	L1PMPLK0	L1P メモリ・プロテクション・ロック・レジスタ 0（L1PMPLK0）	2.9.2.2.1 項
0184 A504h	L1PMPLK1	L1P メモリ・プロテクション・ロック・レジスタ 1（L1PMPLK1）	2.9.2.2.2 項
0184 A508h	L1PMPLK2	L1P メモリ・プロテクション・ロック・レジスタ 2（L1PMPLK2）	2.9.2.2.3 項
0184 A50Ch	L1PMPLK3	L1P メモリ・プロテクション・ロック・レジスタ 3（L1PMPLK3）	2.9.2.2.4 項
0184 A510h	L1PMPLKCMD	L1P メモリ・プロテクション・ロック・コマンド・レジスタ（L1PMPLKCMD）	2.9.2.2.5 項
0184 A514h	L1PMPLKSTAT	L1P メモリ・プロテクション・ロック・ステータス・レジスタ（L1PMPLKSTAT）	2.9.2.2 項
0184 A400h	L1PMPFAR	L1P メモリ・プロテクション・フォールト・アドレス・レジスタ（L1PMPFAR）	2.9.2.3.1 項
0184 A404h	L1PMPFSR	L1P メモリ・プロテクション・フォールト・セット・レジスタ（L1PMPFSR）	2.9.2.3.2 項
0184 A408h	L1PMPFCLR	L1P メモリ・プロテクション・フォールト・クリア・レジスタ（L1PMPFCLR）	2.9.2.3.3 項

2.9.2.1 メモリ・ページ・プロテクション・アトリビュート・レジスタ

表 2-16 に、メモリ・ページ・プロテクション・レジスタを示します。

L1P は 32 ページのメモリを保護する機能を実装しています。L1PMPPA0 ～ L1PMPPA15 は領域 0 に対応し、L1PMPPA16 ～ L1PMPPA31 は領域 1 にそれぞれ対応します。

特定のデバイス上で使用されるページ・サイズおよびページ数を確認するには、各デバイスのデータ・マニュアルを参照してください。

表 2-16. メモリ・ページ・プロテクション・アトリビュート・レジスタ

アドレス	略称	レジスタの説明	参照先
0184 A600h	L1PMPPA0	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 0	2.9.2.1.1 項
0184 A604h	L1PMPPA1	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 1	2.9.2.1.1 項
0184 A608h	L1PMPPA2	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 2	2.9.2.1.1 項
0184 A60Ch	L1PMPPA3	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 3	2.9.2.1.1 項
0184 A610h	L1PMPPA4	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 4	2.9.2.1.1 項
0184 A614h	L1PMPPA5	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 5	2.9.2.1.1 項
0184 A618h	L1PMPPA6	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 6	2.9.2.1.1 項
0184 A61Ch	L1PMPPA7	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 7	2.9.2.1.1 項
0184 A620h	L1PMPPA8	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 8	2.9.2.1.1 項
0184 A624h	L1PMPPA9	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 9	2.9.2.1.1 項
0184 A628h	L1PMPPA10	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 10	2.9.2.1.1 項
0184 A62Ch	L1PMPPA11	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 11	2.9.2.1.1 項
0184 A630h	L1PMPPA12	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 12	2.9.2.1.1 項
0184 A634h	L1PMPPA13	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 13	2.9.2.1.1 項
0184 A638h	L1PMPPA14	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 14	2.9.2.1.1 項
0184 A63Ch	L1PMPPA15	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 15	2.9.2.1.1 項
0184 A640h	L1PMPPA16	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 16	2.9.2.1.1 項
0184 A644h	L1PMPPA17	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 17	2.9.2.1.1 項
0184 A648h	L1PMPPA18	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 18	2.9.2.1.1 項
0184 A64Ch	L1PMPPA19	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 19	2.9.2.1.1 項
0184 A650h	L1PMPPA20	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 20	2.9.2.1.1 項
0184 A654h	L1PMPPA21	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 21	2.9.2.1.1 項
0184 A658h	L1PMPPA22	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 22	2.9.2.1.1 項
0184 A65Ch	L1PMPPA23	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 23	2.9.2.1.1 項
0184 A660h	L1PMPPA24	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 24	2.9.2.1.1 項
0184 A664h	L1PMPPA25	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 25	2.9.2.1.1 項
0184 A668h	L1PMPPA26	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 26	2.9.2.1.1 項
0184 A66Ch	L1PMPPA27	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 27	2.9.2.1.1 項
0184 A670h	L1PMPPA28	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 28	2.9.2.1.1 項
0184 A674h	L1PMPPA29	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 29	2.9.2.1.1 項
0184 A678h	L1PMPPA30	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 30	2.9.2.1.1 項
0184 A67Ch	L1PMPPA31	L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ 31	2.9.2.1.1 項

2.9.2.1.1 L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ（L1PMPPA*x*）

各ページのサイズは、領域ごとに、デバイスごとにそれぞれ異なります。ページによっては、特定のデバイスでは使用できない場合があります。デバッグする目的で、使用していないページにも、すべてゼロの値をセットするようにプログラムしてください。

特定のデバイス上で使用されるページ・サイズおよびページ数を確認するには、各デバイスのデータ・マニュアルを参照してください。

メモリ・ページ・プロテクション・アトリビュート・レジスタ（L1PMPPA*xx*）の一般的な構造を図 2-7 に示し、表 2-17 で説明します。

図 2-7. L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ（L1PMPPA*x*）

31															16
Reserved															
R-0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AID5	AID4	AID3	AID2	AID1	AID0	AIDX	LOCAL	Reserved		SR	SW	SX	UR	UW	UX
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0		R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

凡例：R/W = リード / ライト。-*n* = リセット後の値。

表 2-17. L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ（L1PMPPA*x*）フィールドの説明

ビット	フィールド	値	説明
31-16	Reserved	0	予約。
15	AID5		ID = 5 からのアクセスを制御します。
		0	アクセスは拒否。
		1	アクセスは許可。
14	AID4		ID = 4 からのアクセスを制御します。
		0	アクセスは拒否。
		1	アクセスは許可。
13	AID3		ID = 3 からのアクセスを制御します。
		0	アクセスは拒否。
		1	アクセスは許可。
12	AID2		ID = 2 からのアクセスを制御します。
		0	アクセスは拒否。
		1	アクセスは許可。
11	AID1		ID = 1 からのアクセスを制御します。
		0	アクセスは拒否。
		1	アクセスは許可。
10	AID0		ID = 0 からのアクセスを制御します。
		0	アクセスは拒否。
		1	アクセスは許可。
9	AIDX		ID>=6 からのアクセスを制御します。
		0	アクセスは拒否。
		1	アクセスは許可。
8	LOCAL		CPU からローカル・メモリ（L1/L2）へのアクセスを制御します。
		0	アクセスは拒否。
		1	アクセスは許可。

表 2-17. L1P メモリ・ページ・プロテクション・アトリビュート・レジスタ（L1PMPPA*x*）フィールドの説明（続き）

ビット	フィールド	値	説明
7-6	Reserved	0	予約。
5	SR		スーパーバイザによるリード・アクセス・タイプ。
		0	通常動作。
		1	スーパーバイザによるリード・リクエストを示します。
4	SW		スーパーバイザによるライト・アクセス・タイプ。
		0	通常動作。
		1	スーパーバイザによるライト・リクエストを示します。
3	SX		スーパーバイザによる実行アクセス・タイプ。
		0	通常動作。
		1	スーパーバイザによる実行リクエストを示します。
2	UR		ユーザによるリード・アクセス・タイプ。
		0	通常動作。
		1	ユーザによるリード・リクエストを示します。
1	UW		ユーザによるライト・アクセス・タイプ。
		0	通常動作。
		1	ユーザによるライト・リクエストを示します。
0	UX		ユーザによる実行アクセス・タイプ。
		0	通常動作。
		1	ユーザによる実行リクエストを示します。

2.9.2.2 メモリ・プロテクション・ロック・レジスタ

L1P は、メモリ・プロテクション・レジスタへのライト・アクセスを制御するために、64 ビットのロック・レジスタを実装しています。

表 2-18 に、メモリ・プロテクション・ロック・レジスタを示します。これらのレジスタのメモリ・アドレスについては、各デバイスのデータ・マニュアルを参照してください。

表 2-18. メモリ・プロテクション・ロック・レジスタ

アドレス	略称	レジスタの説明	参照先
0184 A500h	L1PMPLK0	L1P メモリ・プロテクション・ロック・レジスタ 0（L1PMPLK0）	2.9.2.2.1 項
0184 A504h	L1PMPLK1	L1P メモリ・プロテクション・ロック・レジスタ 1（L1PMPLK1）	2.9.2.2.2 項
0184 A508h	L1PMPLK2	L1P メモリ・プロテクション・ロック・レジスタ 2（L1PMPLK2）	2.9.2.2.3 項
0184 A50Ch	L1PMPLK3	L1P メモリ・プロテクション・ロック・レジスタ 3（L1PMPLK3）	2.9.2.2.4 項
0184 A510h	L1PMPLKCMD	L1P メモリ・プロテクション・ロック・コマンド・レジスタ（L1PMPLKCMD）	2.9.2.2.5 項
0184 A514h	L1PMPLKSTAT	L1P メモリ・プロテクション・ロック・ステータス・レジスタ（L1PMPLKSTAT）	2.9.2.2.6 項

2.9.2.2.1 L1P メモリ・プロテクション・ロック・レジスタ 0（L1PMPLK0）

L1P メモリ・プロテクション・ロック・レジスタ 0（L1PMPLK0）を図 2-8 に示します。

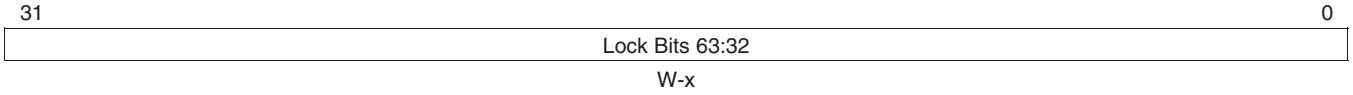
図 2-8. L1P メモリ・プロテクション・ロック・レジスタ 0（L1PMPLK0）

凡例：W = ライト専用。-*x* = 値は不定。詳細については、各デバイスのデータ・マニュアルを参照してください。

2.9.2.2.2 L1P メモリ・プロテクション・ロック・レジスタ 1（L1PMPLK1）

L1P メモリ・プロテクション・ロック・レジスタ 1（L1PMPLK1）を図 2-9 に示します。

図 2-9. L1P メモリ・プロテクション・ロック・レジスタ 1（L1PMPLK1）

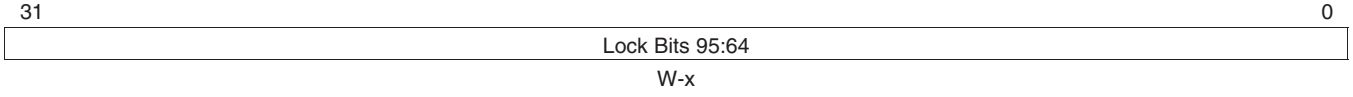


凡例：W = ライト専用。-*x* = 値は不定。詳細については、各デバイスのデータ・マニュアルを参照してください。

2.9.2.2.3 L1P メモリ・プロテクション・ロック・レジスタ 2（L1PMPLK2）

L1P メモリ・プロテクション・ロック・レジスタ 2（L1PMPLK2）を図 2-10 に示します。

図 2-10. L1P メモリ・プロテクション・ロック・レジスタ 2（L1PMPLK2）

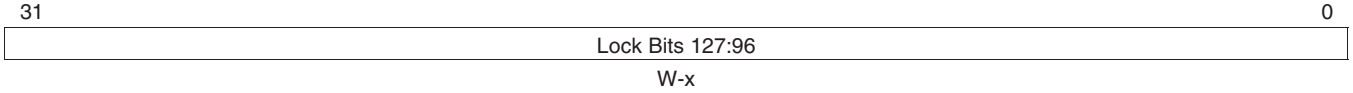


凡例：W = ライト専用。-*x* = 値は不定。詳細については、各デバイスのデータ・マニュアルを参照してください。

2.9.2.2.4 L1P メモリ・プロテクション・ロック・レジスタ 3（L1PMPLK3）

L1P メモリ・プロテクション・ロック・レジスタ 3（L1PMPLK3）を図 2-11 に示します。

図 2-11. L1P メモリ・プロテクション・ロック・レジスタ 3（L1PMPLK3）

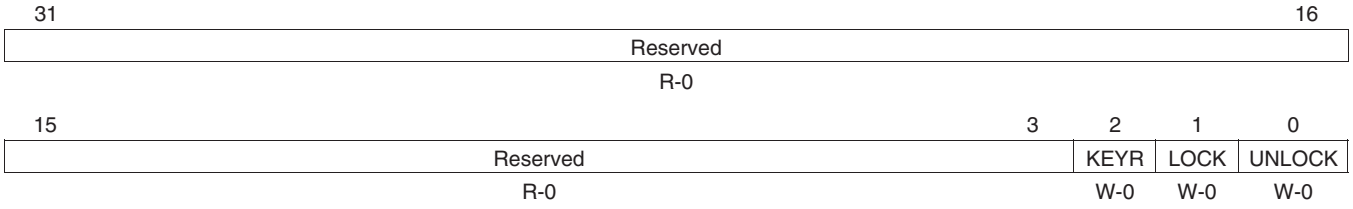


凡例：W = ライト専用。*-x* = 値は不定。詳細については、各デバイスのデータ・マニュアルを参照してください。

2.9.2.2.5 L1P メモリ・プロテクション・ロック・コマンド・レジスタ（L1PMPLKCMD）

L1P メモリ・プロテクション・ロック・コマンド・レジスタ（L1PMPLKCMD）を図 2-12 に示し、表 2-19 で説明します。

図 2-12. L1P メモリ・プロテクション・ロック・コマンド・レジスタ（L1PMPLKCMD）



凡例：R = リード専用。W = ライト専用。*-n* = リセット後の値。

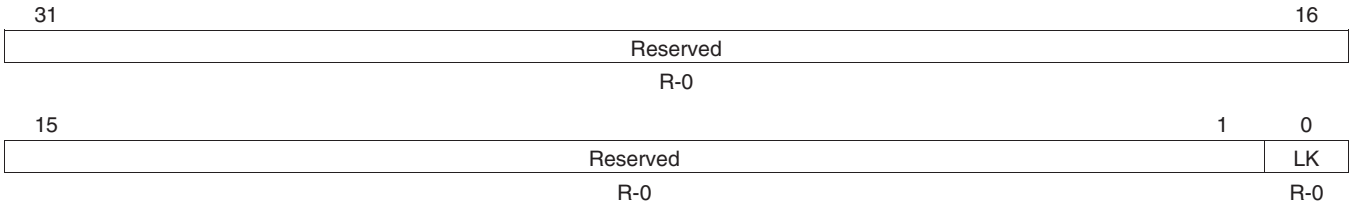
表 2-19. L1P メモリ・プロテクション・ロック・コマンド・レジスタ（L1PMPLKCMD）フィールドの説明

ビット	フィールド	値	説明
31-3	Reserved	0	予約。
2	KEYR		ステータスをリセットします。
		0	影響なし。
		1	ステータスをリセットします。
1	LOCK		ロック・シーケンスを完了するためのインターフェイス。
		0	影響なし。
		1	ソフトウェアがシーケンスを正しく実行した場合に、ロックします。
0	UNLOCK		ロック解除シーケンスを完了するためのインターフェイス。
		0	影響なし。
		1	ソフトウェアがシーケンスを正しく実行した場合に、ロックを解除します。

2.9.2.2.6 L1P メモリ・プロテクション・ロック・ステータス・レジスタ（L1PMPLKSTAT）

L1P メモリ・プロテクション・ロック・ステータス・レジスタ（L1PMPLKSTAT）を図 2-13 に示し、表 2-20 で説明します。

図 2-13. L1P メモリ・プロテクション・ロック・ステータス・レジスタ（L1PMPLKSTAT）



凡例：R = リード専用。*-n* = リセット後の値。

表 2-20. L1P メモリ・プロテクション・ロック・ステータス・レジスタ（L1PMPLKSTAT）フィールドの説明

ビット	フィールド	値	説明
31-1	Reserved	0	予約。
0	LK		ロックの現在のステータスを示します。
		0	ロックは解除されています。
		1	ロックされています。

上記に示したように、メモリ保護アーキテクチャは、最大で 128 ビットのロック・サイズを許容します。L1P は、ロック・インターフェイスに対し 64 ビット・ロックのみを実装しています。そのため、L1PMPLCK1、L1PMPLCK2、L1PMPLCK3 にライトされた値は無視されます。128 ビットより短いキーに関するロック・メカニズムの動作については、第 8 章を参照してください。

2.9.2.3 メモリ・プロテクション・フォールト・レジスタ

表 2-21 に、メモリ・プロテクション・フォールト・レジスタを示します。これらのレジスタのメモリ・アドレスについては、各デバイスのデータ・マニュアルを参照してください。

例外発生後にプログラムでメモリ保護障害を診断するために、L1P には障害に関する情報を格納する専用の 2 つのレジスタが実装されています。

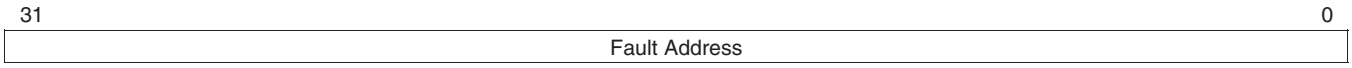
表 2-21. メモリ・プロテクション・フォールト・レジスタ

アドレス	略称	レジスタの説明	参照先
0184 A400h	L1PMPFAR	L1P メモリ・プロテクション・フォールト・アドレス・レジスタ	2.9.2.3.1 項
0184 A404h	L1PMPFSR	L1P メモリ・プロテクション・フォールト・セット・レジスタ	2.9.2.3.2 項
0184 A408h	L1PMPFCLR	L1P メモリ・プロテクション・フォールト・クリア・レジスタ	2.9.2.3.3 項

2.9.2.3.1 L1P メモリ・プロテクション・フォールト・アドレス・レジスタ（L1PMPFAR）

L1P メモリ・プロテクション・フォールト・アドレス・レジスタ（L1PMPFAR）を図 2-14 に示し、表 2-22 で説明します。

図 2-14. L1P メモリ・プロテクション・フォールト・アドレス・レジスタ（L1PMPFAR）



凡例：R = リード専用。*-x* = 値は不定。詳細については、各デバイスのデータ・マニュアルを参照してください。

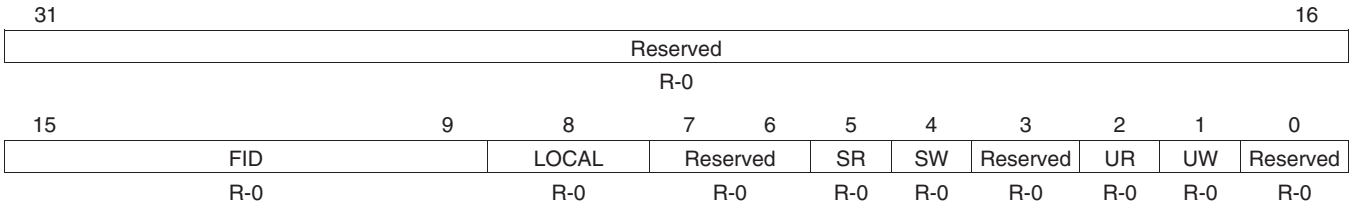
表 2-22. L1P メモリ・プロテクション・フォールト・アドレス・レジスタ（L1PMPFAR）フィールドの説明

ビット	フィールド	値	説明
31-0	Fault Address	0 ～ FFFF FFFFh	予約。

2.9.2.3.2 L1P メモリ・プロテクション・フォールト・セット・レジスタ（L1PMPFSR）

L1P メモリ・プロテクション・フォールト・セット・レジスタ（L1PMPFSR）を図 2-15 に示し、表 2-23 で説明します。

図 2-15. L1P メモリ・プロテクション・フォールト・セット・レジスタ（L1PMPFSR）



凡例：R = リード専用。*-n* = リセット後の値。

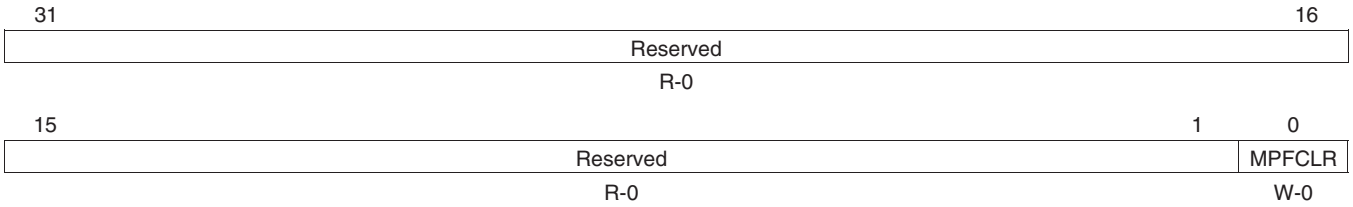
表 2-23. L1P メモリ・プロテクション・フォールト・セット・レジスタ（L1PMPFSR）フィールドの説明

ビット	フィールド	値	説明
31-16	Reserved	0	予約。
15-9	FID	0 ～ 7Fh	ビット 6:0 は障害を引き起こしたリクエスタの ID を示します。ID の幅が 7 ビットより狭い場合、残りのビットは 0 を返します。ID の幅が 7 ビットより広い場合、他のビットは切り捨てられます。LOCAL = 1 の場合、FID = 0。
8	LOCAL	0	通常動作。
		1	アクセスは「ローカル」。
7-6	Reserved	0	予約。
5	SR		スーパーバイザによるリード・アクセス・タイプ。
		0	通常動作。
		1	スーパーバイザによるリード・リクエストを示します。
4	SW		スーパーバイザによるライト・アクセス・タイプ。
		0	通常動作。
		1	スーパーバイザによるライト・リクエストを示します。
3	Reserved	0	予約。
2	UR		ユーザによるリード・アクセス・タイプ。
		0	通常動作。
		1	ユーザによるリード・リクエストを示します。
1	UW		ユーザによるライト・アクセス・タイプ。
		0	通常動作。
		1	ユーザによるライト・リクエストを示します。
0	Reserved	0	予約。

2.9.2.3.3 L1P メモリ・プロテクション・フォールト・クリア・レジスタ（L1PMPFCLR）

L1P メモリ・プロテクション・フォールト・クリア・レジスタ（L1PMPFCLR）を図 2-16 に示し、表 2-24 で説明します。

図 2-16. L1P メモリ・プロテクション・フォールト・クリア・レジスタ（L1PMPFCLR）



凡例：R = リード専用。W = ライト専用。*-n* = リセット後の値。

表 2-24. L1P メモリ・プロテクション・フォールト・クリア・レジスタ（L1PMPFCLR）フィールドの説明

ビット	フィールド	値	説明
31-1	Reserved	0	予約。
0	MPFCLR		L1DMPFAR レジスタをクリアするためのコマンド。
		0	影響なし。
		1	L1DMPFAR および L1DMPFCR レジスタをクリアします。

L1PMPFAR および L1PMPFSR レジスタは、1 つの障害に関する十分な情報のみを格納します。一般的に、ハードウェアは最初の障害に関する情報を記録し、その障害のみの例外を生成します。L1P は、「ローカルな」（CPU によってトリガされる）および「リモートの」（DMA/IDMA によってトリガされる）障害という概念を使用しています。L1P によって、「ローカルな」障害は「リモートの」障害を置き換えることができ、かつ新たな例外を発生させることができます。

障害情報は、ソフトウェアで L1PMPFCR レジスタの MPFCLR ビットに 1 をライトしてクリアするまで、保持されます。L1PMPFCR レジスタの MPFCLR ビットに 0 をライトしても影響はありません。

2.9.2.3.4 メモリ・プロテクション・レジスタへのアクセス時に行われる保護チェック

L1P は、メモリ・プロテクション・レジスタ自体にアクセス権限チェック機能を実装しています。ルールは次のとおりです。

- すべてのリクエスタは、いつでもどんな状況でも任意のメモリ・プロテクション（MP）レジスタを *リード*できます。ただし、L1PMPLK0 ～ L1PMPLK3 は除きます。
- スーパーバイザはレジスタを *ライト*できます。

表 2-25 に、役割ごとにアクセスできる L1P メモリ・プロテクション・レジスタおよびメガモジュールで行われる保護チェックの内容について概要を示します。

表 2-25. L1P メモリ・プロテクション・レジスタのアクセス権限

レジスタ	スーパーバイザ	ユーザ
L1PMPFAR	R	R
L1PMPFSR	R	R
L1PMPFCR	W	/
L1PMPLK0	W	/
L1PMPLK1	W	/
L1PMPLK2	W	/
L1PMPLK3	W	/
L1PMPLKCMD	W	/
L1PMPLKSTAT	R	R
L1PMPPAxx	R/W	R

レベル 1 データ・メモリ / キャッシュ

項目 ページ

3.1 はじめに

3.1.1 レベル 1 データ（L1D）メモリ / キャッシュの目的

L1D メモリ / キャッシュの目的は、データの処理性能を最大限に高めることです。L1D メモリ / キャッシュはさまざまな設定を行えるため、柔軟性をもってシステム内で L1D キャッシュや L1D メモリを使用することができます。

3.1.2 機能

L1D メモリ / キャッシュが提供している機能は次のとおりです。

- 設定可能な L1D キャッシュ・サイズ：0K、4K、8K、16K、32K
- メモリ保護
- キャッシュ・ブロックおよびグローバルなコヒーレンス動作

3.1.3 用語と定義

本章で使用する用語の詳細な定義については、本書の付録 A および付録 B を参照してください。付録 A では、本書全体で使用している一般的な用語について説明し、付録 B ではメモリ / キャッシュ・アーキテクチャ関連の用語を定義しています。

3.2 L1D メモリ・アーキテクチャ

3.2.1 L1D メモリ

L1D メモリは、メモリ・マップド RAM と ROM を合わせて最大で 1MB までサポートします。L1D メモリは、同一メガモジュール内の L1D キャッシュ、L1P キャッシュ、L2 キャッシュのいずれにもキャッシュされない場合があります。

L1D メモリのベース・アドレスには、1MB 境界の制約があります。L1D メモリの合計サイズは、16KB の倍数にする必要があります。

L1D メモリは、同一メガモジュール内の L1D キャッシュ、L1P キャッシュ、L2 キャッシュのいずれにもキャッシュされません。

3.2.1.1 L1D 領域

L1D メモリは、2 つの領域に分割されています（L1D 領域 0 および L1D 領域 1）。このような領域には、次の異なる機能があります。

- それぞれの領域には、別々のメモリ保護エントリがある
- L1D 領域 1 の部分は、データ・キャッシュに変換できる

この 2 つの領域は、メモリ内では連続しています。領域 0 のサイズは、0KB（そのためディスエーブル）または 16K ～ 512KB の範囲の 2 の累乗で表される数値になります。領域 1 は、領域 0 の最後から始まります。そのサイズは、16K の倍数で 16K ～ 512KB の範囲にあります。領域 1 のサイズは、領域 0 のサイズ以下にする必要があります（領域 0 がイネーブルの場合）。

2 つの L1D 領域では、メモリ保護エントリは 2 つのセットに分割されています。L1D は 32 ページのメモリを保護できます（3.7.2 項を参照）。先頭の 16 ページは領域 0 を、次の 16 ページは領域 1 をそれぞれ対象とします。領域 0 のサイズが 0KB の場合、メモリ保護ページは使われません。

実際のメモリ構成は、デバイスごとに異なります。詳細については、各デバイスのデータシートを参照してください。

3.3 L1D キャッシュ

C64x+ L1D メモリ / キャッシュ・アーキテクチャは、L1D 領域 1 の一部またはすべてをリード・アロケート、ライトバック、2 ウェイ・セット・アソシアティブ・キャッシュに変換できます。L1D キャッシュは、大容量のシステム・メモリを保持しながら、高速なクロック・レートでデータをリード / ライトする効率を高める上で必要です。このキャッシュは、低速のシステム・メモリ間でリード / ライト時に発生するレイテンシの多くを見かけ上なくすようにします。

キャッシュ・コントローラは、4K ～ 32K のキャッシュ・サイズをサポートできるように設計されています。ただし、特定のデバイスでは、領域 1 の L1D RAM は 32K 未満しか実装されていない場合があります。

L1D キャッシュは L1D メモリを、L1D 領域 1 の L1D メモリの最上位アドレスから始まり、下方向へキャッシュに変換します。

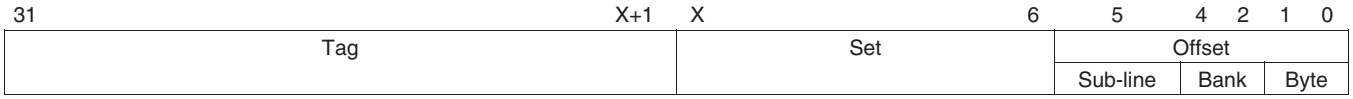
L1D メモリは、リセット時に「すべて RAM」または「最大のキャッシュ」として初期化されます。デバイス固有の動作については、各デバイスのデータ・マニュアルを参照してください。

L1D キャッシュの動作は、いくつかのレジスタによって制御されています。このようなレジスタの詳細については、3.4 節を参照してください。

3.3.1 L1D キャッシュ・アーキテクチャ

L1D キャッシュは、2 ウェイ・セット・アソシアティブ・キャッシュです。つまり、システムの物理メモリ・ロケーションはいずれも、内蔵可能なキャッシュに指定できるロケーションを 2 つもっているということです。CPU が一部のデータにアクセスしようとした場合、L1D はリクエストされたアドレスが L1D キャッシュのどのロケーション（ウェイ）に存在するかチェックする必要があります。これを行うには、CPU が提供する 32 ビット・アドレスを 6 つのフィールドに分割します（図 3-1 を参照）。

図 3-1. データ・アクセス・アドレス構成



6 ビットからなる Offset は、L1D ラインのサイズが 64 バイトであるということを示します。キャッシュ制御ロジックは、このアドレスのビット 0 ～ 5 を無視します（Byte、Bank、Sub-line の各フィールド）。ビット 0 ～ 5 だけが、Bank 内でアクセスするために Bank の位置、バイトを決定します。そのため、キャッシュのタグ比較ロジックとは無関係です。Set フィールドは、データが存在する L1D キャッシュ・ライン・アドレスを示します（データがキャッシュされている場合）。Set フィールドの幅は、キャッシュとして構成された L1D の量によって異なります（下記の表 3-1 を参照）。Set フィールドを使用して、有効なビットだけでなく、すでにキャッシュされているデータに対してそのアドレスから各ロケーション（ウェイ）でそのタグを調べチェックしてください。そうすれば、Tag のアドレスがキャッシュに保持されている有効なアドレスを実際に表しているかどうかがわかります。

Tag フィールドは、このアドレスの上位部分で、データ・エレメントの正しい物理ロケーションを識別します。このキャッシュは、L1D キャッシュの両方のロケーション（ウェイ）で、タグを格納されているタグと比較します。

リード時に Tag の 1 つが一致し、かつ対応する有効なビットがセットされると、「ヒット」になり、データ・キャッシュは L1D キャッシュからデータを CPU に直接戻します。それ以外の場合、リードは「ミス」になり、リクエストがシステムの他の L1D キャッシュ・ロケーションからデータをフェッチするために、レベル 2（L2）メモリに送られている間、CPU はストールします。

また CPU は、データを L1D へライトすることもできます。CPU がストア命令を実行する場合、L1D は CPU がリードのために実行したものと同じ Tag を比較します。有効な一致する Tag が見つかると、ライトは「ヒット」になり、データは L1D キャッシュ・ロケーションに直接ライトされます。それ以外の場合、ライトは「ミス」になり、データは L1D ライト・バッファのキューに積まれます。このバッファは、ライト・ミス時に CPU をストールさせないようにするために使われます。CPU はライト時にデータが戻ってくるまで待てないため、L2 アクセス中にストールする理由はありません。

L1D キャッシュを設定すると、Set と Tag フィールドのサイズが決定されます（表 3-1 を参照）。

表 3-1. データ・アクセス・アドレスの Set フィールドの幅

L1DCFG レジスタの L1DMODE 設定	L1D キャッシュの量	'X' ビットの位置	説明
000b	0K	N/A	L1D はすべての RAM
001b	4K	10	32 本の L1D キャッシュ・ライン
010b	8K	11	64 本の L1D キャッシュ・ライン
011b	16K	12	128 本の L1D キャッシュ・ライン
100b	32K	13	256 本の L1D キャッシュ・ライン
101b	予約。32K にマップ。		
110b			
111b	「最大のキャッシュ」。32K にマップ。		

データ・キャッシュのもう 1 つの特性は、L1D キャッシュからのデータを L2 に追い出すことができるということです。CPU は L1D キャッシュの内容を変更できるため、実際の物理ロケーションにあるデータを更新できる必要があります。このようなことが起きるのは、新たな L1D ラインで変更されたラインを置き換える場合、あるいは CPU が L1D キャッシュに対してソフトウェア制御を通じて、変更されたデータをライトバックするように指示した場合です。

3.3.2 リプレースメントおよびアロケーション処理

L1D キャッシュは、すべてのキャッシュ設定で固定された 2 ウェイ・セット・アソシアティブ・キャッシュとして動作します。これは、システム・メモリの各ロケーションは L1D キャッシュの 2 つのロケーションのいずれかに存在するということです。

L1D キャッシュは、リード・アロケート専用キャッシュです。これは、L1D キャッシュがリード・ミス時にのみ 64 バイトのライン全体をフェッチすることになるということです。ライト・ミスは、L1D ライト・バッファから L2 へ直接送られます。リプレースメント処理は、最低使用頻度（LRU）の L1D ラインを要求し、新たなラインで置き換えます。これにより、いつでも L1D キャッシュの最高使用頻度のデータが保持されます。

L1D は、ライトバック・キャッシュです。ライト・ヒットは、L1D 内で直接処理されます。更新は、L2 にも他のメモリ・システムにもただちに渡されることはありません。キャッシュ・ラインが変更された場合、そのキャッシュ・ラインに対応する「ダーティ・ビット」は 1 にセットされます。L1D がダーティ・ラインのみをライトバックするのは、そのダーティ・ラインを追い出して新たにキャッシュされたデータの空きを確保した場合か、プログラムがコヒーレンス操作を手動で開始してライトバックを強制した場合です。

3.3.3 L1D のモード変更動作

L1D のキャッシュ・サイズを実行時に設定することができます。プログラムで L1D のキャッシュ・サイズを選択するには、リクエストされたモードを L1DCFG レジスタの L1DMODE フィールドにライトします。

L1DCFG レジスタの L1DMODE フィールドは、表 3-2 に従って L1D キャッシュ・モードを選択します。

表 3-2. L1DCFG の L1DMODE で指定されるキャッシュ・サイズ

L1DCFG レジスタの L1DMODE 設定	L1D キャッシュの量
000b	0K
001b	4K
010b	8K
011b	16K
100b	32K
101b	予約。32K にマップ。
110b	
111b	「最大のキャッシュ」。32K にマップ。

L1D キャッシュ・モードの実際の範囲は、L1D 領域 1 のサイズの制約を受けます。

一般に、L1DMODE の値を大きくしたら、キャッシュ・サイズの指定も大きくします（実装されている L1D メモリの最大サイズまで）。L1D の最大キャッシュ・サイズは、「L1D 領域 1 の RAM サイズに収まる最大の 2 の累乗」と 32K の小さい方になります。

たとえば、L1D 領域 1 のサイズがわずか 16K しかない場合、L1D キャッシュは 16K より大きくすることはできません。この場合、011b ～ 111b のエンコーディングは、16K にマップされます。このようなデバイスでは、L1DMODE の設定が 100b ～ 111b の場合、32K キャッシュ・モードではなく、16K キャッシュ・モードが選択されます。つまり、モード 000b ～ 011b ではリクエストされたサイズ 0K ～ 16K が常に選択されます。モード 100b ～ 111b では、L1D のメモリ・サイズによって指定された最大サイズが選択されます（16K または 32K）。

このポリシーに基づいて、一定量のキャッシュしか必要としないプログラムでは、上限値に対応する値をプログラムしてください。「できるだけ多くのキャッシュ」を必要とするプログラムでは、L1DMODE に 111b をプログラムしてください。

プログラムでキャッシュ・モードの変更を開始する場合、L1D キャッシュ自体はデータが損失しないように現在の内容をライトバックし、インバリデートします。

ライトバック・インバリデートは、正しいキャッシュ動作を保証し、キャッシュされたデータが失われないことを確実にするために必要です。ただし、アドレス可能な L1D メモリ・ロケーションがキャッシュされるため、データを損失しないようにするには十分とはいえません。そのため、L1D キャッシュ・モードを安全に変更するために、アプリケーションは表 3-3 に示す手順に従う必要があります。

表 3-3. L1D モードの切り換え

変更前のモード	変更後のモード	プログラムは次の手順を実行する必要がある
L1D キャッシュをまったく使用しないか、一部使用するモード	L1D キャッシュをより多く使用するモード	1. DMA、IDMA を使用して、L1D RAM の影響を受ける範囲から必要なデータをコピーします。 2. 目的のキャッシュ・モードを L1DCFG レジスタの L1DMODE フィールドにライトします。 3. L1DCFG をリードバックします。これにより、モード変更が完了するまで CPU はストールします。
L1D キャッシュを一部使用するモード	L1D キャッシュをより少なく、あるいはまったく使用しないモード	1. 目的のキャッシュ・モードを L1DCFG レジスタの L1DMODE フィールドにライトします。 2. L1DCFG をリードバックします。これにより、モード変更が完了するまで CPU はストールします。

3.3.4 L1D フリーズ・モード

L1D キャッシュは、アプリケーションのフリーズ・モードを直接サポートします。このモードを使用すると、リアルタイム・アプリケーションは、割り込みハンドラなどのコードのさまざまなセクションで L1D から追い出されるデータ量を制限することができます。L1D フリーズ・モードは L1D キャッシュにのみ影響を与えます。L1D RAM は、フリーズ・モードの影響を受けません。

L1D キャッシュは、フリーズ・モードで通常通りリード・ヒットおよびライト・ヒットを処理します。ただし、LRU ビットが変更されないというわずかな例外があります。リード・ヒットでは、キャッシュからデータが戻されます。ライト・ヒットは、キャッシュ・ラインに対してキャッシュされたデータを更新し、必要に応じてそのデータをダーティとマークします。LRU ビットは更新されません。LRU ビットは、影響を受けるキャッシュ・ラインに対する最低使用頻度のウェイを示します。フリーズ・モードでは、L1D キャッシュはリード・ミス時に新たなキャッシュ・ラインを割り当てません。また、フリーズ・モードでは既存のキャッシュ内容を追い出すことはありません。通常、L1D ライト・バッファのライト・ミスは、キューに積まれます。

フリーズ・モードでは通常通り、L1D キャッシュはプログラムによって開始されるキャッシュ制御（ライトバック、インバリデート、ライトバック・インバリデート、モード変更）だけでなく、L2 から発行されるキャッシュ・コヒーレンス・コマンドに応答します（スヌープ・リード、スヌープ・ライト）。L1D のフリーズ・モードは、L2 がキャッシュ・ラインを割り当てるかどうかには影響を与えません。同様に、L2 のフリーズ・モードは、L1D がキャッシュ・ラインを割り当てるかどうかには影響を与えません。

L1DCC レジスタの OPER フィールドは、L1D フリーズ・モードを制御します。OPER フィールドに 1 をライトすると、CPU は L1D をフリーズ・モードに設定します。L1DCC レジスタの OPER フィールドに 0 をライトすると、CPU は L1D を正常動作に戻します。

L1DCC レジスタの POPER フィールドは、OPER フィールドの 1 つ前の値を保持しています。L1DCC レジスタの OPER フィールドの値は、L1DCC レジスタへライトすると、L1DCC レジスタの POPER フィールドにコピーされます。これにより、（OPER フィールドの 1 つ前の値を保存するために）L1DCC レジスタのリードにかかるサイクル・コストが軽減されます。もし仮に POPER フィールドが L1DCC レジスタにない場合、プログラムで以前の動作モードを記録してから OPER フィールドの値のリードとライトを行い、再度リードしてキャッシュを完全にフリーズする必要があります。POPER フィールドが L1DCC レジスタにある場合、この動作により 1 回ライトしてからリードするだけになります。

L1DCC レジスタにライトした場合、次の処理が行われます。

1. OPER フィールドの内容を L1DCC レジスタの POPER フィールドにコピーします。
2. POPER フィールドの以前の内容が失われます。
3. OPER フィールドは、CPU が L1DCC レジスタのビット 0 にライトした値に従って更新されます。そのため、L1DCC レジスタにライトすると、このレジスタの OPER フィールドのみが変更されます。

L1PCC レジスタが確実に更新されるようにするためには、ソフトウェアで L1PCC レジスタへライトしてから L1PCC をリードします。これにより、リクエストしたモードが有効になることが保証されます。

プログラムでは、1 回のライトで POPER フィールドを直接変更することができません。

L1DCC レジスタの OPER および POPER フィールドを使用する目的は、他の場合に必要となる、重要な CPU サイクルのペナルティおよびリード - ライト - 再リードの順番に関わるコード・サイズを回避することです。したがって、アプリケーションは L1D を迅速にフリーズし、L1D の以前「フリーズしていた」ステートを記録することができます。簡潔な一連のコードを例 3-1 に示します。

例 3-1. L1D を迅速にフリーズさせるコード・シーケンスの例

```
    MVKL    L1DCC,      A0      ; Point to L1DCC
    MVKH    L1DCC,      A0      ;
||  MVK     1,       B0      ; OPER encoding for 'freeze'

    STW     B0,         *A0[0]  ; Write 1 to L1DCC.OPER
    LDW     *A0[0],     A1      ; Read L1DCC to get L1DCC.POPER
    NOP     4
    ; At this point, L1D is frozen, and the CPU has the old OPER value
    ; in bit 16 of A1.
```

L1DCC レジスタを使用すると、キャッシュをフリーズしたときと同じ方法で、キャッシュのフリーズを解除できます。簡潔な一連のコードを例 3-2 に示します。

例 3-2. L1DCC の OPER ビットを復元するコード・シーケンスの例

```
    MVKL    L1DCC,      A0      ; Point to L1DCC
    MVKH    L1DCC,      A0      ;
||  SHRU    A1,   16,   A1      ; Shift POPER field into OPER's position

    STW     A1,         *A0[0]  ; Write to L1DCC, restoring old value of OPER
    LDW     *A0[0],     A1      ; Read back L1DCC to ensure change is complete
    NOP     4
    ; At this point, L1D is in its previous state (frozen or unfrozen).
```

注： L1D と L1P は両方とも、このようなメカニズムでフリーズ・モードを実現しています（L1P の実装方法の詳細については、第 2 章を参照してください）。多くの場合において、2 つのキャッシュを同時にフリーズさせることが望ましいです。したがって、L1DCC レジスタおよび L1PCC レジスタへ連続してライトしてから L1DCC レジスタおよび L1PCC レジスタの両方をリードするのは、L1P と L1D が両方ともフリーズしたことを確認できる確実な方法です。例 3-3 に、L1P と L1D の両方をフリーズさせるコード・シーケンスを示します。

例 3-3. L1P と L1D の両方を同時にフリーズさせるコード・シーケンスの例

```
    MVKL    L1DCC,      A0      ; \
||  MVKL    L1PCC,      B0      ;  |__ Generate L1DCC pointer in A0
    MVKH    L1DCC,      A0      ;  |   and L1PCC pointer in B0
||  MVKH    L1PCC,      B0      ; /
||  MVK     1,       A1      ; \___ OPER encoding for 'freeze'
||  MVK     1,       B1      ; /    in both A1 and B1.

    STW     A1,         *A0     ; Write to L1DCC.OPER
||  STW     B1,         *B0     ; Write to L1PCC.OPER

    LDW     *A0,        A1      ; Get old freeze state into A1 from L1DCC
||  LDW     *B0,        B1      ; Get old freeze state into B1 from L1PCC

    NOP     4
    ; At this point, L1P and L1D are frozen.
    ; The old value of L1DCC.OPER is in bit 16 of A1.
    ; The old value of L1PCC.OPER is in bit 16 of B1.
```

3.3.5 プログラムによって開始されるキャッシュ・コヒーレンス動作

C64x+ L1D キャッシュ・アーキテクチャは、プログラムによって開始されるキャッシュ・コヒーレンス動作をサポートします。これらの動作は、アドレス・ブロックまたは L1D キャッシュ全体に作用します。

サポートされているキャッシュ・コヒーレンス動作は、次のとおりです。

- インバリデート：有効なキャッシュ・ラインは、無効になります。影響を受けるキャッシュ・ラインの内容は、廃棄されます。
- ライトバック：すべてのダーティ・キャッシュ・ラインの内容は、より低いレベルのメモリへライトされます。
- ライトバック・インバリデート：ライトバック動作に続きインバリデートが行われます。ダーティ・キャッシュ・ラインの内容のみが、低レベル・メモリへライトされますが、すべてのラインがインバリデートされます。

3.3.5.1 グローバル・コヒーレンス動作

グローバル・キャッシュ動作は、L1D キャッシュ全体で実行されます。サポートされているグローバル・コヒーレンス動作は、インバリデート、ライトバック、およびライトバック・インバリデートの 3 つです。

プログラムで、グローバル・インバリデート動作を開始するために、L1DINV レジスタの I ビットに 1 をライトする必要があります。

グローバル・インバリデート動作が完了すると、L1DINV レジスタの I ビットは 0 にリセットされます。プログラムでこのビットをポーリングすると、動作の完了を検出できます。

プログラムで、グローバル・ライトバック動作を開始するために、L1DWB レジスタの C ビットに 1 をライトする必要があります。

完了時に、L1DWB レジスタの C ビットは 0 にリセットされます。プログラムで、このフィールドをポーリングすると、動作の完了を検出できます。

ライトバック・インバリデート動作も同様に制御されます。プログラムで、グローバル・ライトバック・インバリデート動作を開始するために、L1DWBINV レジスタの C ビットに 1 をライトする必要があります。

表 3-4 に、L1P グローバル・コヒーレンス動作の概要を示します。

表 3-4. グローバル・コヒーレンス動作

キャッシュ動作	使用レジスタ	L1D への影響
L1D ライトバック	L1DWB	すべての更新されたデータは L2 / 外部メモリへライトバックされますが、L1D では有効なままです。
L1D ライトバック・インバリデート	L1DWBINV	すべての更新されたデータは L2 / 外部メモリへライトバックされます。すべてのラインは、L1D でインバリデートされます。
L1D インバリデート	L1DINV	すべてのラインは、L1D でインバリデートされます。更新されたデータは、ドロップされます。

注意

L1D グローバル・インバリデートによって、L1D の更新されたデータはすべて低レベル・メモリへライトバックされるのではなく、廃棄されます。これにより、更新は低レベル・メモリへライトされると考えられているプログラムで、不正な動作が行われる場合があります。したがって、ほとんどのプログラムでは、L1D グローバル・インバリデートではなく、L1D ブロック・ライトバック・インバリデート（3.3.5.2 項 を参照）またはグローバル L2 動作を使用してください。

また旧製品との互換性を確保するために、L2CFG レジスタの ID ビットを 1 にセットすると、L1D キャッシュをグローバルにインバリデートすることもできます。ID フィールドは、C64x デバイスとの下位互換性を確保するために用意されています。新規に開発するアプリケーションではこのフィールドを使用するのではなく、L1DINV レジスタを使用してください。

3.3.5.2 ブロック・コヒーレンス動作

ブロック・コヒーレンス動作は、グローバル・コヒーレンス動作と機能的にはほぼ同じです。ただし、指定されたデータ・ブロックにのみ適用される点が異なります。このブロックは、対応するメモリ・マップド・レジスタのベース・アドレスおよびワード・サイズ（32 ビット）でそれぞれ指定されます。

ブロック・コヒーレンス動作は、CPU 上で同時に動作しているタスクに与える影響を最小限に抑制しながら、できるだけ効率的に動作するように設計されています。ブロック・キャッシュ動作は、通常、CPU が動作している「バックグラウンド」で動作しています。

サポートされているブロック・コヒーレンス動作は、インバリデート、ライトバック、およびライトバック・インバリデートの 3 つです。各動作には、それぞれに対応する 2 つのレジスタがあります。L1DXXBAR レジスタはブロックのベース・アドレスを指定し、L1DXXWC レジスタはブロックのワード・サイズを指定しています。

ゼロ以外の値を L1DXXWC レジスタのワード・カウント・フィールドにライトすると、ブロック・コヒーレンス動作が開始されます。

ワード・カウント・フィールドは、動作中にブロック・コヒーレンスが完了するとデクリメントし、ブロック・コヒーレンス動作で 0 にセットされます。プログラムで、このビットをポーリングすると、動作の完了を検出できます。

表 3-5 に、L1D ブロック・キャッシュ・コヒーレンス動作の概要を示します。

表 3-5. ブロック・キャッシュ・コヒーレンス動作

キャッシュ動作	使用レジスタ	L1D への影響
L1D ライトバック	L1DWBAR L1DWWC	更新されたデータは L2 / 外部メモリへライトされますが、L1D では有効なままです。
L1D ライトバック・インバリデート	L1DWIBAR L1DWIWC	更新されたデータは L2 / 外部メモリへライトバックされます。範囲内のすべてのラインは、L1D でインバリデートされます。
L1D インバリデート	L1DIBAR L1DIWC	範囲内のすべてのラインは、L1D でインバリデートされます。更新されたデータは、ドロップされます。

注 : ブロック・キャッシュ動作が進行中に、ブロック内のアドレス間でリード / ライトを行っても、そのようなアドレスでは要求したようにはライトバックもインバリデートも行われない場合があります。この問題を回避するためには、ブロック・キャッシュ動作の影響を受けるキャッシュ・ラインの範囲内のアドレスを、動作が進行中にアクセスしないようにプログラムしてください。適切なワード・カウント・フィールドをポーリングして、ブロック動作が完了したタイミングを判別するようにプログラムしてください。

同一バンクに対して 2 つの同時アクセスを行うと、1 サイクル・ストールのペナルティが発生します。ただし、次の特殊な場合を除きます。

- メモリ・アクセスは、同一ワード内のオーバーラップしないバイトへの 2 つのライトです。したがって、このアドレスのビット 31 ～ 2 は同じものになります。
- メモリ・アクセスは、同一ワード内のすべてまたは一部をアクセスする 2 つのリードです。したがって、このアドレスのビット 31 ～ 2 は同じものになります。この場合、2 つのアクセスはオーバーラップすることがあります。
- メモリ・アクセスの一方または両方とも、L1D をミスしたライトで、その代わりにライト・バッファで処理されます（ライト・バッファの詳細については、3.5.3 項を参照してください）。
- メモリ・アクセスは、アラインされない単一アクセスになります。アラインされないアクセスは、メモリ・システムがそのアクセスを複数に分割したとしても、バンク競合ストールを引き起こすことはありません。

同一バンクへのリード・アクセスとライト・アクセスを同時に行うと、常にストールが引き起こされることに注意してください。同一バンクへのリード・アクセスとライト・アクセスをそれぞれ 2 回行っても、上記条件を満たす限り、ストールしない場合があります。

CPU と DMA/IDMA が独立した L1D メモリ・バンクへ同時にアクセスしてもストールすることはありません。同一バンクへアクセスすると、CPU と DMA/IDMA 間で競合を引き起こします。CPU と DMA/IDMA のいずれかは、ルールに基づいてストールします（第 6 章を参照）。

3.3.6 キャッシュ・コヒーレンス・プロトコル

C64x+ L1D キャッシュは、L2 RAM の DMA 動作に関連してコヒーレントを保ちます。このパラダイムをサポートするために、L1D キャッシュは L2 から届くキャッシュ・コヒーレンス・コマンドを受け入れます。

3.3.6.1 L2 から L1D へのキャッシュ・コヒーレンス・プロトコル

L2 RAM 内の DMA/IDMA トラフィックに関連して L1D キャッシュ・コヒーレンスをサポートするために、L1D コントローラは L2 から届く、スヌープ・リード（SNPR）およびスヌープ・ライト（SNPW）の 2 つのキャッシュ・コヒーレンス・コマンドをサポートします。L2 は、L2 RAM 内の DMA / IDMA 動作に応答し、必要に応じてこれらのスヌープ・コマンドを送ります。

スヌープ・リード・コマンドが L1D に送られるのは、L2 で L1D キャッシュがリクエストされたラインを保持したことを検出し、かつそのラインがダーティであることを検出した場合です。L1D はリクエストされたデータを戻して、応答します。

スヌープ・ライト・コマンドが L1D に送られるのは、L2 で L1D キャッシュがリクエストされたラインを保持したことを検出した場合です。このラインが L1D で変更されたかどうかは問題にはなりません。L1D はその内容に従って更新します。

3.3.6.2 L1D から L2 へのキャッシュ・コヒーレンス・プロトコル

L1D への過剰なスヌープ・トラフィックを削減するために、L2 はスヌープをフィルタリングします。その結果、不必要なスヌープは L1D へは送られないようになります。

L2 は、L1D のタグ・メモリのシャドウ・コピーを保持しています。L2 は、L1D タグのローカル・コピーを調べ、L1D へ送るスヌープ・コマンドが保証されているかどうかを決定します。

L2 は、最初、L1D のリード・ミス・リクエストに応答してシャドウ・タグを更新します。次に、L1D ビクティム・ライトバックに応答してシャドウ・タグを更新します。L1D がリード・リクエストを発行する場合、ラインが L1D でアロケートされているかどうかも示します。L1D でアロケートされている場合には、ラインがアロケートされているセット内におけるロケーション（ウェイ）の状況を示します。L2 は、この情報に基づいてシャドウ・タグの対応するセットを更新できます。

L2 は、L1D キャッシュに存在するアドレスを記録するだけでなく、これらのラインが C64x+ でダーティであるかどうかも記録します。

3.4 L1D キャッシュ・コントロール・レジスタ

3.4.1 メモリ・マップド L1D キャッシュ・コントロール・レジスタの概要

C64x+ メガモジュールのメモリ・システムには、L1D キャッシュの動作を管理するレジスタ・セットがあります。これらのレジスタを使用すると、キャッシュ・モードの変更が可能になり、キャッシュ・コヒーレンス動作を手動で開始できます。

表 3-6 に、L1D キャッシュ・コントロール・レジスタを示します。

表 3-6. L1D 固有のキャッシュ・コントロール動作を行うレジスタ

動作タイプ	レジスタ名	アドレス	動作	参照先
モードの選択	L1DCFG	0184 0040h	L1D のキャッシュ・サイズを設定します。	3.4.3.1 項
	L1DCC	0184 0044h	L1D の動作モードを制御します（フリーズ / 正常）。	3.4.3.2 項
ブロック・キャッシュ動作	L1DWIBAR	0184 4030h	指定範囲は L1D でライトバックされ、インバリデートされます。	3.4.4.2 項
	L1DWIWC	0184 4034h		
	L1DWBAR	0184 4040h	指定範囲は L1D からライトバックされ、有効なままです。	
	L1DWWC	0184 4044h		
	L1DIBAR	0184 4048h	指定範囲はライトバックされずに L1D でインバリデートされます。	
	L1DIWC	0184 404Ch		
グローバル・キャッシュ動作	L1DWB	0184 5040h	L1D の内容全体がライトバックされますが、有効なままです。	3.4.4.1 項
	L1DWBINV	0184 5044h	L1D の内容全体がライトバックされ、インバリデートされます。	
	L1DINV	0184 5048h または CCFG の ID ビット	L1P の内容全体はライトバックされずにインバリデートされます。	

前述の L1D 固有のレジスタだけでなく、L1D キャッシュも同様に L2 固有のコントロールへライトすると直接影響を受けます。キャッシュ・コントロール動作および L1D キャッシュに与える影響の詳細については、第 4 章を参照してください。

3.4.2 CPU L1D キャッシュ・コントロール・レジスタ

CPU には内部コントロール・レジスタである、コントロール・ステート・レジスタ（CSR）が 1 つあります。このレジスタにはキャッシュ・コントロール動作専用のフィールド（CSR レジスタの PCC フィールド）が用意されています。DCC フィールドは、以前のデバイスではさまざまな点で L1D の動作を制御していました。

現在、CSR レジスタは、C64x+ では L1D の動作を制御していません。L1D は DCC フィールドに保持されている値を無視します。DCC フィールドの以前の動作は、現在、L1DCC および L1DCFG レジスタの一部になっています（3.3.4 項を参照）。

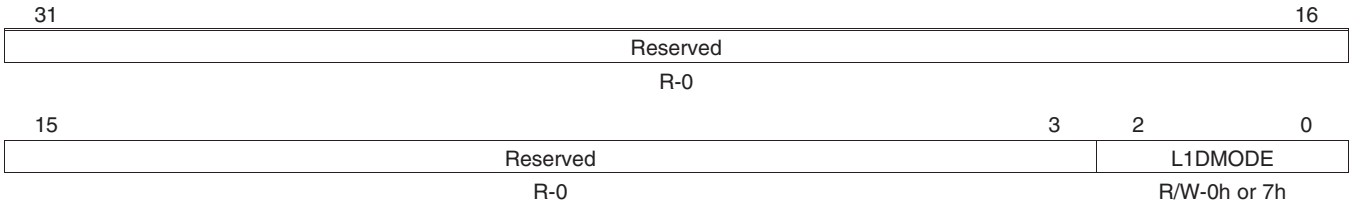
3.4.3 L1D キャッシュ・コンフィギュレーション・レジスタ

L1DCFG および L1DCC レジスタは、L1D の動作を制御します。

3.4.3.1 L1D キャッシュ・コンフィギュレーション・レジスタ（L1DCFG）

L1D キャッシュ・コンフィギュレーション・レジスタ（L1DCFG）は、L1D のキャッシュ・サイズを制御します。L1DCFG を図 3-2 に示し、表 3-7 で説明します。

図 3-2. L1D キャッシュ・コンフィギュレーション・レジスタ（L1DCFG）



凡例：R/W = リード / ライト。R = リード専用。*-n* = リセット後の値。

表 3-7. L1D キャッシュ・コンフィギュレーション・レジスタ（L1DCFG）フィールドの説明

ビット	フィールド	値	説明
31-3	Reserved	0	予約。
2-0	L1DMODE	0 ～ 7h	L1D のキャッシュ・サイズを指定します。L1DMODE フィールドのデフォルトは、0h または 7h です。詳細については、各デバイスのデータ・マニュアルを参照してください。
		0h	L1D キャッシュはディスエーブル。
		1h	4K
		2h	8K
		3h	16K
		4h	32K
		5h	最大のキャッシュ・サイズ。
		6h	最大のキャッシュ・サイズ。
		7h	最大のキャッシュ・サイズ。

3.4.3.2 L1D キャッシュ・コントロール・レジスタ（L1DCC）

L1D キャッシュ・コントロール・レジスタ（L1DCC）は、L1D がフリーズされているか、フリーズ解除されているかを制御します。L1DCC を図 3-3 に示し、表 3-8 で説明します。

図 3-3. L1D キャッシュ・コントロール・レジスタ（L1DCC）

31	17	16
Reserved		POPER
R-0		R-0

15	1	0
Reserved		OPER
R-0		R/W-0

凡例：R/W = リード / ライト。R = リード専用。-*n* = リセット後の値。

表 3-8. L1D キャッシュ・コントロール・レジスタ（L1DCC）フィールドの説明

ビット	フィールド	値	説明
31-17	Reserved	0	予約。
16	POPER	0-1	OPER フィールドの 1 つ前の値を保持します。
15-1	Reserved	0	予約。
0	OPER		L1D フリーズ・モードを制御します。
		0	フリーズ・モードはディスエーブル。
		1	フリーズ・モードはイネーブル。

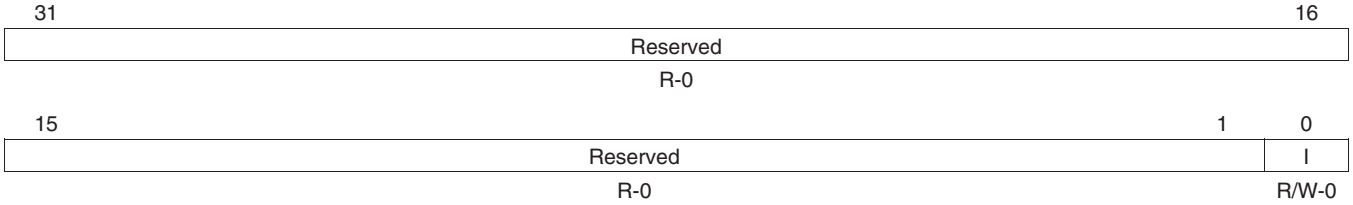
3.4.4 L1D キャッシュ・コヒーレンス・オペレーション・レジスタ

3.4.4.1 グローバル・コヒーレンス・オペレーション・レジスタ

3.4.4.1.1 L1D インバリデート・レジスタ（L1DINV）

L1D インバリデート・レジスタ（L1DINV）は、L1D キャッシュのグローバル・インバリデートを制御します。L1DINV を図 3-4 に示し、表 3-9 で説明します。

図 3-4. L1D インバリデート・レジスタ（L1DINV）



凡例：R/W = リード / ライト。R = リード専用。-*n* = リセット後の値。

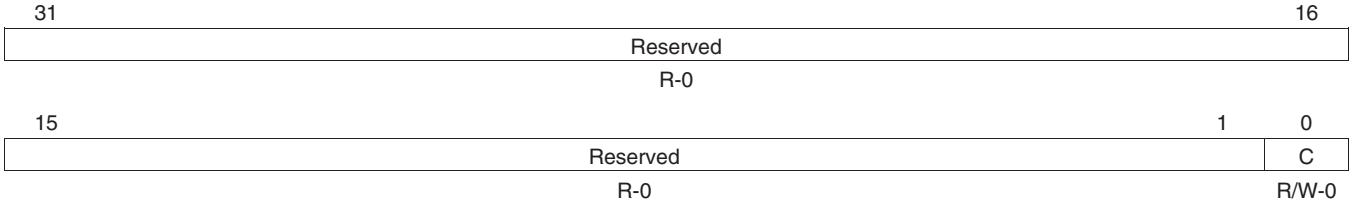
表 3-9. L1D インバリデート・レジスタ（L1DINV）フィールドの説明

ビット	フィールド	値	説明
31-1	Reserved	0	予約。
0	I		L1D キャッシュのグローバル・インバリデートを制御します。
		0	通常動作。
		1	すべての L1D キャッシュ・ラインはインバリデートされます。

3.4.4.1.2 L1D ライトバック・レジスタ（L1DWB）

L1D ライトバック・レジスタ（L1DWB）を図 3-5 に示し、表 3-10 で説明します。

図 3-5. L1D ライトバック・レジスタ（L1DWB）



凡例：R/W = リード / ライト。R = リード専用。-*n* = リセット後の値。

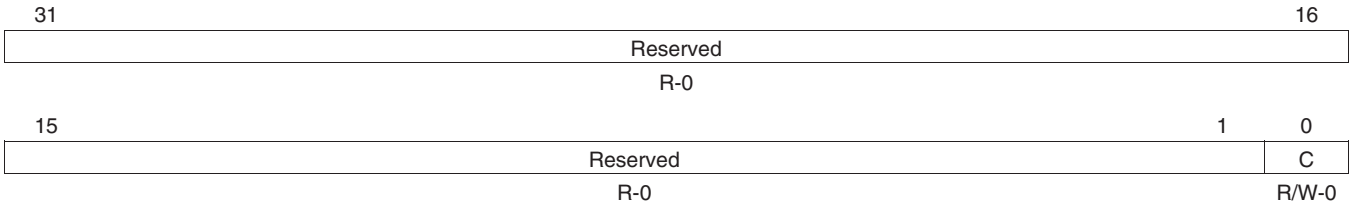
表 3-10. L1D ライトバック・レジスタ（L1DWB）フィールドの説明

ビット	フィールド	値	説明
31-1	Reserved	0	Reserved
0	C		L1D キャッシュのグローバル・ライトバック動作を制御します。
		0	通常の L1D 動作。
		1	ダーティ L1D ラインがライトバックされます。

3.4.4.1.3 L1D ライトバック・インバリデート・レジスタ（L1DWBINV）

L1D ライトバック・インバリデート・レジスタ（L1DWBINV）は、L1D キャッシュのグローバル・ライトバック・インバリデート動作を制御します。L1DWBINV を図 3-6 に示し、表 3-11 で説明します。

図 3-6. L1D ライトバック・インバリデート・レジスタ（L1DWBINV）



凡例：R/W = リード / ライト。R = リード専用。-*n* = リセット後の値。

表 3-11. L1D ライトバック・インバリデート・レジスタ（L1DWBINV）フィールドの説明

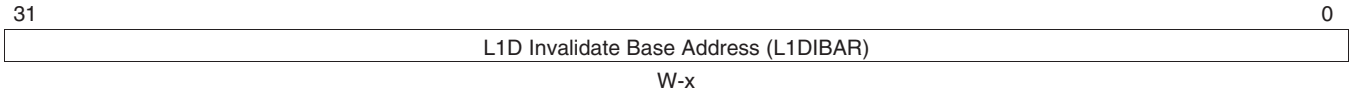
ビット	フィールド	値	説明
31-1	Reserved	0	予約。
0	C		L1D キャッシュのグローバル・ライトバック・インバリデート動作を制御します。
		0	通常の L1D 動作。
		1	ダーティ L1D ラインをライトバックし、すべての L1D ラインをインバリデートします。

3.4.4.2 ブロック・コヒーレンス・オペレーション・レジスタ

3.4.4.2.1 L1D インバリデート・ベース・アドレス・レジスタ（L1DIBAR）

L1D インバリデート・ベース・アドレス・レジスタ（L1DIBAR）は、インバリデートされるブロックのベース・アドレスを指定します。L1DIBAR を図 3-7 に示し、表 3-12 で説明します。

図 3-7. L1D インバリデート・ベース・アドレス・レジスタ（L1DIBAR）



凡例：W = ライト専用。*-x* = 値は不定。詳細については、各デバイスのデータ・マニュアルを参照してください。

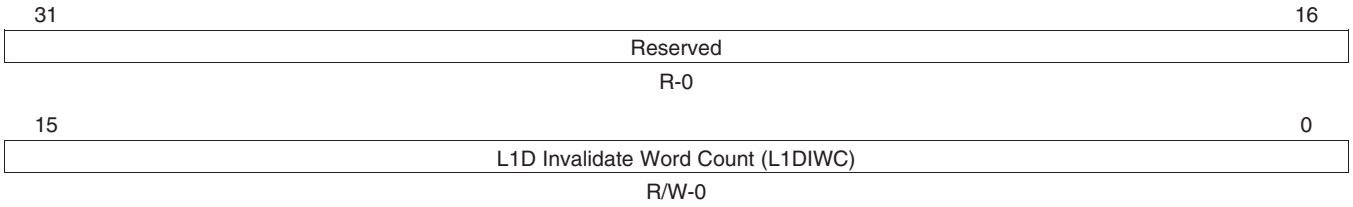
表 3-12. L1D インバリデート・ベース・アドレス・レジスタ（L1DIBAR）フィールドの説明

ビット	フィールド	値	説明
31-0	L1DIBAR	0 ～ FFFF FFFFh	L1D ブロック・インバリデート動作を行うベース・アドレスを指定します。

3.4.4.2.2 L1D インバリデート・ワード・カウント・レジスタ（L1DIWC）

L1D インバリデート・ワード・カウント・レジスタ（L1DIWC）は、インバリデートされるブロック・サイズを指定します。サイズは、32 ビット・ワードで指定されます。L1DIWC を図 3-8 に示し、表 3-13 で説明します。

図 3-8. L1D インバリデート・ワード・カウント・レジスタ（L1DIWC）



凡例：R/W = リード / ライト。R = リード専用。*-n* = リセット後の値。

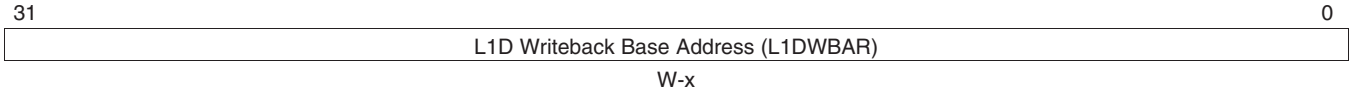
表 3-13. L1D インバリデート・ワード・カウント・レジスタ（L1DIWC）フィールドの説明

ビット	フィールド	値	説明
31-16	Reserved	0	予約。
15-0	L1DIWC	0 ～ FFFFh	ブロック・インバリデートを行うワード・カウント。

3.4.4.2.3 L1D ライトバック・ベース・アドレス・レジスタ（L1DWBAR）

L1D ライトバック・ベース・アドレス・レジスタ（L1DWBAR）は、ライトバックされるブロックのベース・アドレスを指定します。L1DWBAR を図 3-9 に示し、表 3-14 で説明します。

図 3-9. L1D ライトバック・ベース・アドレス・レジスタ（L1DWBAR）



凡例：W = ライト専用。*-x* = 値は不定。詳細については、各デバイスのデータ・マニュアルを参照してください。

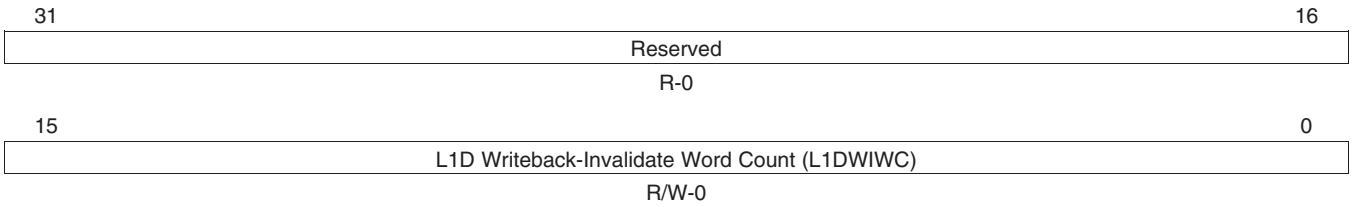
表 3-14. L1D ライトバック・ベース・アドレス・レジスタ（L1DWBAR）フィールドの説明

ビット	フィールド	値	説明
31-0	L1DWBAR	0 ～ FFFF FFFFh	L1D ブロック・ライトバック動作を行うベース・アドレスを指定します。

3.4.4.3 L1D ライトバック・インバリデート・ワード・カウント・レジスタ（L1DWIWC）

L1D ライトバック・インバリデート・ワード・カウント・レジスタ（L1DWIWC）は、インバリデートされるブロック・サイズを指定します。サイズは、32 ビット・ワードで指定されます。L1DWIWC を図 3-10 に示し、表 3-15 で説明します。

図 3-10. L1D ライトバック・インバリデート・ワード・カウント・レジスタ（L1DWIWC）



凡例：R/W = リード / ライト。R = リード専用。*-n* = リセット後の値。

表 3-15. L1D ライトバック・インバリデート・ワード・カウント・レジスタ（L1DWIWC）フィールドの説明

ビット	フィールド	値	説明
31-16	Reserved	0	予約。
15-0	L1DWIWC	0 ～ FFFFh	ブロック・インバリデートを行うワード・カウント。

3.4.5 権限およびキャッシュ・コントロール動作

キャッシュ・コントロール動作が受ける権限の影響は、次のように要約できます。

- スーパーバイザ・コードは、L1D のキャッシュ・サイズを変更できる
- ユーザ・モード・コードは、L1D のキャッシュ・サイズを変更できない
- スーパーバイザ・コードだけが、L1D に対してグローバル・インバリデートを発行できる
- スーパーバイザおよびユーザ・モードは両方ともいつでも L1D のフリーズまたはフリーズ解除できる

表 3-16 に、役割ごとにアクセスできる L1D キャッシュ・コントロール・レジスタおよびメガモジュールで行われる保護チェックの内容について概要を示します。

表 3-16. L1D キャッシュ・コントロール・レジスタのアクセス権限

レジスタ	スーパーバイザ	ユーザ
L1DCFG	R/W	R
L1DCC	R/W	R/W
L1DWIBAR	W	W
L1DWIWC	R/W	R/W
L1DWBAR	W	W
L1DWWC	R/W	R/W
L1DIBAR	W	W
L1DIWC	R/W	R/W
L1DWB	R/W	R/W
L1DWBINV	R/W	R/W
L1DINV	R/W	R

3.5 L1D メモリ・パフォーマンス

L1D メモリ・パフォーマンスは、いくつかの要因によって異なります。ここでは、L1D メモリ・パフォーマンスに関するバンキング・アーキテクチャ、ライト・バッファ、およびミス・パイプライン化の影響について説明します。

3.5.1 L1D メモリ・バンキング

L1D は、最下位ビット（LSB）ベースのメモリ・バンキング構成を備え、メモリを 8 個の 32 ビット幅のバンクに分割します。これらのバンクはシングル・ポートで、サイクルごとに 1 アクセスのみが可能です。L1D RAM と L1D キャッシュは両方とも、同一バンク構成を共有します。

バンクは、アドレスの下位ビットに基づき、インターリーブされます。具体的には、アラインされたメモリ・アクセスの場合、アドレス・ビット [4:2] がバンク番号を決定します。ビットのバンク番号へのマッピングは、デバイスのエンディアン・モードによって異なります。

図 3-11 では、このアドレスのビット 4 ～ 2 はバンクを、ビット 1 ～ 0 はバンク内のバイトをそれぞれ選択しています。

図 3-11. バンク番号へのアドレスのマッピング

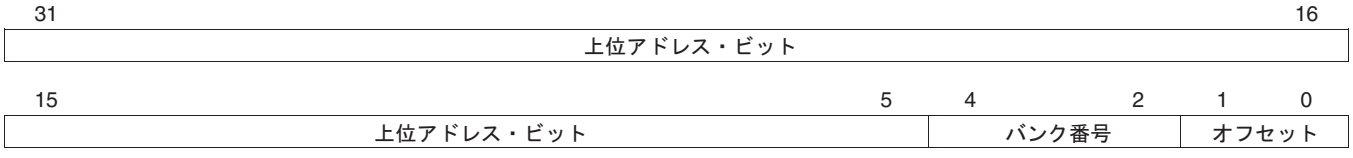


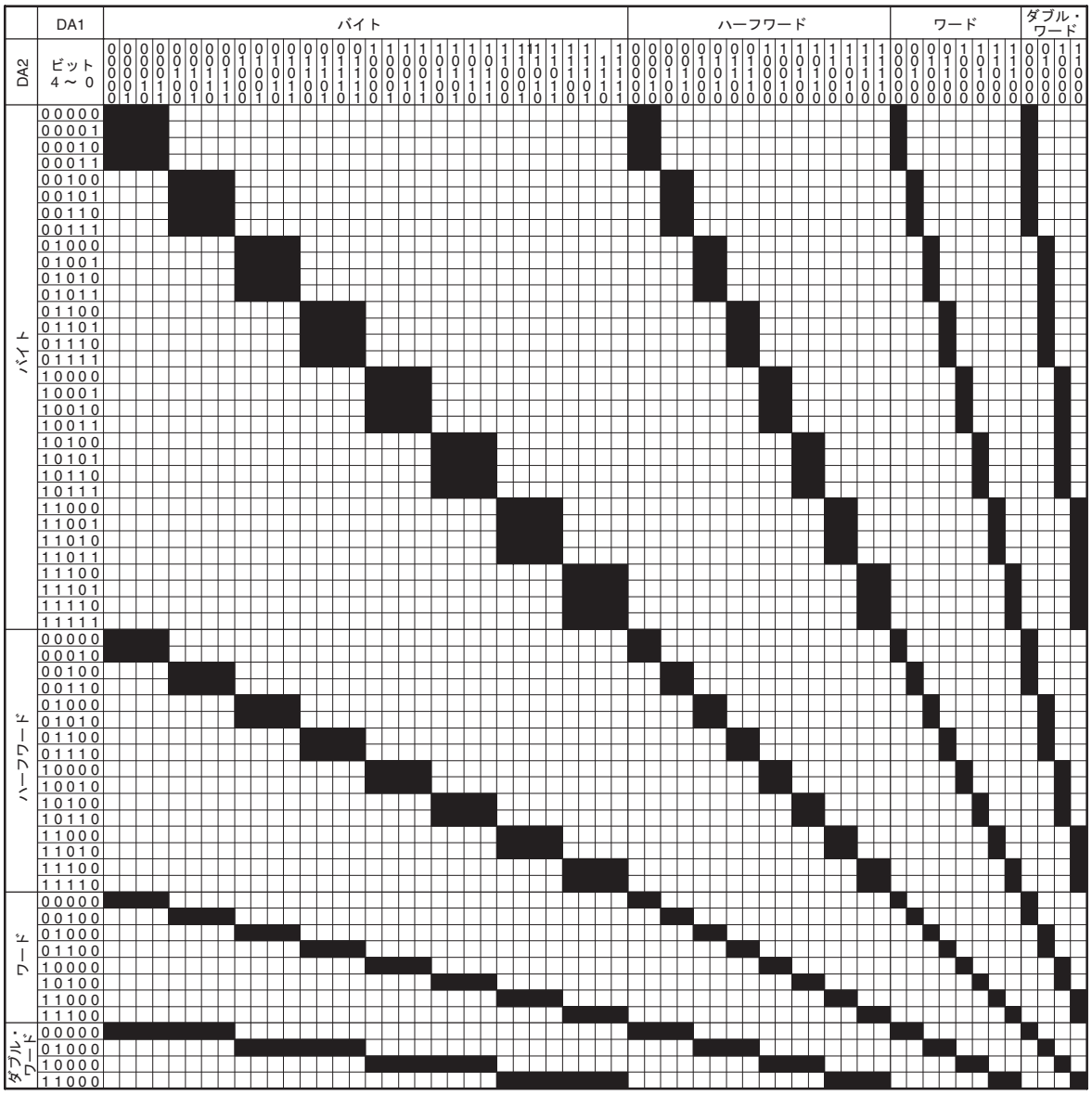
図 3-12 の陰影を付けた領域は、バンク競合ストールを引き起こす可能性がある同時アクセスにおけるアドレスの LSB の組み合わせを示しています。同一バンクに対する 2 つの同時アクセスを行うと、1 サイクル・ストールのペナルティが発生します。ただし、次の特殊な場合を除きます。

- メモリ・アクセスは、同一ワード内のオーバーラップしないバイトへの 2 つのライトです。そのため、このアドレスのビット 2 ～ 31 は同じものになります。
- メモリ・アクセスは、同一ワード内のすべてまたは一部をアクセスする 2 つのリードです。そのため、このアドレスのビット 2 ～ 31 は同じものになります。この場合、2 つのアクセスはオーバーラップすることがあります。
- メモリ・アクセスは、1 つのアラインされないアクセスになります。アラインされないアクセスは、メモリ・システムがそのアクセスを複数に分割したとしても、バンク競合ストールを引き起こすことはありません。

同一バンクへのリード・アクセスとライト・アクセスを同時に行うと、常にストールが引き起こされることに注意してください。同一バンクへのリード・アクセスとライト・アクセスをそれぞれ 2 回行っても、上記条件を満たす限り、ストールしない場合があります。

CPU と DMA/IDMA が同時に独立した L1D メモリ・バンクへアクセスしてもストールすることはありません。同一バンクへアクセスすると、CPU と DMA/IDMA 間で競合を引き起こします。CPU と DMA/IDMA のどれか 1 つは、ルールに基づいてストールします（第 6 章を参照）。

図 3-12. 潜在的に競合するメモリ・アクセス



3.5.2 L1D ミス・ペナルティ

L1D はサイクルごとに、CPU からのデータ・アクセス・リクエストを 2 つまで処理できます。バンク競合が起こらない限り、L1D にヒットするアクセスはストールなしで完了します（3.3.6.1 項を を参照）。

L1D でミスしたリードは、リクエストされたデータがフェッチされている間、CPU をストールさせます。L1D は、リード・アロケート・キャッシュです。したがって、リクエストされたデータのために新しいラインがアロケートされます。

L2 でもミスになる L1D リード・ミスは、L2 が外部メモリからそのデータを取り出す間、CPU をストールさせます。データが取り出されると、L2 にストアされ、L1D に転送されます。外部ミス・ペナルティは、システム負荷に関するその他の状況や、外部データを保持するために使用される外部メモリのタイプや幅によって変化します。

同一サイクルの同一ラインで 2 つのリード・ミスがある場合、1 つのミス・ペナルティのみが課せられます。同様に、同一ラインに 2 回のアクセスが連続してあり、最初のアクセスがミスの場合、2 番目のアクセスに対して追加のミス・ペナルティは何ら与えられません。

L1D 内でラインをアロケートするプロセスは、ビクティム・ライトバックを引き起こします。ビクティム・ライトバックは、L1D の更新データを下位レベルのメモリに移動します。更新データが L1D から追い出されるとき、キャッシュはそのデータをビクティム・バッファに移動します。データがビクティム・バッファに移動されると、L1D は現行のリード・ミス処理を再開します。ビクティム・ライトバックの続きの処理は、バックグラウンドで行われます。しかし、続いて起こるリード・ミスやライト・ミスは、ビクティム・ライトバックが処理される間、待機しなくてはなりません。リード・ミスが既存のビクティムと競合しない場合、リード・ミスは性能上のペナルティを削減するために、ビクティム・ライトバックを使ってパイプライン化されます。

L1D は、リード・ミスをパイプライン化します。異なるラインに対する連続したリード・ミスはオーバーラップされることがあり、全体的なストール・ペナルティが軽減されます。

ライト・ミスは、直接 CPU をストールさせることはありません。その代わり、ライト・ミスは L1D と L2 間にあるライト・バッファのキューに積まれます。CPU はライト・ミスによりストールしませんが、ライト・バッファはさまざまな状況下で CPU をストールさせます。ライト・バッファの影響については、3.5.3 項を参照してください。

3.5.3 L1D ライト・バッファ

L1D は、ライト・アロケートを行いません。その代わり、ライト・ミスは、L1D でのラインのアロケートなしに L2 に直接渡されます。これらのライト・ミスを取り込むために、4 エントリの 128 ビットの幅をもつライト・バッファが L1D キャッシュと L2 メモリ間に存在します。ライト・バッファは L1D から L2 へのライト用に、128 ビットのパスを用意し、4 つの未処理のライト・リクエストを格納する空きを備えています。

ライト・バッファに空きがある場合、L1D をミスしたライトは CPU をストールさせることはありません。ライト・バッファがフルの場合、バッファにライトのための空きができるまで、ライト・ミスは間接的に CPU をストールさせます。またライト・バッファは、リード・ミスの時間が延長されることで、間接的に CPU をストールすることもあります。L1D をミスしたリードは、ライト・バッファがエンプティにならない間、処理されません。ライト・バッファがエンプティになると、リード・ミスが処理されます。これは、リード・ミスのアドレスがライト・バッファ内でペンディングになっている、ライトのアドレスとオーバーラップする場合があるので必要です。

L2 は、リクエストされた L2 バンクがビジーでなければ、L2 サイクル（L2 サイクル = 2 x CPU サイクル）ごとに、ライト・バッファからの新しいリクエストを処理することができます。メモリ内で複数のエレメントが連続している場合、単一メモリ・アクセスを行うために、バッファ内にそのエレメントを一緒にマージすることができます。

ライト・バッファは、ライト・リクエストをマージすることできます。また、ライト・リクエストが次のルールに従う場合は、2 つのライト・ミスを単一のトランザクションにマージします。

- 新たなライト・ミスが直前のライト・ミスとして、同一の 128 ビット・クワッドワード内に存在する。
- 2 つのライトが L2 SRAM のロケーション（L2 キャッシュ内に保持されている可能性のあるロケーションではない）に対するものである。
- ちょうど最初のライトがライト・バッファのキューに積まれたところである。
- 現在、2 番目のライトがバッファのキューに積まれようとしている。
- 最初のライトが未だ L2 コントローラに渡されていない。
- 2 つのライトが同一権限レベルをもっている。

前述の条件は、プログラムが大規模で連続的なライトを行うとき、またはメモリ内の構造体に小規模のライトをバーストで行うときのような多くの状況で発生します。このような場合、ライト・マージングは、ライト・バッファ内にある独立したストアの数を減少させることで、ライト・バッファの実質的な容量を増加させます。また、多数のライト・ミスのあるプログラムに対するストール・ペナルティを軽減します。

二次的な効果として、ライト・マージングは、L2 で実行されるメモリ操作の数を減らします。これは、L2 が処理する必要のある個々のライト操作の合計数を減らすことで、L2 メモリの全体的なパフォーマンスを向上させます。アクセス先が、隣接している場合、L2 バンクに複数回アクセスをするのではなく、複数のアクセスを 1 回にまとめます。これにより、他のリクエスタがそのバンクをより迅速にアクセスすることが可能になり、また、CPU が次のサイクルで直ちに次のバンクに移動できるようになります。

3.5.4 L1D ミス・パイプライン化

L1D キャッシュは、リード・ミスをパイプライン化します。ミス・パイプライン化は、いくつかのキャッシュ・ミスの処理をオーバーラップさせることで、このオーバーヘッドの多くを見かけ上なくすことができます。

表 3-17 に、L1D パフォーマンスの概要を示します。2 種類の異なる設定が存在します。最初の設定の特長は、L2SRAM で、2 x 128 ビット・バンクの場合 0 ウェイト・ステートということです。この設定は、DM644x デバイスで利用できます。2 番目の設定の特長は、L2SRAM で、4 x 128 ビット・バンクの場合 1 ウェイト・ステートということです。この設定は、C645x デバイスで利用できます。

表 3-17. L1D パフォーマンスの概要

L2 のタイプ	0 ウェイト・ステート、2 x 128 ビット・バンク		1 ウェイト・ステート、4 x 128 ビット・バンク	
パラメータ	L2 SRAM	L2 キャッシュ	L2 SRAM	L2 キャッシュ
単一のリード・ミス	10.5	12.5	12.5	14.5
2 つの同時リード・ミス（パイプライン化）	10.5 + 4	12.5 + 8	12.5 + 4	14.5 + 8
M 個の連続リード・ミス（パイプライン化）	10.5 + 3 × (-1)	12.5 + 7 × (-1)	12.5 + 3 × (-1)	14.5 + 7 × (-1)
M 個の連続同時リード・ミス（パイプライン化）	10.5 + 4 × (M/2-1) + 3 × M/2	12.5 + 8 × (M/2-1) + 7 × M/2	12.5 + 4 × (-1)	14.5 + 8 × (M/2-1) + 7 × M/2

3.6 L1D パワーダウン・サポート

CPU がアイドル状態のとき、L1D メモリをパワーダウン・モードにセットすることができます。C64x+ メガモジュールをパワーダウンさせるのに必要なソフトウェア・シーケンスは、次のとおりです。

1. PDCCMD レジスタの MEGPD フィールドを 1 にセットして、パワーダウンをイネーブルします。
2. メガモジュールをウェイクアップする CPU 割り込みをイネーブルします。他の割り込みをすべてディスエーブルします。
3. IDLE 命令を実行します。

メガモジュールは、上記ステップ 2 でイネーブルされた割り込みが発生するまでパワーダウン・モードになったままです。

メガモジュールがパワーダウンしている間に、DMA アクセスが L1D、L1P、L2 のいずれかのメモリに対して行われた場合、パワーダウン・コントローラ（PDC）は 3 つのメモリ・コントローラをウェイクアップします。PDC は DMA アクセスが処理されると、メモリ・コントローラを再度パワーダウンします。

C64x+ メガモジュールの PDCCMD レジスタおよびパワーダウン機能の詳細については、第 9 章を参照してください。

注： ここで説明したように、メガモジュールをパワーダウンすることは、多くの場合、静的パワーダウンといいます。この用語は、このモードを説明するために使われます。これはこの機能が長時間にわたり使われていることが多いためです。

3.7 L1D メモリ保護

L1D メモリはメモリ保護をサポートし、多数のシステムで求められる堅牢性をもたらします。メモリ保護を行うレベルがいくつか用意されています。すべてのレベルがすべてのデバイスで利用できるとは限りません。詳細については、各デバイスのデータ・マニュアルを参照してください。理解が十分ではない場合には、第 8 章を参照して理解を深めてから、ここの説明をお読みください。

3.7.1 L1D アクセス時に行われる保護チェック

3.7.1.1 CPU、IDMA、および他のシステム・マスタからのアクセス時に行われる保護チェック

保護チェックは、メモリ保護サポートをはじめとする、デバイスに搭載されている L1D によって直接処理されるすべてのアクセスに対して行われます。これには、CPU、IDMA、および他のシステム・マスタからのアクセスも含まれます。

L2 コントローラでは、CPU による特定のリクエストを許可または拒否するかを決定します。これはそのリクエストに対応する権限レベル、およびそのリクエストによってアクセスされるアドレス範囲でのアクセス権限の設定に基づきます。これらのチェックに関する正しいルールは、第 8 章で説明しています。

L1D メモリ・コントローラには、C64x+ 割り込みコントローラへ送る 2 つの例外出力を備えているという特長があります。この例外出力の 1 つは、CPU によってトリガされる（「ローカルな」）メモリ例外が発生したことを示します。もう一方は、システム・マスタによってトリガされる（「リモートの」）例外が発生したことを示します。

3.7.1.2 プログラムによって開始されるキャッシュ・コヒーレンス動作時に行われる他の保護チェック

保護チェックは、メモリ保護の整合性を保証するために、プログラムによって開始されるキャッシュ・コヒーレンス動作時に行われます。ユーザとスーパーバイザ・コードは両方とも手動でキャッシュ・コヒーレンス動作を発行できます。

ただし、ユーザ・コードは L1D キャッシュをグローバルにインバリデートできません。また L1D のキャッシュ・サイズも変更できません。スーパーバイザ・コードだけが、グローバルにインバリデートを開始したり、キャッシュにアロケートするメモリ量を変更することができます。

3.7.2 L1D メモリ・プロテクション・レジスタ

次のレジスタは、L1D メモリを保護する動作を管理しています。これらのレジスタは 3 つの主要なカテゴリに分類されます。

- メモリ・プロテクション・ページ・アトリビュート・レジスタ（MPPA）：これらのレジスタは、保護されたページごとに対応するアクセス権限を格納します。
- メモリ・プロテクション・ロック・レジスタ（MPLK）：ペリフェラルは、ハードウェアによるメモリ保護ロック機能を実現するために選択できます。ペリフェラルで他のリクエストを処理できない場合、ロックをかけてそのペリフェラルのメモリ保護エントリに対するすべての更新をディスエーブルします。
- メモリ・プロテクション・フォールト・レジスタ（MPF*x*R）：メモリ保護障害を生成する各ペリフェラルには、障害の詳細を記録するために MPFAR、MPFSR、MPFCR の各レジスタが用意されています。

表 3-18 に、L1D メモリ・プロテクション・ロック・レジスタを示します。これらのレジスタのメモリ・アドレスについては、各デバイスのデータ・マニュアルを参照してください。

表 3-18. L1D メモリ・プロテクション・ロック・レジスタ

アドレス	略称	レジスタの説明	参照先
0184 AExxh	L1DMPPAxx	L1D メモリ・プロテクション・ページ・アトリビュート・レジスタ	3.7.2.1 項
0184 AD00h	L1DMPLK0	L1D メモリ・プロテクション・ロック・レジスタ 0	3.7.2.2.1 項
0184 AD04h	L1DMPLK1	L1D メモリ・プロテクション・ロック・レジスタ 1	3.7.2.2.2 項
0184 AD08h	L1DMPLK2	L1D メモリ・プロテクション・ロック・レジスタ 2	3.7.2.2.3 項
0184 AD0Ch	L1DMPLK3	L1D メモリ・プロテクション・ロック・レジスタ 3	3.7.2.2.4 項
0184 AD10h	L1DMPLKCMD	L1D メモリ・プロテクション・ロック・コマンド・レジスタ	3.7.2.2.5 項
0184 AD14h	L1DMPLKSTAT	L1D メモリ・プロテクション・ロック・ステータス・レジスタ	3.7.2.2.6 項
0184 AC00h	L1DMPFAR	L1D メモリ・プロテクション・フォールト・アドレス・レジスタ	3.7.2.3.1 項
0184 AC04h	L1DMPFSR	L1D メモリ・プロテクション・フォールト・セット・レジスタ	3.7.2.3.2 項
0184 AC08h	L1DMPFCR	L1D メモリ・プロテクション・フォールト・クリア・レジスタ	3.7.2.3.3 項

3.7.2.1 L1D メモリ・プロテクション・アトリビュート・レジスタ

L1D は 32 ページのメモリを保護する機能を実装しています。L1DMPPA0 ～ L1DMPPA15 は領域 0 に対応し、L1DMPPA16 ～ L1DMPPA31 は領域 1 にそれぞれ対応します。

表 3-19. L1D メモリ・プロテクション・アトリビュート・レジスタのアドレス

L1D 領域 0		L1D 領域 1	
レジスタ	**アドレス**	**レジスタ**	**アドレス**
L1DMPPA0	0184 AE00h	L1DMPPA16	0184 AE40h
L1DMPPA1	0184 AE04h	L1DMPPA17	0184 AE44h
L1DMPPA2	0184 AE08h	L1DMPPA18	0184 AE48h
L1DMPPA3	0184 AE0Ch	L1DMPPA19	0184 AE4Ch
L1DMPPA4	0184 AE10h	L1DMPPA20	0184 AE50h
L1DMPPA5	0184 AE14h	L1DMPPA21	0184 AE54h
L1DMPPA6	0184 AE18h	L1DMPPA22	0184 AE58h
L1DMPPA7	0184 AE1Ch	L1DMPPA23	0184 AE5Ch
L1DMPPA8	0184 AE20h	L1DMPPA24	0184 AE60h
L1DMPPA9	0184 AE24h	L1DMPPA25	0184 AE64h
L1DMPPA10	0184 AE28h	L1DMPPA26	0184 AE68h
L1DMPPA11	0184 AE2Ch	L1DMPPA27	0184 AE6Ch
L1DMPPA12	0184 AE30h	L1DMPPA28	0184 AE70h
L1DMPPA13	0184 AE34h	L1DMPPA29	0184 AE74h
L1DMPPA14	0184 AE38h	L1DMPPA30	0184 AE78h
L1DMPPA15	0184 AE3Ch	L1DMPPA31	0184 AE7Ch

3.7.2.1.1 L1D メモリ・プロテクション・アトリビュート・レジスタ（L1DMPPA*xx*）

各ページのサイズは、領域ごとに、デバイスごとにそれぞれ異なります。ページによっては、特定のデバイスでは使用できない場合があります。デバッグする目的で、使用していないページにも、すべてゼロの値をセットするようにプログラムしてください。

特定のデバイス上で使用されるページ・サイズおよびページ数を確認するには、各デバイスのデータ・マニュアルを参照してください。

L1D メモリ・プロテクション・アトリビュート・レジスタ（MPPA*xx*）を図 3-13 に示し、表 3-20 で説明します。

図 3-13. L1D メモリ・プロテクション・アトリビュート・レジスタ（L1DMPPA*xx*）

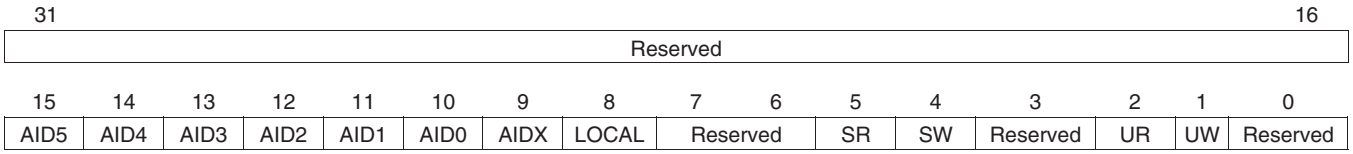


表 3-20. L1D メモリ・プロテクション・アトリビュート・レジスタ（L1DMPPA*xx*）フィールドの説明

ビット	フィールド	値	説明
31-16	Reserved	0	予約。
15	AID5		ID = 5 からのアクセスを制御します。
		0	アクセスは拒否。
		1	アクセスは許可。
14	AID4		ID = 4 からのアクセスを制御します。
		0	アクセスは拒否。
		1	アクセスは許可。
13	AID3		ID = 3 からのアクセスを制御します。
		0	アクセスは拒否。
		1	アクセスは許可。
12	AID2		ID = 2 からのアクセスを制御します。
		0	アクセスは拒否。
		1	アクセスは許可。
11	AID1		ID = 1 からのアクセスを制御します。
		0	アクセスは拒否。
		1	アクセスは許可。
10	AID0		ID = 0 からのアクセスを制御します。
		0	アクセスは拒否。
		1	アクセスは許可。
9	AIDX		Controls ID >=6 からのアクセスを制御します。
		0	アクセスは拒否。
		1	アクセスは許可。
8	LOCAL		CPU からローカル・メモリ（L1/L2）へのアクセスを制御します。
		0	アクセスは拒否。
		1	アクセスは許可。
7-6	Reserved	0	予約。

表 3-20. L1D メモリ・プロテクション・アトリビュート・レジスタ（L1DMPPA*xx*）フィールドの説明（続き）

ビット	フィールド	値	説明
5	SR		スーパーバイザによるリード・アクセス・タイプ。
		0	通常動作。
		1	スーパーバイザによるリード・リクエストを示します。
4	SW		スーパーバイザによるライト・アクセス・タイプ。
		0	通常動作。
		1	スーパーバイザによるライト・リクエストを示します。
3	Reserved	0	予約。
2	UR		ユーザによるリード・アクセス・タイプ。
		0	通常動作。
		1	ユーザによるリード・リクエストを示します。
1	UW		ユーザによるライト・アクセス・タイプ。
		0	通常動作。
		1	ユーザによるライト・リクエストを示します。
0	Reserved	0	予約。

L2 や L1P とは対照的に、L1D は SX（スーパーバイザ実行）および UX（ユーザ実行）ビットを実装していません。L1DMPPA レジスタの SX および UX フィールドは、常にゼロとしてリードされ、ライトには応答しません。

表 3-21 に、メモリ保護に関する 2 つのデフォルト設定を示します。

表 3-21. メモリ保護に関するデフォルト

許可 ID（ビット 15:8）	予約ビット（ビット 7:6）	アクセス・タイプ（ビット 5:0）	注釈
1111 1111	11	110 110	すべてのデバイスには、ユーザ・モードとスーパーバイザ・モードの両方からアクセスできます。

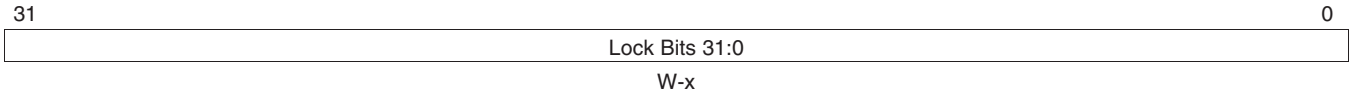
3.7.2.2 L1D メモリ・プロテクション・ロック・レジスタ

L1D コントローラは、メモリ・プロテクション・レジスタへのライト・アクセスを制御するために、64 ビットのロック・レジスタを実装しています。このようなロック・レジスタの動作については、第 8 章を参照してください。

3.7.2.2.1 L1D メモリ・プロテクション・ロック・レジスタ 0（L1DMPLK0）

L1D メモリ・プロテクション・ロック・レジスタ 0（L1DMPLK0）を図 3-14 に示します。

図 3-14. L1D メモリ・プロテクション・ロック・レジスタ 0（L1DMPLK0）

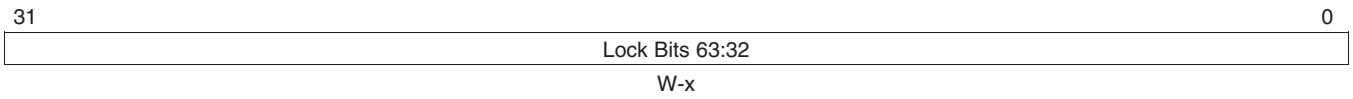


凡例：W = ライト専用。*-x* = 値は不定。詳細については、各デバイスのデータ・マニュアルを参照してください。

3.7.2.2.2 L1D メモリ・プロテクション・ロック・レジスタ 1（L1DMPLK1）

L1D メモリ・プロテクション・ロック・レジスタ 1（L1DMPLK1）を図 3-15 に示します。

図 3-15. L1D メモリ・プロテクション・ロック・レジスタ 1（L1DMPLK1）

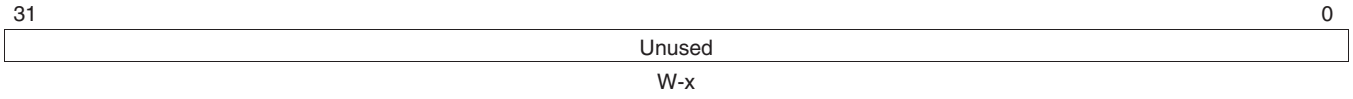


凡例：W = ライト専用。*-x* = 値は不定。詳細については、各デバイスのデータ・マニュアルを参照してください。

3.7.2.2.3 L1D メモリ・プロテクション・ロック・レジスタ 2（L1DMPLK2）

L1D メモリ・プロテクション・ロック・レジスタ 2（L1DMPLK2）を図 3-16 に示します。

図 3-16. L1D メモリ・プロテクション・ロック・レジスタ 2（L1DMPLK2）

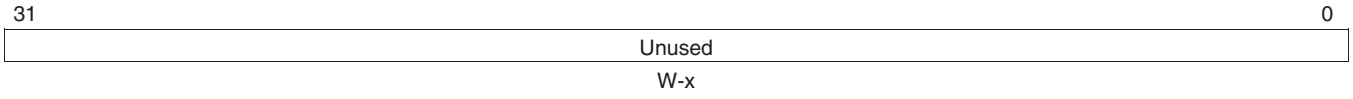


凡例：W = ライト専用。*-x* = 値は不定。詳細については、各デバイスのデータ・マニュアルを参照してください。

3.7.2.2.4 L1D メモリ・プロテクション・ロック・レジスタ 3（L1DMPLK3）

L1D メモリ・プロテクション・ロック・レジスタ 3（L1DMPLK3）を図 3-17 に示します。

図 3-17. L1D メモリ・プロテクション・ロック・レジスタ 3（L1DMPLK3）

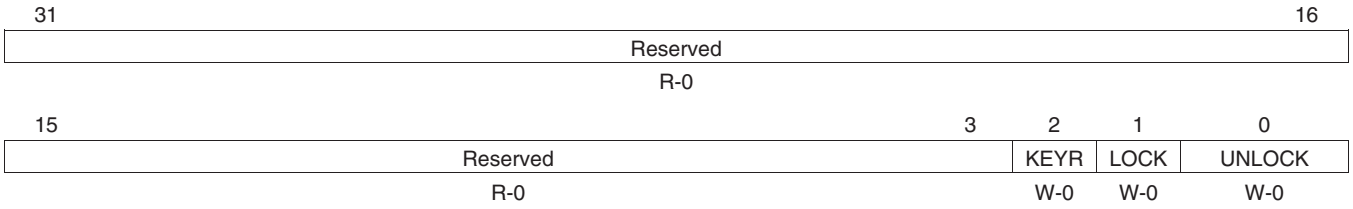


凡例：W = ライト専用。*-x* = 値は不定。詳細については、各デバイスのデータ・マニュアルを参照してください。

3.7.2.2.5 L1D メモリ・プロテクション・ロック・コマンド・レジスタ（L1DMPLKCMD）

L1D メモリ・プロテクション・ロック・コマンド・レジスタ（L1DMPLKCMD）を図 3-18 に示し、表 3-22 で説明します。

図 3-18. L1D メモリ・プロテクション・ロック・コマンド・レジスタ（L1DMPLKCMD）



凡例：R = リード専用。W = ライト専用。*-n* = リセット後の値。

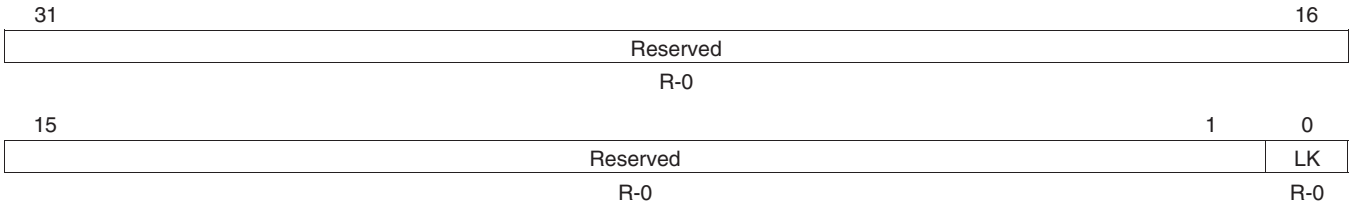
表 3-22. L1D メモリ・プロテクション・ロック・コマンド・レジスタ（L1DMPLKCMD）フィールドの説明

ビット	フィールド	値	説明
31-3	Reserved	0	予約。
2	KEYR		ステータスをリセットします。
		0	影響なし。
		1	ステータスをリセットします。
1	LOCK		ロック・シーケンスを完了するためのインターフェイス。
		0	影響なし。
		1	ソフトウェアがシーケンスを正しく実行した場合に、ロックします。
0	UNLOCK		ロック解除シーケンスを完了するためのインターフェイス。
		0	影響なし。
		1	ソフトウェアがシーケンスを正しく実行した場合に、ロックを解除します。

3.7.2.2.6 L1D メモリ・プロテクション・ロック・ステータス・レジスタ（L1DMPLKSTAT）

L1D メモリ・プロテクション・ロック・ステータス・レジスタ（L1DMPLKSTAT）を図 3-19 に示し、表 3-23 で説明します。

図 3-19. L1D メモリ・プロテクション・ロック・ステータス・レジスタ（L1DMPLKSTAT）



凡例：R = リード専用。*-n* = リセット後の値。

表 3-23. L1D メモリ・プロテクション・ロック・ステータス・レジスタ（L1DMPLKSTAT）フィールドの説明

ビット	フィールド	値	説明
31-1	Reserved	0	予約。
0	LK		ロックの現在のステータスを示します。
		0	ロックは解除されています。
		1	ロックされています。

上記に示したように、メモリ保護アーキテクチャは、最大で 128 ビットのロック・サイズを許容します。L1D は、ロック・インターフェイスに対し 64 ビット・ロックのみを実装しています。そのため、L1DMPLCK2 および L1DMPLCK3 にライトされた値は無視されます。128 ビットより短いキーに関するロック・メカニズムの動作については、第 8 章を参照してください。

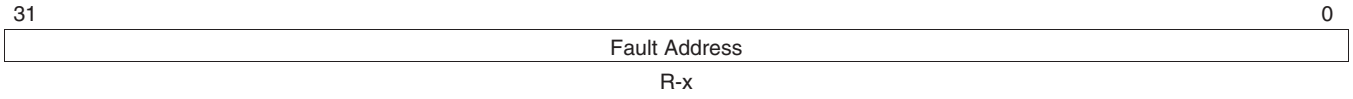
3.7.2.3 L1D メモリ・プロテクション・フォールト・レジスタ

例外発生後にプログラムでメモリ保護障害を診断するために、L1D には障害に関する情報を格納する専用の レジスタが実装されています。このようなレジスタを図 3-20 ～ 図 3-22 に示します。

3.7.2.3.1 L1D メモリ・プロテクション・フォールト・アドレス・レジスタ（L1DMPFAR）

L1D メモリ・プロテクション・フォールト・アドレス・レジスタ（L1DMPFAR）を図 3-20 に示し、表 3-24 で説明します。

図 3-20. L1D メモリ・プロテクション・フォールト・アドレス・レジスタ（L1DMPFAR）



凡例：R = リード専用。*-x* = 値は不定。詳細については、各デバイスのデータ・マニュアルを参照してください。

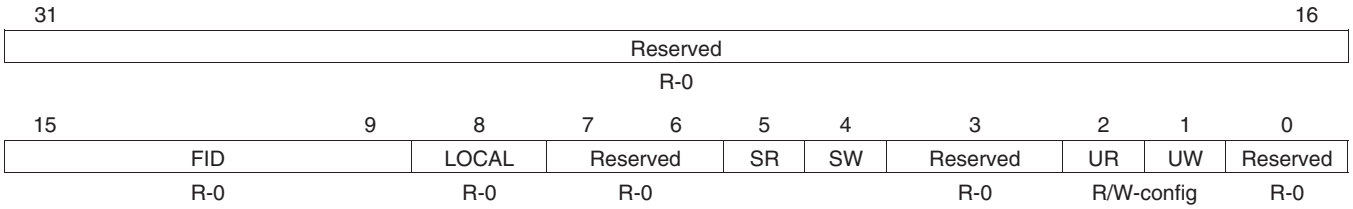
表 3-24. L1D メモリ・プロテクション・フォールト・アドレス・レジスタ（L1DMPFAR）フィールドの説明

ビット	フィールド	値	説明
31-0	Fault Address	0 ～ FFFF FFFFh	障害が発生したアドレス。

3.7.2.3.2 L1D メモリ・プロテクション・フォールト・セット・レジスタ（L1DMPFSR）

L1D メモリ・プロテクション・フォールト・セット・レジスタ（L1DMPFSR）を図 3-21 に示し、表 3-25 で説明します。

図 3-21. L1D メモリ・プロテクション・フォールト・セット・レジスタ（L1DMPFSR）



凡例：R/W = リード / ライト。R = リード専用。-*n* = リセット後の値。

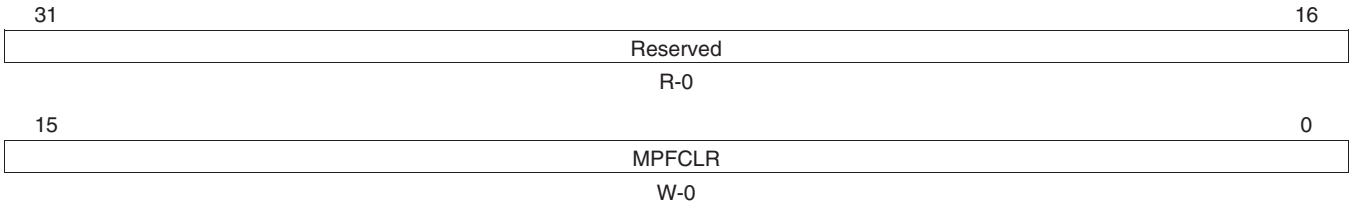
表 3-25. L1D メモリ・プロテクション・フォールト・セット・レジスタ（L1DMPFSR）フィールドの説明

ビット	フィールド	値	説明
31-16	Reserved	0	予約。
15-9	FID	0 ～ 7Fh	ビット 6:0 は障害を引き起こしたリクエスタ の ID を示します。ID の幅が 7 ビットより狭い場合、残りのビットは 0 を返します。ID の幅が 7 ビットより広い場合、他のビットは切り捨てられます。LOCAL = 1 の場合、FID = 0。
8	LOCAL		ローカル・アクセス。
		0	通常動作。
		1	アクセスはローカル。
7-6	Reserved	0	予約。
5	SR		スーパーバイザによるリード・アクセス・タイプ。
		0	通常動作。
		1	スーパーバイザによるリード・リクエストを示します。
4	SW		スーパーバイザによるライト・アクセス・タイプ。
		0	通常動作。
		1	スーパーバイザによるライト・リクエストを示します。
3	Reserved	0	予約。
2	UR		ユーザによるリード・アクセス・タイプ。
		0	通常動作。
		1	ユーザによるリード・リクエストを示します。
1	UW		ユーザによるライト・アクセス・タイプ。
		0	通常動作。
		1	ユーザによるライト・リクエストを示します。
0	Reserved	0	予約。

3.7.2.3.3 L1D メモリ・プロテクション・フォールト・クリア・レジスタ（L1DMPFCR）

L1D メモリ・プロテクション・フォールト・クリア・レジスタ（L1DMPFCR）を図 3-22 に示し、表 3-26 で説明します。

図 3-22. L1D メモリ・プロテクション・フォールト・クリア・レジスタ（L1DMPFCR）



凡例：R = リード専用。W = ライト専用。*-n* = リセット後の値。

表 3-26. L1D メモリ・プロテクション・フォールト・クリア・レジスタ（L1DMPFCR）フィールドの説明

ビット	フィールド	値	説明
31-1	Reserved	0	予約。
0	MPFCLR		L1DMPFAR レジスタをクリアするためのコマンド。
		0	影響なし。
		1	L1DMPFAR および L1DMPFCR レジスタをクリアします。

これらのレジスタの定義と意味については、第 8 章で説明します。

L1DMPFAR および L1DMPFSR レジスタは、1 つの障害に関する十分な情報のみを格納します。ハードウェアは最初の障害に関する情報を記録し、その障害のみの例外を生成します。

障害情報は、ソフトウェアで L1DMPFCR レジスタの MPFCLR フィールドに 1 をライトしてクリアするまで、保持されます。L1DMPFCR レジスタの MPFCLR フィールドに 0 をライトしても影響はありません。L1D は、L1DMPFCR レジスタのビット 31:1 にライトされた値を無視します。

3.7.3 メモリ・プロテクション・レジスタへのアクセス時に行われる保護チェック

L1D は、メモリ・プロテクション・レジスタ自体にアクセス権限チェック機能を実装しています。ルールは次のとおりです。

- すべてのリクエスタは、いつでもどんな状況でも任意のメモリ・プロテクション（MP）レジスタをリードできます。ただし、メモリ・プロテクション・ロック・レジスタ（L1DMPLK0 ～ L1DMPLK3）は除きます。
- スーパーバイザはレジスタをライトできます。

表 3-27 に、役割ごとにアクセスできる L1D メモリ・プロテクション・レジスタおよびメガモジュールで行われる保護チェックの内容について概要を示します。

表 3-27. L1D メモリ・プロテクション・レジスタのアクセス権限

レジスタ	スーパーバイザ	ユーザ
L1DMPFAR	R	R
L1DMPFSR	R	R
L1DMPFCR	W	/
L1DMPLK0	W	/
L1DMPLK1	W	/
L1DMPLK2	W	/
L1DMPLK3	W	/
L1DMPLKCMD	W	/
L1DMPLKSTAT	R	R
L1DMPPAxx	R/W	R

第 4 章

JAJU058

レベル 2 メモリ / キャッシュ

項目 **ページ**

4.1 はじめに

4.1.1 レベル 2（L2）メモリ / キャッシュの目的

L2 メモリ・コントローラは、高速なレベル 1 メモリ（L1D、L1P）と低速な外部メモリ間でオンチップ・メモリ・ソリューションを実現します。また L1 メモリより大きいメモリ・サイズをサポートするという点で優位性があり、外部メモリより高速にアクセスできます。

L1 メモリと同様に、L2 を設定して、キャッシュ・メモリとキャッシュ以外のメモリ（つまり、アドレス可能なメモリ）の両方に使用することができます。

4.1.2 機能

L2 メモリ / キャッシュは、C64x+ メガモジュールを使用するデバイスで求められるメモリの柔軟性を提供します。

- 2 つのメモリ・ポート（ポート 0、ポート 1）
- 設定可能な L2 キャッシュ・サイズ：32KB、64KB、128KB、256KB
- メモリ保護
- キャッシュ・ブロックおよびグローバルなコヒーレンス動作をサポート
- 4 ページの設定可能なパワーダウン

4.1.3 用語と定義

本章で使用する用語の詳細な定義については、本書の付録 A および付録 B を参照してください。付録 A では、本書全体で使用している一般的な用語について説明し、付録 B ではメモリ / キャッシュ・アーキテクチャ関連の用語を定義しています。

4.2 レベル 2 メモリ・アーキテクチャ

4.2.1 L2 メモリ

4.2.1.1 L2 メモリ・ポート

L2 メモリには 2 つの 256 ビット幅のメモリ・ポートが用意されています（ポート 0 およびポート 1）。2 つのポートの使用方法は、デバイスによって異なります。ほとんどのデバイスでは、2 つのメモリ・ポートは次のように使われます。

- ポート 0
  - L2 RAM
  - L2 キャッシュ
- ポート 1
  - L2 ROM
  - L2 RAM
  - 共有メモリ・インターフェイス

4.2.1.2 L2 メモリ・サイズ

L2 コントローラは、ポートごとに 64K ～ 819K の範囲内のメモリ・サイズをサポートします。

4.2.1.3 L2 メモリ・バンキング

L2 メモリは、2 つの異なるメモリ・ポートを実装しています。それぞれのメモリ・ポートは次のいずれかを制御できます。

- 4 x 128 ビット・バンク
- 2 x 128 ビット・バンク
- 1 x 256 ビット・バンク

特定のデバイスに実装されているバンキング方式の詳細については、各デバイスのデータ・マニュアルを参照してください。

2 つのメモリ・ポートは、メモリ・セクションをアドレス指定できます。このメモリ・セクションは連続している場合もそうでない場合もあります。

表 4-1 に、2 x 128 ビットの場合のポート 0 およびポート 1 のバンキングの様子を示します（リトルエンディアン・モード）

表 4-1. C64x+ メガモジュールにおける 2 x 128 ビットのバンキング方式

ポート 1							
バンク 1				バンク 0			
xx14	xx16	xx15	xx14	xx13	xx12	xx11	xx10
xx0F	xx0E	xx0D	xx0C	xx0B	xx0A	xx09	xx08
xx07	xx06	xx05	xx04	xx03	xx02	xx01	xx00
ポート 0							
バンク 1				バンク 0			
yy17	yy16	yy15	yy14	yy13	yy12	yy11	yy10
yy0F	yy0E	yy0D	yy0C	yy0B	yy0A	yy09	yy08
yy07	yy06	yy05	yy04	yy03	yy02	yy01	yy00

注： 2 つのメモリ・ポートは、連続している場合もそうでない場合もあります。

L1P リード・ミス（32 バイト）は、シングルポートではすべてのメモリ・バンクが必要です。L2 メモリがハイレイテンシの場合、同一サイクル時、つまり、アクセスが完了するまで、そのポートではそれ以外のアクセスは行われません（メモリのレイテンシはチップ設計時に決定されます。詳細については、各デバイスのデータ・マニュアルを参照してください）。L1P リード・ミス（64 バイト）およびビクティム・ライトバックは、2 つのアクセスを連続して行うために、シングルポートではすべてのメモリ・バンクが必要です。

4.2.1.4 L2 メモリへの同時アクセス

L1P、L1D、IDMA などさまざまなリクエスタが L2 メモリへ同時にアクセスしようとすると、そのアクセスは第 6 章で説明しているルールによって調停されます。

4.3 L2 キャッシュ

C64x+ CPU のデフォルト設定は、すべての L2 メモリを RAM/ROM としてマップします。L2 コントローラのポート 0 は 32KB、64KB、128KB、256KB のいずれかの 4 ウェイ・セット・アソシアティブ・キャッシュをサポートします。ポート 0 の 256KB を超える他のメモリおよびポート 1 に接続されたすべてのメモリは、常に RAM か ROM になります。

L2 キャッシュの動作は、いくつかのレジスタによって制御されます。表 4-2 に、このようなレジスタの概要を示します。ここではこのようなレジスタを簡単に説明します。詳細については 4.4 節を参照してください。

表 4-2. キャッシュ・レジスタの概要

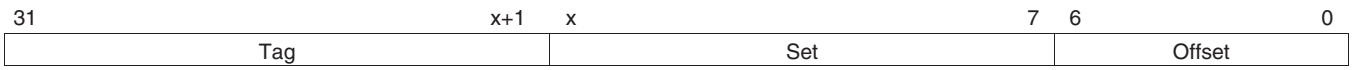
略称	レジスタの説明	参照先
L2CFG	レベル 2 コンフィギュレーション・レジスタ	4.4.2 項
L2WBAR	レベル 2 ライトバック・ベース・アドレス・レジスタ	4.4.3.1.1 項
L2WWC	レベル 2 ライトバック・ワード・カウント・レジスタ	4.4.3.1.2 項
L2WIBAR	レベル 2 ライトバック・インバリデート・ベース・アドレス・レジスタ	4.4.3.1.3 項
L2WIWC	レベル 2 ライトバック・インバリデート・ワード・カウント・レジスタ	4.4.3.1.4 項
L2IBAR	レベル 2 インバリデート・ベース・アドレス・レジスタ	4.4.3.1.5 項
L2IWC	レベル 2 インバリデート・ワード・カウント・レジスタ	4.4.3.1.6 項
L2WB	レベル 2 ライトバック・レジスタ	4.4.3.2.1 項
L2WBINV	レベル 2 ライトバック・インバリデート・レジスタ	4.4.3.2.2 項
L2INV	レベル 2 インバリデート・レジスタ	4.4.3.2.3 項
MARn	メモリ・アトリビュート・レジスタ	4.4.4 項

4.3.1 L2 キャッシュ・アーキテクチャ

L2 キャッシュは、リード・アンド・ライト・アロケートな 4 ウェイ・セット・アソシアティブ・キャッシュです。L2 キャッシュのライン・ステートを記録するために、4 ウェイ・タグ RAM が組み込まれています。L2 タグ内のアドレス構成は、キャッシュと RAM 間で行われる分割機能で、L2CFG レジスタの L2MODE フィールドのビット設定で制御されます。

図 4-1 に、サポートされているさまざまなキャッシュ・サイズに対応した機能を示します。

図 4-1. L2 キャッシュ・アドレス構成



7 ビットからなる Offset は、L2 ラインのサイズが 128 バイトであるということを示します。キャッシュ制御ロジックは、このアドレス部分を無視します。Set フィールドは、データがそれぞれのロケーション（ウェイ）に存在する L2 キャッシュ・ライン・アドレスを示します（データがキャッシュされている場合）。Set フィールドの幅は、キャッシュとして構成された L2 の量によって異なります（表 4-3 を参照）。L2 コントローラは Set フィールドを使用して、すでにキャッシュされているデータのタグを各ロケーション（ウェイ）で調べ、チェックします。また有効なビットを調べ、タグを比較するために、そのラインの内容が有効であると見なすかどうかを示します。

L2 キャッシュを設定すると、Set と Tag フィールドのサイズが決定されます（表 4-3 を参照）。

表 4-3. L2MODE の説明

L2CFG レジスタの L2MODE 設定	L2 キャッシュの量	X ビットの位置	説明
000b	0K	N/A	L2 はすべての RAM
001b	32K	12	64 本の L2 キャッシュ・ライン
010b	64K	13	128 本の L2 キャッシュ・ライン
011b	128K	14	256 本の L2 キャッシュ・ライン
100b	256K	15	512 本の L2 キャッシュ・ライン
101b	予約。256K にマップ。		
110b			
111b	最大のキャッシュ。256K にマップ。		

注： 一般に、L2MODE の値を大きくしたら、キャッシュ・サイズの指定も大きくします（ただし、ポート 0 に実装されている L2 メモリの最大サイズまで）。L2 キャッシュの実際の最大サイズは、L2 RAM サイズに収まる最大の 2 の累乗と 256K の小さい方になります。

Tag フィールドは、このアドレスの上位部分で、キャッシュ・ラインの正しい物理ロケーションを識別します。キャッシュは、特定のアドレスの Tag フィールドを L2 キャッシュの 4 ウェイすべてに格納されているタグと比較します。

タグのいずれかが一致し、かつキャッシュされたデータが有効の場合、そのアクセスは「ヒット」になり、そのエレメントは L2 キャッシュ・ロケーション間で直接リード / ライトが行われます。それ以外の場合、アクセスは「ミス」になり、リクエスタは L2 がシステム・メモリ・ロケーションから完全なラインをフェッチしている間、ストールしたままになります。リード・ミスが発生すると、データはフェッチの一部として適切な L1 キャッシュに直接渡されます。ライト・ミスが発生すると、L2 はライトをフェッチ対象ラインにマージします。

L2 の内容を変更できるため、L2 キャッシュは正しい物理ロケーションでデータを更新できます。L2 キャッシュはライトバック・キャッシュです。これは、必要なときにのみ更新を完全にライトするということです。データは L2 キャッシュから追い出され、システム・メモリの適切なロケーションにライトバックされます。これが行われるのは、新たな L2 ラインが変更されたラインを置き換えるときか、L2 コントローラに対して（ソフトウェアで）CPU から変更されたデータをライトバックするように指示されたときです。データの追い出しまたはライトバックが行われると、データは EMC によってシステム・メモリの該当ロケーションに送られます。

4.3.2 リプレースメントおよびアロケーション処理

L2 キャッシュは、すべてのキャッシュ・モードで固定された 4 ウェイ・セット・アソシアティブ・キャッシュとして動作します。これは、システム・メモリの各ロケーションは L2 キャッシュの 4 つの指定可能なロケーションのいずれかに存在するということです。

L2 コントローラは、リード / ライト・アロケート・キャッシュを実装しています。これは、リードまたはライトに関係なく、L2 がキャッシュ可能なロケーションに対するミス時に 128 バイトのライン全体をフェッチすることになるということです。リプレースメント処理は、最低使用頻度（LRU）の L2 ラインを新しいラインで置き換えるという点で、L1D の処理と同等です。

4.3.3 リセット動作

グローバル・リセットに対応して、L2 キャッシュは「すべて RAM モード」に切り替わります。

ローカル・リセットに対して、L2 キャッシュは現在の動作モードを保持します。ただし、キャッシュの内容全体はインバリデートされます。すべてのリクエスタは、このインバリデートが有効になっている間ストールします。

レベル 1 キャッシュのサポートがメガモジュール内でイネーブルの場合、L2 コントローラはレベル 1 キャッシュがハードウェア・リセットおよびソフトウェア・リセットに対して L2 と同じように応答しているかを確認するのに必要な手順を取ります。

4.3.4 L2 モード変更動作

L2 のキャッシュ・サイズは、実行時に設定できます。プログラムで L2 のキャッシュ・サイズを選択するには、リクエストされたモードを L2CFG レジスタの L2MODE フィールドにライトします。表 4-4 に、L2MODE モードで有効な設定を示します。

表 4-4. L2CFG.L2MODE で指定されるキャッシュ・サイズ

L2CFG レジスタの L2MODE 設定	L2 キャッシュの量
000b	0K
001b	32K
010b	64K
011b	128K
100b	256K
101b	予約。C64x+ メガモジュールでは、256K にマップ。
110b	
111b	最大のキャッシュ。C64x+ メガモジュールでは、256K にマップ。

通常、プログラムでは、リセット直後に L2 モードにセットし、そのモードを変更しません。ただし、プログラムによっては、複雑なシステムの特に OS タスク周辺では、L2 キャッシュ・モードをオンザフライで変更するものもあります。この手順に確実に従うことで、メモリ・システムのコヒーレンスおよび正しいキャッシュ動作が維持されることに注意してください。

表 4-5 に、必要な対策を示します。

表 4-5. L2 モードへの切り換え

変更前のモード	変更後のモード	プログラムで、次のステップを実行する必要がある
L2 キャッシュをまったく使用しないか、一部使用するモード	L2 キャッシュをより多く使用するモード	1. DMA、IDMA を使用して、L2 RAM の影響を受ける範囲から必要なデータをコピーします（保存する必要がなければ、DMA は不要）。 2. 前のステップで発行された DMA/IDMA の完了を待ちます。 3. 目的のキャッシュ・モードを L2CFG レジスタの L2MODE フィールドにライトします。 4. L2CFG レジスタをリードバックします。これにより、モード変更が完了するまで CPU はストールします。
L2 キャッシュを一部使用するモード	L2 キャッシュをより少なく、あるいはまったく使用しないモード	1. 目的のキャッシュ・モードを L2CFG レジスタの L2MODE フィールドにライトします。 2. L2CFG レジスタをリードバックします。これにより、モード変更が完了するまで CPU はストールします。

プログラムで、新しいキャッシュ・モードを L2CFG レジスタへライトする場合、L2 は次のステップを実行します。

- L2 がイネーブルの場合、ライトバックされ、かつインバリデートされます。
- L2 キャッシュはリクエストされたモードにセットされます。

注： L2 のモード変更は、L1 キャッシュのいずれ（L1D、L1P）の内容にも影響を与えません。

4.3.5 L2 フリーズ・モード

L2 キャッシュは、フリーズ・モードをサポートします。このモードでは、L2 キャッシュの内容はフリーズされます（つまり、正常動作中、その内容は更新されません）。L2 モードを使用すると、リアルタイム・アプリケーションは、割り込みハンドラなどのコードのさまざまなセクションで L2 から追い出されるデータ量を制限することができます。L2CFG レジスタの L2CC フィールドを使用して、このモードをセットします。

フリーズ・モードは、L2 キャッシュの動作にのみ影響を与えます。L2 RAM は、フリーズ・モードの影響を受けません。L2 のフリーズ・モードは、L1D または L1P キャッシュのいずれにも影響を与えません。同様に、L1 のフリーズ・モードは、L2 キャッシュに影響を与えません。

L2 キャッシュは、フリーズ・モードの場合、通常、リード・ヒットおよびライト・ヒットに応答します。L2 は、L2 キャッシュが存在しなかったものとして、リード・ミスおよびライト・ミスを直接外部メモリへ送ります。L2 は、フリーズ中に、新たなキャッシュ・ラインを割り当てません。フリーズ・モードでは、プログラムによって開始されるキャッシュ・コヒーレンス動作によって、ラインは L2 からのみ追い出される場合があります（4.3.6 項 を参照）。

表 4-6 に、L2CFG レジスタの L2CC フィールドでセットされる L2 フリーズ・モードの概要を示します。

表 4-6. フリーズ・モードの概要

L2 モード	L2MODE	L2 キャッシュはイネーブル L2CC = 0	L2 キャッシュはイネーブル L2CC = 0	L2 キャッシュはフリーズ L2CC = 1
すべての RAM	000	影響なし（L2 はすべて RAM のため）。		
キャッシュ、RAM、またはすべてのキャッシュの混合	1000	キャッシュは通常動作。	キャッシュはフリーズ。ヒットは正常に進行。L1D ミスは、リクエストされたバイトのみをロング・ディスタンス・アクセスとして処理されます。L1P ミスは、1 フェッチ・パケットをロング・ディスタンス・フェッチとして処理されます。このモードでは、LRU の更新は行われません。	

4.3.6 プログラムによって開始されるキャッシュ・コヒーレンス動作

L2 メモリ・アーキテクチャは、2 つの主要なカテゴリに分類するさまざまなコヒーレンス動作をサポートします。1 つは固有のアドレス範囲上で動作するブロック動作で、もう 1 つは 1 つ以上のキャッシュからなる内容全体で動作するグローバル動作です。

サポートされているキャッシュ・コヒーレンス動作は、次のとおりです。

- インバリデート：有効なキャッシュ・ラインは、無効になります。影響を受けるキャッシュ・ラインの内容は、廃棄されます。
- ライトバック：有効なダーティ・キャッシュ・ラインの内容は、低レベル・メモリへライトされます。
- ライトバック・インバリデート：ライトバック動作に続きインバリデートが行われます。影響を受けるキャッシュ・ラインの内容のみが、低レベル・メモリへライトされますが、すべてのラインがインバリデートされます。

4.3.6.1 グローバル・コヒーレンス動作

グローバル・コヒーレンス動作は、L2 キャッシュ全体で実行されます。またグローバル・コヒーレンス動作によっては、L1 キャッシュにも影響を与えるものもあります。

表 4-7 に、3 つのキャッシュそれぞれで行われる L2 グローバル・キャッシュ・コマンドおよびその動作についてすべて示します。

表 4-7. グローバル・コヒーレンス動作

キャッシュ動作	使用レジスタ	L1P への影響	L1D への影響	L2 への影響
L2 ライトバック	L2WB	影響なし。	すべての更新されたデータは L2 / 外部メモリへライトバックされますが、L1D では有効なままです。	すべての更新されたデータは、外部メモリへライトバックされますが、L2 キャッシュでは有効なままです。
L2 ライトバック・インバリデート	L2WBINV	すべてのラインは、L1P でインバリデートされます。	すべての更新されたデータは L2 / 外部メモリへライトバックされます。すべてのラインは、L1D 内でインバリデートされます。	すべての更新されたデータは、外部メモリへライトバックされます。すべてのラインは、L2 でインバリデートされます。
L2 インバリデート	L2INV	すべてのラインは、L1P でインバリデートされます。	すべてのラインは、L1D でインバリデートされます。更新されたデータは、ドロップされます。	すべてのラインは、L2 でインバリデートされます。更新されたデータは、ドロップされます。

プログラムで L2WB、L2WBINV、L2INV の各レジスタの適切なレジスタ・ビットに 1 を ライトすると、グローバル・キャッシュ動作を開始します。

プログラムで L2WB、L2WBINV、L2INV の各レジスタに対してコントロール・レジスタに 1 を ライトすると、コヒーレンス動作を開始します。コントロール・レジスタは、動作が完了すると 0 をセットします。プログラムで、このフィールドをポーリングすると、コマンドの完了を検出できます。

例 4-1 に、L2WBINV レジスタの使用方法の例を示します。

例 4-1. グローバル・コヒーレンス動作の例

```
/* ------------------------------------------------------------ */
/*  Write back and Invalidate anything held in cache.           */
/* ------------------------------------------------------------ */
L2WBINV = 1;

/* ------------------------------------------------------------ */
/*  OPTIONAL:  Spin waiting for operation to complete.          */
/* ------------------------------------------------------------ */
while ((L2WBINV & 1) != 0)
    ;
```

ハードウェアでは、これらのコマンドの完了をポーリングするためのプログラムは必要ありません。ただし、ハードウェアはグローバル・コマンドの実行が進行中にプログラムをストールできます。

グローバル・キャッシュ動作は、L2 のフリーズ状態に関係なく正常に動作します。さらに、グローバル・キャッシュ動作が、L2 キャッシュのフリーズ状態を変更することはありません。

4.3.6.2 ブロック・コヒーレンス動作

ブロック・コヒーレンス動作は、グローバル・コヒーレンス動作と機能的にはほぼ同じです。ただし、指定されたデータ・ブロックにのみ適用される点が異なります。このブロックは、対応するレジスタのベース・アドレスおよびワード・サイズ（32 ビット）で指定されます。

表 4-8 に、3 つのキャッシュそれぞれで行われる L2 グローバル・キャッシュ・コマンドおよびその動作についてすべて示します。

表 4-8. ブロック・キャッシュ・コヒーレンス動作

キャッシュ動作	使用レジスタ	L1P への影響	L1D への影響	L2 への影響
L2 ライトバック	L2WBAR L2WWC	影響なし。	更新されたデータは L2 / 外部メモリへライトされますが、L1D では有効なままです。	更新されたデータは、外部メモリへライトバックされますが、L2 キャッシュでは有効なままです。
L2 ライトバック・インバリデート	L2WIBAR L2WIWC	範囲内のすべてのラインは、L1P でインバリデートされます。	更新されたデータは、L2 / 外部メモリへライトバックされます。範囲内のすべてのラインは、L1D でインバリデートされます。	更新されたデータは、外部メモリへライトバックされます。範囲内のすべてのラインは、L2 でインバリデートされます。
L2 インバリデート	L2IBAR L2IWC	範囲内のすべてのラインは、L1P でインバリデートされます。	範囲内のすべてのラインは、L1D でインバリデートされます。更新されたデータは、ドロップされます。	範囲内のすべてのラインは、L2 でインバリデートされます。更新されたデータは、ドロップされます。

プログラムでワード・アドレスをベース・アドレス・レジスタへライトしてから、ワード・カウントをワード・カウント・レジスタへライトすると、ブロック・キャッシュ動作を開始します（WC へ 1 をライトすることは、4 バイトの長さを示します）。必要に応じて、C64x+ メガモジュールは、次の処理の一方または両方を実行します。

- ただ 1 つのプログラムによって開始されたコヒーレンス動作のみが、一度に実行される
- 別のブロック・コヒーレンス動作またはグローバル・キャッシュ・コヒーレンス動作が進行中に、xxBAR または xxWC のいずれかへライトすると、ストールする

ブロック・キャッシュ動作をセットアップする xxBAR/xxWC メカニズムによって、ワードの粒度に合わせて範囲を指定することができます。ただし、メモリ・システムはキャッシュ・ラインの粒度で動作します。そのため、指定した範囲にオーバーラップするすべてのキャッシュ・ラインは、その影響を受けます。

例 4-2 に、ブロック・キャッシュ・コントロール・レジスタの使用方法の例を示します。

例 4-2. ブロック・コヒーレンス動作の例

```
/* ---------------------------------------------------------------- */
/*  Write base address of array to Base Address Register.  Then     */
/*  Then write the length of the array, in words, to the Word       */
/*  Count register.                                                 */
/* ---------------------------------------------------------------- */
L2WBAR = &array[0];
L2WWC  = size of(array) / size of(int);

                                /* . . . */

/* ---------------------------------------------------------------- */
/*  The CPU can execute other code here.  Block cache operations    */
/*  proceed in parallel with CPU execution, stalling the CPU        */
/*  minimally.                                                      */
/* ---------------------------------------------------------------- */

                                /* . . . */

/* ---------------------------------------------------------------- */
/*  OPTIONAL:  Spin waiting for operation to complete.              */
/* ---------------------------------------------------------------- */
while (L2WWC != 0)
    ;
```

xxBAR レジスタへライトすると、次のキャッシュ・コヒーレンシ動作を行うためのベース・アドレスがセット・アップされます。ゼロ以外の値を xxWC へライトすると、動作を開始します。ブロック・キャッシュ・ロジックは、ライトされた特定の xxWC レジスタに基づき、コヒーレンス・コマンドを開始します。

プログラムは、キャッシュ・コントロール動作中または動作後に、xxBAR の内容を利用してはいけません。その代わりに、プログラムは常に xxWC をライトする前に、新しい値を xxBAR へライトするべきです。ブロック・キャッシュ動作が行われている間に、xxWC をリードすると、ゼロ以外の値が返されます。また、完了時にリードすると、ゼロが返されます。

ブロック・キャッシュ動作は、L2 のフリーズ状態に関係なく正常に動作します。

4.3.7 キャッシュ可能性の制御

アプリケーションでは、一部の特定アドレスをアクセスするたびに、その物理ロケーションからリードされる場合があります（たとえば、FPGA 内のステータス・レジスタ）。

L2 コントローラには、特定の範囲のメモリがキャッシュ可能かどうか、1 つ以上のリクエスタが実際に該当範囲をアクセスできるかどうかを制御するレジスタがあります。このようなレジスタを、MAR（メモリ・アトリビュート・レジスタ）といいます。MAR のレジスタのすべてを 4.4.4 項に示します。

注： C 言語のキーワード volatile を使用しても、変数をキャッシュさせないようにすることはできません。アプリケーションで外部ハードウェアによって定期的に更新されるメモリ・ロケーションを使用する場合、この C コードが動作するようにするためには、次の 2 つのステップに従います。

- キーワード volatile を使用して、コード生成ツールが変数を不適切に最適化しないようにします。
- 変数を含む範囲の MAR レジスタをキャッシュしないようにプログラムする必要があります。

4.3.7.1 MAR の機能

それぞれの MAR レジスタは、1 つのビットを実装しています。各 MAR レジスタの許可コピー（PC）ビットは、キャッシュが影響を受けるアドレス範囲のコピーの保持が許されるかどうかを制御します。PC = 1 の場合、影響を受けるアドレス範囲はキャッシュ可能です。PC = 0 の場合、影響を受けるアドレス範囲はキャッシュ不可能です。

MAR レジスタは、実行時にプログラム可能です。ただし、4.3.7.2 項に記述されたレジスタを除きます。MAR レジスタのすべてのビットは、値 0 にリセットされます。その結果、デフォルトではアドレス空間全体がキャッシュ不可能になります（ただし、4.3.7.2 項に記述されたレジスタを除く）。

4.3.7.2 特殊な MAR レジスタ

MAR0 ～ MAR15 は、C64x+ メガモジュールで予約されているアドレス範囲を示します。そのため、次のように処理されます。

1. MAR0 はリード専用の値として実装されます。MAR0 の PC は、常に 1 としてリードされます。
2. MAR1 ～ MAR15 は、内部および外部のコンフィギュレーション・アドレス空間に対応します。したがって、これらのレジスタはリード専用で、その PC フィールドは 0 としてリードされます。

MAR0 ～ MAR15 はリード専用のため、ソフトウェアでこれらのレジスタを操作する必要はありません。

4.3.7.3 L1 へアクセスする場合の相互の影響

L1P または L1D が、L2 RAM または L2 キャッシュに保持されていないアドレスへアクセスするために、L2 へリクエストする場合、L2 はそのアドレスに対して対応する MAR レジスタを照会します。MAR レジスタの許可コピー（PC）ビットが 0 の場合、L2 キャッシュ・コントローラはこのビットをキャッシュ不可能なアクセスとして処理し、ロング・ディスタンス・アクセスを開始します。アクセスがロング・ディスタンス・リードの場合、CPU はリードされたデータが返るまでストールします。

L1D ロング・ディスタンス・リクエストに関して、MAR の PC ビットは結果的に、キャッシュ不可能なデータを L2 または L1D キャッシュに格納しません。そのため、特定の MAR レジスタで PC = 0 の場合、L1D および L2 キャッシュは両方ともその MAR でカバーされるアドレス範囲内でアクセスされるデータのコピーを保持しません。

MAR レジスタは、L1P には影響を与えません。L1P がイネーブルの場合、MAR の設定には関係なく、プログラム・フェッチを常にキャッシュします。

4.3.8 L1-L2 コヒーレンス・サポート

ここでは、L2 キャッシュと L1D/L1P キャッシュ間のコヒーレンス・ルールにより課せられている相互の影響について説明します。

C64x+ が保持しているコヒーレンス・モデルは次のとおりです。

1. C64x+ メガモジュールの L2 RAM セグメントと L1D キャッシュ間のコヒーレンスは保持されている。
2. C64x+ メガモジュールの L2 RAM セグメントと L1P キャッシュ間のコヒーレンスは保持されていない。
3. 外部メモリと L1/L2 キャッシュにキャッシュされているコピー間のコヒーレンスは保持されていない。

上記の項目 1 と 3 は、C64x の動作と同じです。上記の項目 2 は、C64x の動作とは異なります。C64x 上では、L2 RAM への DMA ライトと L2 RAM からのプログラム・フェッチ間では、コヒーレンスは確保されています。C64x+ メガモジュールを搭載したデバイスは、コード・ブロック内で転送を行う場合、ユーザが手動でブロック・インバリデートを発行する必要があります（4.3.6.2 項を参照）。

これ以降では、L2-RAM-to-L1 キャッシュ・コヒーレンスを実現する機能について概要を説明します。

4.3.8.1 キャッシュ・コヒーレンス・プロトコル

L1 キャッシュと L2 キャッシュ間でコヒーレンスをサポートするために、スヌープ・リードとスヌープ・ライトを行うコマンドを使用します。

キャッシュ・コヒーレンス・プロトコルに実装されている機能は、C64x デバイスに実装されている機能とは異なる場合があります。C64x+ プロトコルでは、コヒーレンスは L2 内の DMA と L1D 間でサポートされていますが、DMA と L1P 間ではサポートされていません。また、C64x メモリ・アーキテクチャでは、L1D キャッシュは L2 内に包含されているため、L2 キャッシュ動作に対応したスヌープが必要です。C64x+ メガモジュールはこの包含を取り除き、その結果、スヌープを DMA の動作によってトリガされるものだけに限定します。L1 が L2 に包含されなくとも、L1 および L2 は、お互いにコヒーレントのままです。

キャッシュ「A」がキャッシュ「B」に包含されるのは、A の内容が常に B のサブセットの場合です。ラインが B ではなく A に保持されていると、A は B には包含されません。アドレスが以前キャッシュ可能だった場合、キャッシュ不可能なライトが L2 にヒットすることがあります。これが起こりうるのは、アプリケーションによって、MAR レジスタの設定が動的に変更される場合です。

表 4-9 に、L2 が L1D に対して発行できるコヒーレンス・コマンドをキャッシュ・ラインごとに示します。

表 4-9. L2 から L1D へのコヒーレンス・コマンド

スヌープ・コマンド	コマンド名	L1D の動作	トリガの原因
SNPR	スヌープ・リード	L1D は、L1D でリクエストされたハーフラインの内容を L2 へ送ります。このラインに対するダーティ / 有効 / LRU 状態を変更しません。	L1D シャドウ・タグは、ラインが L1D に存在し、かつ L1D で変更されたことを示す場合、L2 RAM からリードされる DMA。
SNPW	スヌープ・ライト	新しいデータの最大で 256 ビットが L2 から L1D へ送られます。L1D と L2 は両方ともそれぞれのデータのコピーを更新します。L1D に含まれるラインでは、ダーティ・ビットと有効ビットは変更されません。	L1D シャドウ・タグは、ラインが L1D に存在していることを示す場合、L2 RAM からリードされる DMA。ラインが L1D で変更されたどうかは、問題にはなりません。

注： スヌープ・コマンドは、ユーザには見えないハードウェアの動作を表します。ここの説明は、ユーザがキャッシュ動作について理解を深めることができるようにするためのものです。

4.3.8.2 L2 キャッシュの追い出し

C64x メモリ・アーキテクチャでは、L2 がラインを追い出すと、L2 に包括的保持されている L1D を含め、L1D をスヌープ・インバリデートします。C64x+ メモリ・アーキテクチャでは、L2 がラインを追い出すと、ラインがダーティの場合、L1D を調べることなく、ビクティムをライト・アウトします。L2 は、L1D ではラインをインバリデートしません。また、L2 がラインを追い出す場合、L1P のラインもインバリデートしません。結果的に、L1D も L1P も L2 には包含されません。

4.3.8.3 L1D ビクティム関連のポリシー

L1D ビクティム・ライトバックは、L2 ではライン割り当てをトリガしません。L1D ビクティムは、L2 をミスした場合、外部メモリへ直接ライトされます。

また、L1D ビクティム・ライトバックは、L2 にヒットした場合、L2 の LRU を更新しません。L2 のダーティ状態は、必要に応じて更新されます。

4.3.8.4 DMA/IDMA によるライト時の相互の影響

DMA または IDMA によるライトが L2 RAM に行われた場合、L2 の動作はデータが L1D にキャッシュされたかどうかによって異なります。C64x+ アーキテクチャと C64x アーキテクチャの動作は異なります。C64x アーキテクチャでは、スヌープ・インバリデート・コマンドは、L1P および L1D に送られます。C64x+ アーキテクチャでは、DMA/IDMA によるライトは L1P ではラインをインバリデートしません。DMA/IDMA によるライトは、アドレス範囲が L1D に存在すれば、スヌープ・ライト・コマンドを L1D へ送ります。存在しなければ、コマンドは送られません。

4.3.8.5 DMA/IDMA によるリード時の相互の影響

L2 メモリは、L1D のタグのシャドウ・コピーを保持しています。このシャドウ・コピーには、「ダーティ」と「有効」の両方の状態が含まれます。

DMA/IDMA が L2 RAM をリードする場合、L2 はシャドウ・タグを調べます。特定のアドレスが L1D で「有効」でかつ「ダーティ」とマークされている場合、L2 はそのアドレスへのスヌープ・リード・リクエストを L1D へ送ります。L1D はリクエストされたデータを用いて、応答します。

スヌープ・リードは L1D では有効なデータのままで、L2 へデータを追い出したり、ライトバックしたりしません。結果的に、L1D にアロケートされているバッファは、L1D にアロケートされたままです。そのため、CPU 上で実行されているアルゴリズムは、キャッシュ・ミス・ペナルティを課せられることなく、それ以降でバッファを補充することができます。

4.4 L2 キャッシュ・コントロール・レジスタ

C64x+ メガモジュールのメモリ・システムには、L2 キャッシュの動作を管理するレジスタ・セットがあります。このようなレジスタは、いくつかのカテゴリに分類されており、これ以降で説明します。

- **キャッシュ・サイズおよび動作モードの制御：**このカテゴリに分類されるレジスタは、キャッシュ・サイズを制御し、キャッシュがフリーズされているか正常に動作しているかも制御します。詳細については、4.4.2 項を参照してください。
- **ブロックまたはグローバル・コヒーレンス動作：**このカテゴリに分類されるレジスタを使用すると、手動でキャッシュからデータを転送することができます。
- **キャッシュ可能性の制御：**このカテゴリに分類されるレジスタは、一定の範囲のメモリを格納するために、キャッシュの許可を制御します。詳細については、4.4.4 項を参照してください。

4.4.1 メモリ・マップド L2 キャッシュ・コントロール・レジスタの概要

表 4-10 に、L2 キャッシュ・コントロール・レジスタを示します。

表 4-10. L2 キャッシュ・コントロール・レジスタ

アドレス	略称	レジスタの説明	参照先
0184 0000h	L2CFG	L2 コンフィギュレーション・レジスタ	4.4.2 項
0184 4000h	L2WBAR	L2 ライトバック・ベース・アドレス・レジスタ	4.4.3.1.1 項
0184 4004h	L2WWC	L2 ライトバック・ワード・カウント・レジスタ	4.4.3.1.2 項
0184 4010h	L2WIBAR	L2 ライトバック・インバリデート・ベース・アドレス・レジスタ	4.4.3.1.3 項
0184 4014h	L2WIWC	L2 ライトバック・インバリデート・ワード・カウント・レジスタ	4.4.3.1.4 項
0184 4018h	L2IBAR	L2 インバリデート・ベース・アドレス・レジスタ	4.4.3.1.5 項
0184 401Ch	L2IWC	L2 インバリデート・ワード・カウント・レジスタ	4.4.3.1.6 項
0184 5000h	L2WB	L2 ライトバック・レジスタ	4.4.3.2.1 項
0184 5004h	L2WBINV	L2 ライトバック・インバリデート・レジスタ	4.4.3.2.2 項
0184 5008h	L2INV	L2 インバリデート・レジスタ	4.4.3.2.3 項

4.4.2 L2 コンフィギュレーション・レジスタ（L2CFG）

L2CFG レジスタは L2 キャッシュの動作を制御します。L2CFG は、キャッシュとして機能する L2 メモリの量をセットし、L2 フリーズ・モードを制御し、L1D/L1P インバリデート・ビットを保持します。

L2 コンフィギュレーション・レジスタ（L2CFG）を図 4-2 に示し、表 4-11 で説明します。

図 4-2. L2 コンフィギュレーション・レジスタ（L2CFG）

31 28	27 24	23 20	19 16
Reserved	NUM MM	Reserved	MMID
R-0	R-config	R-0	R-config

15 10	9	8	7 4	3	2 0
Reserved	IP	ID	Reserved	L2CC	L2MODE
R-0	W-0	W-0	R-0	R/W-0	R/W-0

凡例：R/W = リード / ライト。R = リード専用。W = ライト専用。*-n* = リセット後の値。

表 4-11. L2 コンフィギュレーション・レジスタ（L2CFG）フィールドの説明

ビット	フィールド	値	説明
31-28	Reserved	0	予約。
27-24	NUM MM	0 ～ Fh	メガモジュールの数 - 1。複数の処理を行う環境で使用します。
23-20	Reserved	0	予約。
19-16	MMID	0 ～ Fh	メガモジュールの ID 番号が含まれます。複数のメガモジュールが存在する、複数の処理を行う環境で使用します。
15-10	Reserved	0	予約。
9	IP		L1P グローバル・インバリデート・ビット。下位互換性を確保するために用意されています。新規に開発するアプリケーションでは、L1PINV レジスタを使用してください（第 2 章を参照）。
		0	通常の L1P 動作。
		1	すべての L1P ラインはインバリデートされます。
8	ID		L1D グローバル・インバリデート・ビット。下位互換性を確保するために用意されています。新規に開発するアプリケーションでは、L1DINV レジスタを使用してください（第 3 章を参照）。
		0	通常の L1D 動作。
		1	すべての L1D ラインはインバリデートされます。
7-4	Reserved	0	予約。
3	L2CC		L2 フリーズ・モードを制御します。
		0	通常動作。
		1	L2 キャッシュはフリーズされます。
2-0	L2MODE	0 ～ 7h	L2 のキャッシュ・サイズを設定します。
		0h	L2 キャッシュはディスエーブル。
		1h	32K
		2h	64K
		3h	128K
		4h	256K
		5h	最大のキャッシュ。
		6h	最大のキャッシュ。
		7h	最大のキャッシュ。

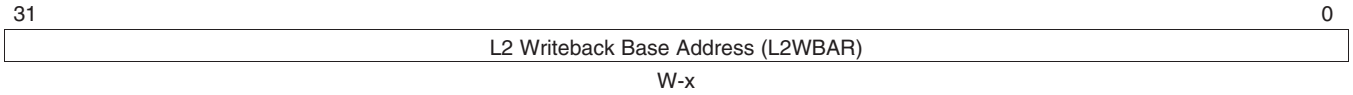
4.4.3 L2 キャッシュ・コヒーレンス・オペレーション・レジスタ

4.4.3.1 ブロック・コヒーレンス・オペレーション・レジスタ

4.4.3.1.1 L2 ライトバック・ベース・アドレス・レジスタ（L2WBAR）

L2 ライトバック・ベース・アドレス・レジスタ（L2WBAR）を図 4-3 に示し、表 4-12 で説明します。

図 4-3. L2 ライトバック・ベース・アドレス・レジスタ（L2WBAR）



凡例：W = ライト専用。*-x* = 値は不定。詳細については、各デバイスのデータ・マニュアルを参照してください。

表 4-12. L2 ライトバック・ベース・アドレス・レジスタ（L2WBAR）フィールドの説明

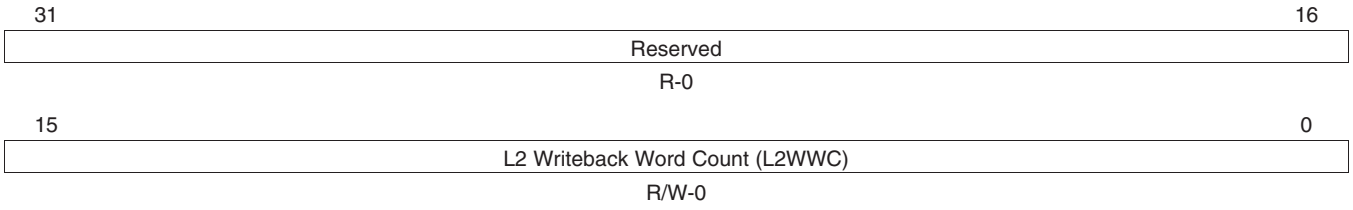
ビット	フィールド	値	説明
31-0	L2WBAR	0 ～ FFFF FFFFh	L2 ブロック・ライトバック動作を行うベース・アドレスを指定します。

4.4.3.1.2 L2 ライトバック・ワード・カウント・レジスタ（L2WWC）

L2 ライトバック・ワード・カウント・レジスタ（L2WWC）は、インバリデートされるブロック・サイズを指定します。サイズは 32 ビット・ワードで指定されています。

L2 ライトバック・ワード・カウント・レジスタ（L2WWC）を図 4-4 に示し、表 4-13 で説明します。

図 4-4. L2 ライトバック・ワード・カウント・レジスタ（L2WWC）



凡例：R/W = リード / ライト。R = リード専用。*-n* = リセット後の値。

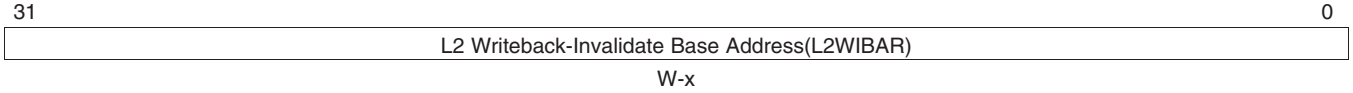
表 4-13. L2 ライトバック・ワード・カウント・レジスタ（L2WWC）フィールドの説明

ビット	フィールド	値	説明
31-16	Reserved	0	予約。
15-0	L2WWC	0 ～ FFFFh	ブロック・インバリデートを行うワード・カウント。

4.4.3.1.3 L2 ライトバック・インバリデート・ベース・アドレス・レジスタ（L2WIBAR）

L2 ライトバック・インバリデート・ベース・アドレス・レジスタ（L2WIBAR）を図 4-5 に示し、表 4-14 で説明します。

図 4-5. L2 ライトバック・インバリデート・ベース・アドレス・レジスタ（L2WIBAR）



凡例：W = ライト専用。-*x* = 値は不定。詳細については、各デバイスのデータ・マニュアルを参照してください。

表 4-14. L2 ライトバック・インバリデート・ベース・アドレス・レジスタ（L2WIBAR）フィールドの説明

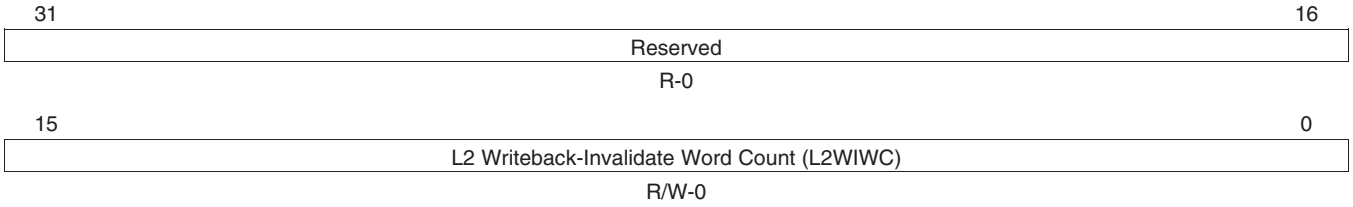
ビット	フィールド	値	説明
31-0	L2WIBAR	0 ～ FFFF FFFFh	L2 ブロック・ライトバック・インバリデート動作を行うベース・アドレスを指定します。

4.4.3.1.4 L2 ライトバック・インバリデート・ワード・カウント・レジスタ（L2WIWC）

L2 ライトバック・インバリデート・ワード・カウント・レジスタ（L2WIWC）は、インバリデートされるブロック・サイズを指定します。サイズは 32 ビット・ワードで指定されています。

L2 ライトバック・インバリデート・ワード・カウント・レジスタ（L2WIWC）を図 4-6 に示し、表 4-15 で説明します。

図 4-6. L2 ライトバック・インバリデート・ワード・カウント・レジスタ（L2WIWC）



凡例：R/W = リード / ライト。R = リード専用。-*n* = リセット後の値。

表 4-15. L2 ライトバック・インバリデート・ワード・カウント・レジスタ（L2WIWC）フィールドの説明

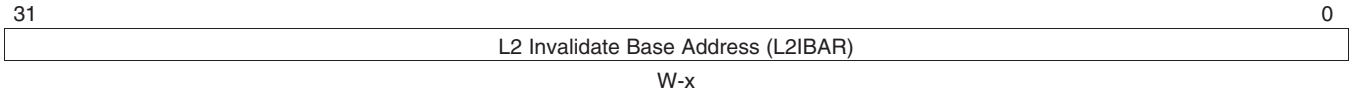
ビット	フィールド	値	説明
31-16	Reserved	0	予約。
15-0	L2WIWC	0 ～ FFFFh	ブロック・インバリデートを行うワード・カウント。

4.4.3.1.5 L2 インバリデート・ベース・アドレス・レジスタ（L2IBAR）

L2 インバリデート・ベース・アドレス・レジスタ（L2IBAR）は、インバリデートされるブロックのベース・アドレスを指定します。

L2 インバリデート・ベース・アドレス・レジスタ（L2IBAR）を図 4-7 に示し、表 4-16 で説明します。

図 4-7. L2 インバリデート・ベース・アドレス・レジスタ（L2IBAR）



凡例：W = ライト専用。-*x* = 値は不定。詳細については、各デバイスのデータ・マニュアルを参照してください。

表 4-16. L2 インバリデート・ベース・アドレス・レジスタ（L2IBAR）フィールドの説明

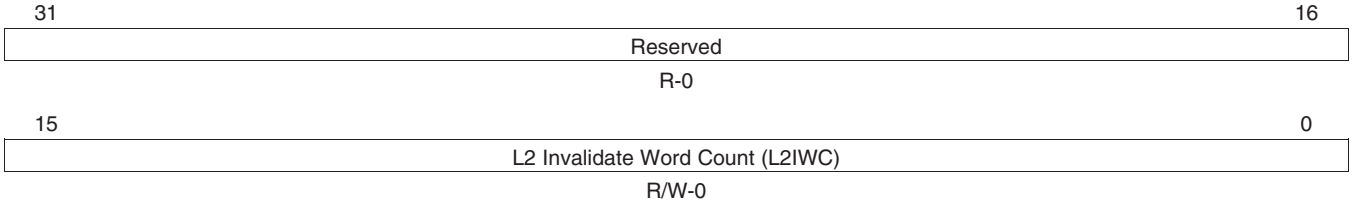
ビット	フィールド	値	説明
31-0	L2IBAR	0 ～ FFFF FFFFh	L2 ブロック・インバリデート動作を行うベース・アドレスを指定します。

4.4.3.1.6 L2 インバリデート・ワード・カウント・レジスタ（L2IWC）

L2 インバリデート・ワード・カウント・レジスタ（L2IWC）は、インバリデートされるブロック・サイズを指定します。サイズは 32 ビット・ワードで指定されています。

L2 インバリデート・ワード・カウント・レジスタ（L2IWC）を図 4-8 に示し、表 4-17 で説明します。

図 4-8. L2 インバリデート・ワード・カウント・レジスタ（L2IWC）



凡例：R/W = リード / ライト。R = リード専用。-*n* = リセット後の値。

表 4-17. L2 インバリデート・ワード・カウント・レジスタ（L2IWC）フィールドの説明

ビット	フィールド	値	説明
31-16	Reserved	0	予約。
15-0	L2IWC	0 ～ FFFFh	ブロック・インバリデートを行うワード・カウント。

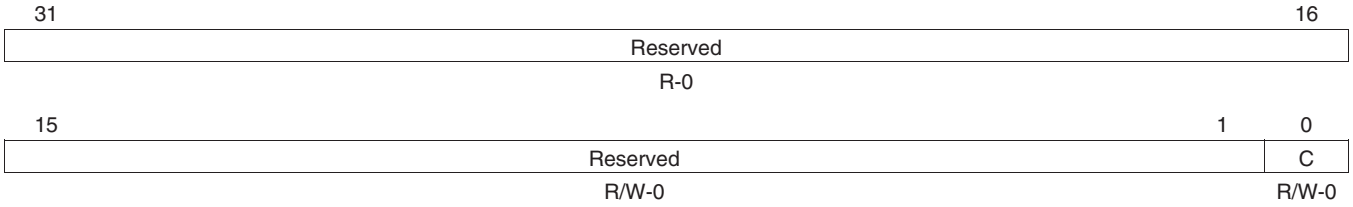
4.4.3.2 グローバル・コヒーレンス・オペレーション・レジスタ

4.4.3.2.1 L2 ライトバック・レジスタ（L2WB）

L2 ライトバック・レジスタ（L2WB）は、L2 キャッシュのグローバル・ライトバック動作を制御します。

L2 ライトバック・レジスタ（L2WB）を図 4-9 に示し、表 4-18 で説明します。

図 4-9. L2 ライトバック・レジスタ（L2WB）



凡例：R/W = リード / ライト。R = リード専用。-*n* = リセット後の値。

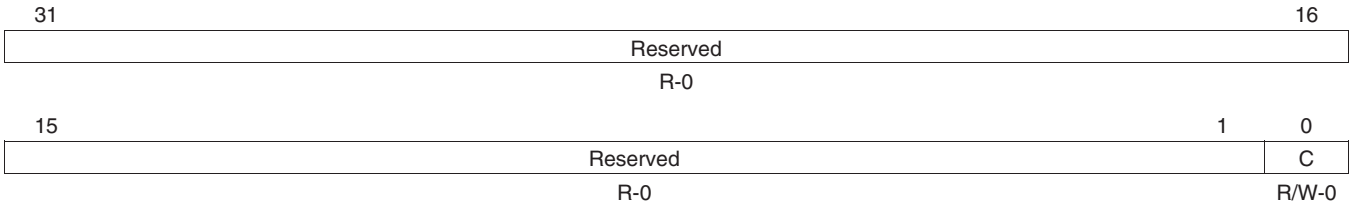
表 4-18. L2 ライトバック・レジスタ（L2WB）フィールドの説明

ビット	フィールド	値	説明
31-1	Reserved	0	予約。
0	C		L2 キャッシュのグローバル・ライトバック動作を制御します（4.3.6.1 項を参照）。
		0	通常動作。
		1	ダーティ L2 キャッシュ・ラインがライトバックされます。

4.4.3.2.2 L2 ライトバック・インバリデート・レジスタ（L2WBINV）

L2 ライトバック・インバリデート・レジスタ（L2WBINV）は、L2 キャッシュのライトバック・インバリデート動作を制御します。L2 ライトバック・インバリデート・レジスタ（L2WBINV）を図 4-10 に示し、表 4-19 で説明します。

図 4-10. L2 ライトバック・インバリデート・レジスタ（L2WBINV）



凡例：R/W = リード / ライト。R = リード専用。-*n* = リセット後の値。

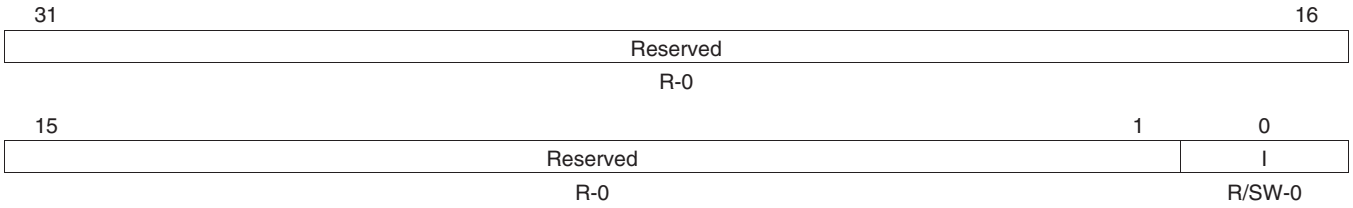
表 4-19. L2 ライトバック・インバリデート・レジスタ（L2WBINV）フィールドの説明

ビット	フィールド	値	説明
31-1	Reserved	0	予約。
0	C		L2 キャッシュのライトバック・インバリデート動作を制御します（4.3.6.1 項を参照）。
		0	通常の L2 動作。
		1	ダーティ L2 キャッシュ・ラインがライトバックされます。すべての L2 キャッシュ・ラインがインバリデートされます。

4.4.3.2.3 L2 インバリデート・レジスタ（L2INV）

L2 インバリデート・レジスタ（L2INV）は、L2 キャッシュのグローバル・インバリデートを制御します。L2INV を図 4-11 に示し、表 4-20 で説明します。

図 4-11. L2 インバリデート・レジスタ（L2INV）



凡例：R/W = リード / ライト。R = リード専用。-*n* = リセット後の値。R/SW = スーパーバイザのみがリード / ライト可能。

表 4-20. L2 インバリデート・レジスタ（L2INV）フィールドの説明

ビット	フィールド	値	説明
31-1	Reserved	0	予約。
0	I		L2 キャッシュのグローバル・インバリデートを制御します。
		0	通常動作。
		1	すべての L2 キャッシュ・ラインがインバリデートされます。

4.4.4 メモリ・アトリビュート・レジスタ（MAR*n*）

L2 メモリは、レジスタ・セットから構成されており、（メガモジュールへの）外部メモリ空間のキャッシュ可能性を指定します。MAR（メモリ・アトリビュート・レジスタ）と呼ばれるレジスタについて、表 4-21 で説明します。MAR レジスタはスーパーバイザ・コードでのみライト可能です。

表 4-21 に、メモリ・アトリビュート・メモリ・マップド・コントロール・レジスタを示します。

表 4-21. メモリ・アトリビュート・レジスタ

アドレス	略称	レジスタの説明	アトリビュート指定対象アドレス
0184 8000h	MAR0	メモリ・アトリビュート・レジスタ 0	ローカルな L2 RAM（固定）
0184 8004h	MAR1	メモリ・アトリビュート・レジスタ 1	0100 0000h ～ 01FF FFFFh
0184 8008h	MAR2	メモリ・アトリビュート・レジスタ 2	0200 0000h ～ 02FF FFFFh
0184 800Ch	MAR3	メモリ・アトリビュート・レジスタ 3	0300 0000h ～ 03FF FFFFh
0184 8010h	MAR4	メモリ・アトリビュート・レジスタ 4	0400 0000h ～ 04FF FFFFh
0184 8014h	MAR5	メモリ・アトリビュート・レジスタ 5	0500 0000h ～ 05FF FFFFh
0184 8018h	MAR6	メモリ・アトリビュート・レジスタ 6	0600 0000h ～ 06FF FFFFh
0184 801Ch	MAR7	メモリ・アトリビュート・レジスタ 7	0700 0000h ～ 07FF FFFFh
0184 8020h	MAR8	メモリ・アトリビュート・レジスタ 8	0800 0000h ～ 08FF FFFFh
0184 8024h	MAR9	メモリ・アトリビュート・レジスタ 9	0900 0000h ～ 09FF FFFFh
0184 8028h	MAR10	メモリ・アトリビュート・レジスタ 10	0A00 0000h ～ 0AFF FFFFh
0184 802Ch	MAR11	メモリ・アトリビュート・レジスタ 11	0B00 0000h ～ 0BFF FFFFh
0184 8030h	MAR12	メモリ・アトリビュート・レジスタ 12	0C00 0000h ～ 0CFF FFFFh
0184 8034h	MAR13	メモリ・アトリビュート・レジスタ 13	0D00 0000h ～ 0DFF FFFFh
0184 8038h	MAR14	メモリ・アトリビュート・レジスタ 14	0E00 0000h ～ 0EFF FFFFh
0184 803Ch	MAR15	メモリ・アトリビュート・レジスタ 15	0F00 0000h ～ 0FFF FFFFh
0184 8040h	MAR16	メモリ・アトリビュート・レジスタ 16	1000 0000h ～ 10FF FFFFh
0184 8044h	MAR17	メモリ・アトリビュート・レジスタ 17	1100 0000h ～ 11FF FFFFh
0184 8048h	MAR18	メモリ・アトリビュート・レジスタ 18	1200 0000h ～ 12FF FFFFh
0184 804Ch	MAR19	メモリ・アトリビュート・レジスタ 19	1300 0000h ～ 13FF FFFFh
0184 8050h	MAR20	メモリ・アトリビュート・レジスタ 20	1400 0000h ～ 14FF FFFFh
0184 8054h	MAR21	メモリ・アトリビュート・レジスタ 21	1500 0000h ～ 15FF FFFFh
0184 8058h	MAR22	メモリ・アトリビュート・レジスタ 22	1600 0000h ～ 16FF FFFFh
0184 805Ch	MAR23	メモリ・アトリビュート・レジスタ 23	1700 0000h ～ 17FF FFFFh
0184 8060h	MAR24	メモリ・アトリビュート・レジスタ 24	1800 0000h ～ 18FF FFFFh
0184 8064h	MAR25	メモリ・アトリビュート・レジスタ 25	1900 0000h ～ 19FF FFFFh
0184 8068h	MAR26	メモリ・アトリビュート・レジスタ 26	1A00 0000h ～ 1AFF FFFFh
0184 806Ch	MAR27	メモリ・アトリビュート・レジスタ 27	1B00 0000h ～ 1BFF FFFFh
0184 8070h	MAR28	メモリ・アトリビュート・レジスタ 28	1C00 0000h ～ 1CFF FFFFh
0184 8074h	MAR29	メモリ・アトリビュート・レジスタ 29	1D00 0000h ～ 1DFF FFFFh
0184 8078h	MAR30	メモリ・アトリビュート・レジスタ 30	1E00 0000h ～ 1EFF FFFFh
0184 807Ch	MAR31	メモリ・アトリビュート・レジスタ 31	1F00 0000h ～ 1FFF FFFFh
0184 8080h	MAR32	メモリ・アトリビュート・レジスタ 32	2000 0000h ～ 20FF FFFFh
0184 8084h	MAR33	メモリ・アトリビュート・レジスタ 33	2100 0000h ～ 21FF FFFFh
0184 8088h	MAR34	メモリ・アトリビュート・レジスタ 34	2200 0000h ～ 22FF FFFFh

表 4-21. メモリ・アトリビュート・レジスタ（続き）

アドレス	略称	レジスタの説明	アトリビュート指定対象アドレス
0184 808Ch	MAR35	メモリ・アトリビュート・レジスタ 35	2300 0000h ～ 23FF FFFFh
0184 8090h	MAR36	メモリ・アトリビュート・レジスタ 36	2400 0000h ～ 24FF FFFFh
0184 8094h	MAR37	メモリ・アトリビュート・レジスタ 37	2500 0000h ～ 25FF FFFFh
0184 8098h	MAR38	メモリ・アトリビュート・レジスタ 38	2600 0000h ～ 26FF FFFFh
0184 809Ch	MAR39	メモリ・アトリビュート・レジスタ 39	2700 0000h ～ 27FF FFFFh
0184 80A0h	MAR40	メモリ・アトリビュート・レジスタ 40	2800 0000h ～ 28FF FFFFh
0184 80A4h	MAR41	メモリ・アトリビュート・レジスタ 41	2900 0000h ～ 29FF FFFFh
0184 80A8h	MAR42	メモリ・アトリビュート・レジスタ 42	2A00 0000h ～ 2AFF FFFFh
0184 80ACh	MAR43	メモリ・アトリビュート・レジスタ 43	2B00 0000h ～ 2BFF FFFFh
0184 80B0h	MAR44	メモリ・アトリビュート・レジスタ 44	2C00 0000h ～ 2CFF FFFFh
0184 80B4h	MAR45	メモリ・アトリビュート・レジスタ 45	2D00 0000h ～ 2DFF FFFFh
0184 80B8h	MAR46	メモリ・アトリビュート・レジスタ 46	2E00 0000h ～ 2EFF FFFFh
0184 80BCh	MAR47	メモリ・アトリビュート・レジスタ 47	2F00 0000h ～ 2FFF FFFFh
0184 80C0h	MAR48	メモリ・アトリビュート・レジスタ 48	3000 0000h ～ 30FF FFFFh
0184 80C4h	MAR49	メモリ・アトリビュート・レジスタ 49	3100 0000h ～ 31FF FFFFh
0184 80C8h	MAR50	メモリ・アトリビュート・レジスタ 50	3200 0000h ～ 32FF FFFFh
0184 80CCh	MAR51	メモリ・アトリビュート・レジスタ 51	3300 0000h ～ 33FF FFFFh
0184 80D0h	MAR52	メモリ・アトリビュート・レジスタ 52	3400 0000h ～ 34FF FFFFh
0184 80D4h	MAR53	メモリ・アトリビュート・レジスタ 53	3500 0000h ～ 35FF FFFFh
0184 80D8h	MAR54	メモリ・アトリビュート・レジスタ 54	3600 0000h ～ 36FF FFFFh
0184 80DCh	MAR55	メモリ・アトリビュート・レジスタ 55	3700 0000h ～ 37FF FFFFh
0184 80E0h	MAR56	メモリ・アトリビュート・レジスタ 56	3800 0000h ～ 38FF FFFFh
0184 80E4h	MAR57	メモリ・アトリビュート・レジスタ 57	3900 0000h ～ 39FF FFFFh
0184 80E8h	MAR58	メモリ・アトリビュート・レジスタ 58	3A00 0000h ～ 3AFF FFFFh
0184 80ECh	MAR59	メモリ・アトリビュート・レジスタ 59	3B00 0000h ～ 3BFF FFFFh
0184 80F0h	MAR60	メモリ・アトリビュート・レジスタ 60	3C00 0000h ～ 3CFF FFFFh
0184 80F4h	MAR61	メモリ・アトリビュート・レジスタ 61	3D00 0000h ～ 3DFF FFFFh
0184 80F8h	MAR62	メモリ・アトリビュート・レジスタ 62	3E00 0000h ～ 3EFF FFFFh
0184 80FCh	MAR63	メモリ・アトリビュート・レジスタ 63	3F00 0000h ～ 3FFF FFFFh
0184 8100h	MAR64	メモリ・アトリビュート・レジスタ 64	4000 0000h ～ 40FF FFFFh
0184 8104h	MAR65	メモリ・アトリビュート・レジスタ 65	4100 0000h ～ 41FF FFFFh
0184 8108h	MAR66	メモリ・アトリビュート・レジスタ 66	4200 0000h ～ 42FF FFFFh
0184 810Ch	MAR67	メモリ・アトリビュート・レジスタ 67	4300 0000h ～ 43FF FFFFh
0184 8110h	MAR68	メモリ・アトリビュート・レジスタ 68	4400 0000h ～ 44FF FFFFh
0184 8114h	MAR69	メモリ・アトリビュート・レジスタ 69	4500 0000h ～ 45FF FFFFh
0184 8118h	MAR70	メモリ・アトリビュート・レジスタ 70	4600 0000h ～ 46FF FFFFh
0184 811Ch	MAR71	メモリ・アトリビュート・レジスタ 71	4700 0000h ～ 47FF FFFFh
0184 8120h	MAR72	メモリ・アトリビュート・レジスタ 72	4800 0000h ～ 48FF FFFFh
0184 8124h	MAR73	メモリ・アトリビュート・レジスタ 73	4900 0000h ～ 49FF FFFFh
0184 8128h	MAR74	メモリ・アトリビュート・レジスタ 74	4A00 0000h ～ 4AFF FFFFh
0184 812Ch	MAR75	メモリ・アトリビュート・レジスタ 75	4B00 0000h ～ 4BFF FFFFh

表 4-21. メモリ・アトリビュート・レジスタ（続き）

アドレス	略称	レジスタの説明	アトリビュート指定対象アドレス
0184 8130h	MAR76	メモリ・アトリビュート・レジスタ 76	4C00 0000h ～ 4CFF FFFFh
0184 8134h	MAR77	メモリ・アトリビュート・レジスタ 77	4D00 0000h ～ 4DFF FFFFh
0184 8138h	MAR78	メモリ・アトリビュート・レジスタ 78	4E00 0000h ～ 4EFF FFFFh
0184 813Ch	MAR79	メモリ・アトリビュート・レジスタ 79	4F00 0000h ～ 4FFF FFFFh
0184 8140h	MAR80	メモリ・アトリビュート・レジスタ 80	5000 0000h ～ 50FF FFFFh
0184 8144h	MAR81	メモリ・アトリビュート・レジスタ 81	5100 0000h ～ 51FF FFFFh
0184 8148h	MAR82	メモリ・アトリビュート・レジスタ 82	5200 0000h ～ 52FF FFFFh
0184 814Ch	MAR83	メモリ・アトリビュート・レジスタ 83	5300 0000h ～ 53FF FFFFh
0184 8150h	MAR84	メモリ・アトリビュート・レジスタ 84	5400 0000h ～ 54FF FFFFh
0184 8154h	MAR85	メモリ・アトリビュート・レジスタ 85	5500 0000h ～ 55FF FFFFh
0184 8158h	MAR86	メモリ・アトリビュート・レジスタ 86	5600 0000h ～ 56FF FFFFh
0184 815Ch	MAR87	メモリ・アトリビュート・レジスタ 87	5700 0000h ～ 57FF FFFFh
0184 8160h	MAR88	メモリ・アトリビュート・レジスタ 88	5800 0000h ～ 58FF FFFFh
0184 8164h	MAR89	メモリ・アトリビュート・レジスタ 89	5900 0000h ～ 59FF FFFFh
0184 8168h	MAR90	メモリ・アトリビュート・レジスタ 90	5A00 0000h ～ 5AFF FFFFh
0184 816Ch	MAR91	メモリ・アトリビュート・レジスタ 91	5B00 0000h ～ 5BFF FFFFh
0184 8170h	MAR92	メモリ・アトリビュート・レジスタ 92	5C00 0000h ～ 5CFF FFFFh
0184 8174h	MAR93	メモリ・アトリビュート・レジスタ 93	5D00 0000h ～ 5DFF FFFFh
0184 8178h	MAR94	メモリ・アトリビュート・レジスタ 94	5E00 0000h ～ 5EFF FFFFh
0184 817Ch	MAR95	メモリ・アトリビュート・レジスタ 95	5F00 0000h ～ 5FFF FFFFh
0184 8180h	MAR96	メモリ・アトリビュート・レジスタ 96	6000 0000h ～ 60FF FFFFh
0184 8184h	MAR97	メモリ・アトリビュート・レジスタ 97	6100 0000h ～ 61FF FFFFh
0184 8188h	MAR98	メモリ・アトリビュート・レジスタ 98	6200 0000h ～ 62FF FFFFh
0184 818Ch	MAR99	メモリ・アトリビュート・レジスタ 99	6300 0000h ～ 63FF FFFFh
0184 8190h	MAR100	メモリ・アトリビュート・レジスタ 100	6400 0000h ～ 64FF FFFFh
0184 8194h	MAR101	メモリ・アトリビュート・レジスタ 101	6500 0000h ～ 65FF FFFFh
0184 8198h	MAR102	メモリ・アトリビュート・レジスタ 102	6600 0000h ～ 66FF FFFFh
0184 819Ch	MAR103	メモリ・アトリビュート・レジスタ 103	6700 0000h ～ 67FF FFFFh
0184 81A0h	MAR104	メモリ・アトリビュート・レジスタ 104	6800 0000h ～ 68FF FFFFh
0184 81A4h	MAR105	メモリ・アトリビュート・レジスタ 105	6900 0000h ～ 69FF FFFFh
0184 81A8h	MAR106	メモリ・アトリビュート・レジスタ 106	6A00 0000h ～ 6AFF FFFFh
0184 81ACh	MAR107	メモリ・アトリビュート・レジスタ 107	6B00 0000h ～ 6BFF FFFFh
0184 81B0h	MAR108	メモリ・アトリビュート・レジスタ 108	6C00 0000h ～ 6CFF FFFFh
0184 81B4h	MAR109	メモリ・アトリビュート・レジスタ 109	6D00 0000h ～ 6DFF FFFFh
0184 81B8h	MAR110	メモリ・アトリビュート・レジスタ 110	6E00 0000h ～ 6EFF FFFFh
0184 81BCh	MAR111	メモリ・アトリビュート・レジスタ 111	6F00 0000h ～ 6FFF FFFFh
0184 81C0h	MAR112	メモリ・アトリビュート・レジスタ 112	7000 0000h ～ 70FF FFFFh
0184 81C4h	MAR113	メモリ・アトリビュート・レジスタ 113	7100 0000h ～ 71FF FFFFh
0184 81C8h	MAR114	メモリ・アトリビュート・レジスタ 114	7200 0000h ～ 72FF FFFFh
0184 81CCh	MAR115	メモリ・アトリビュート・レジスタ 115	7300 0000h ～ 73FF FFFFh
0184 81D0h	MAR116	メモリ・アトリビュート・レジスタ 116	7400 0000h ～ 74FF FFFFh

表 4-21. メモリ・アトリビュート・レジスタ（続き）

アドレス	略称	レジスタの説明	アトリビュート指定対象アドレス
0184 81D4h	MAR117	メモリ・アトリビュート・レジスタ 117	7500 0000h ～ 75FF FFFFh
0184 81D8h	MAR118	メモリ・アトリビュート・レジスタ 118	76000000h ～ 76FFFFFFh
0184 81DCh	MAR119	メモリ・アトリビュート・レジスタ 119	7700 0000h ～ 77FF FFFFh
0184 81E0h	MAR120	メモリ・アトリビュート・レジスタ 120	7800 0000h ～ 78FF FFFFh
0184 81E4h	MAR121	メモリ・アトリビュート・レジスタ 121	7900 0000h ～ 79FF FFFFh
0184 81E8h	MAR122	メモリ・アトリビュート・レジスタ 122	7A00 0000h ～ 7AFF FFFFh
0184 81ECh	MAR123	メモリ・アトリビュート・レジスタ 123	7B00 0000h ～ 7BFF FFFFh
0184 81F0h	MAR124	メモリ・アトリビュート・レジスタ 124	7C00 0000h ～ 7CFF FFFFh
0184 81F4h	MAR125	メモリ・アトリビュート・レジスタ 125	7D00 0000h ～ 7DFF FFFFh
0184 81F8h	MAR126	メモリ・アトリビュート・レジスタ 126	7E00 0000h ～ 7EFF FFFFh
0184 81FCh	MAR127	メモリ・アトリビュート・レジスタ 127	7F00 0000h ～ 7FFF FFFFh
0184 8200h	MAR128	メモリ・アトリビュート・レジスタ 128	8000 0000h ～ 80FF FFFFh
0184 8204h	MAR129	メモリ・アトリビュート・レジスタ 129	8100 0000h ～ 81FF FFFFh
0184 8208h	MAR130	メモリ・アトリビュート・レジスタ 130	8200 0000h ～ 82FF FFFFh
0184 820Ch	MAR131	メモリ・アトリビュート・レジスタ 131	8300 0000h ～ 83FF FFFFh
0184 8210h	MAR132	メモリ・アトリビュート・レジスタ 132	8400 0000h ～ 84FF FFFFh
0184 8214h	MAR133	メモリ・アトリビュート・レジスタ 133	8500 0000h ～ 85FF FFFFh
0184 8218h	MAR134	メモリ・アトリビュート・レジスタ 134	8600 0000h ～ 86FF FFFFh
0184 821Ch	MAR135	メモリ・アトリビュート・レジスタ 135	8700 0000h ～ 87FF FFFFh
0184 8220h	MAR136	メモリ・アトリビュート・レジスタ 136	8800 0000h ～ 88FF FFFFh
0184 8224h	MAR137	メモリ・アトリビュート・レジスタ 137	8900 0000h ～ 89FF FFFFh
0184 8228h	MAR138	メモリ・アトリビュート・レジスタ 138	8A00 0000h ～ 8AFF FFFFh
0184 822Ch	MAR139	メモリ・アトリビュート・レジスタ 139	8B00 0000h ～ 8BFF FFFFh
0184 8230h	MAR140	メモリ・アトリビュート・レジスタ 140	8C00 0000h ～ 8CFF FFFFh
0184 8234h	MAR141	メモリ・アトリビュート・レジスタ 141	8D00 0000h ～ 8DFF FFFFh
0184 8238h	MAR142	メモリ・アトリビュート・レジスタ 142	8E00 0000h ～ 8EFF FFFFh
0184 823Ch	MAR143	メモリ・アトリビュート・レジスタ 143	8F00 0000h ～ 8FFF FFFFh
0184 8240h	MAR144	メモリ・アトリビュート・レジスタ 144	9000 0000h ～ 90FF FFFFh
0184 8244h	MAR145	メモリ・アトリビュート・レジスタ 145	9100 0000h ～ 91FF FFFFh
0184 8248h	MAR146	メモリ・アトリビュート・レジスタ 146	9200 0000h ～ 92FF FFFFh
0184 824Ch	MAR147	メモリ・アトリビュート・レジスタ 147	9300 0000h ～ 93FF FFFFh
0184 8250h	MAR148	メモリ・アトリビュート・レジスタ 148	9400 0000h ～ 94FF FFFFh
0184 8254h	MAR149	メモリ・アトリビュート・レジスタ 149	9500 0000h ～ 95FF FFFFh
0184 8258h	MAR150	メモリ・アトリビュート・レジスタ 150	9600 0000h ～ 96FF FFFFh
0184 825Ch	MAR151	メモリ・アトリビュート・レジスタ 151	9700 0000h ～ 97FF FFFFh
0184 8260h	MAR152	メモリ・アトリビュート・レジスタ 152	9800 0000h ～ 98FF FFFFh
0184 8264h	MAR153	メモリ・アトリビュート・レジスタ 153	990 00000h ～ 99FF FFFFh
0184 8268h	MAR154	メモリ・アトリビュート・レジスタ 154	9A00 0000h ～ 9AFF FFFFh
0184 826Ch	MAR155	メモリ・アトリビュート・レジスタ 155	9B00 0000h ～ 9BFF FFFFh
0184 821Ch	MAR156	メモリ・アトリビュート・レジスタ 156	9C00 0000h ～ 9CFF FFFFh
0184 8274h	MAR157	メモリ・アトリビュート・レジスタ 157	9D00 0000h ～ 9DFF FFFFh

表 4-21. メモリ・アトリビュート・レジスタ（続き）

アドレス	略称	レジスタの説明	アトリビュート指定対象アドレス
0184 8278h	MAR158	メモリ・アトリビュート・レジスタ 158	9E00 0000h ～ 9EFF FFFFh
0184 827Ch	MAR159	メモリ・アトリビュート・レジスタ 159	9F00 0000h ～ 9FFF FFFFh
0184 8280h	MAR160	メモリ・アトリビュート・レジスタ 160	A000 0000h ～ A0FF FFFFh
0184 8284h	MAR161	メモリ・アトリビュート・レジスタ 161	A100 0000h ～ A1FF FFFFh
0184 8288h	MAR162	メモリ・アトリビュート・レジスタ 162	A200 0000h ～ A2FF FFFFh
0184 828Ch	MAR163	メモリ・アトリビュート・レジスタ 163	A300 0000h ～ A3FF FFFFh
0184 8290h	MAR164	メモリ・アトリビュート・レジスタ 164	A400 0000h ～ A4FF FFFFh
0184 8294h	MAR165	メモリ・アトリビュート・レジスタ 165	A500 0000h ～ A5FF FFFFh
0184 8298h	MAR166	メモリ・アトリビュート・レジスタ 166	A600 0000h ～ A6FF FFFFh
0184 829Ch	MAR167	メモリ・アトリビュート・レジスタ 167	A700 0000h ～ A7FF FFFFh
0184 82A0h	MAR168	メモリ・アトリビュート・レジスタ 168	A800 0000h ～ A8FF FFFFh
0184 82A4h	MAR169	メモリ・アトリビュート・レジスタ 169	A900 0000h ～ A9FF FFFFh
0184 82A8h	MAR170	メモリ・アトリビュート・レジスタ 170	AA00 0000h ～ AAFF FFFFh
0184 82ACh	MAR171	メモリ・アトリビュート・レジスタ 171	AB00 0000h ～ ABFF FFFFh
0184 82B0h	MAR172	メモリ・アトリビュート・レジスタ 172	AC00 0000h ～ ACFF FFFFh
0184 82B4h	MAR173	メモリ・アトリビュート・レジスタ 173	AD00 0000h ～ ADFF FFFFh
0184 82B8h	MAR174	メモリ・アトリビュート・レジスタ 174	AE00 0000h ～ AEFF FFFFh
0184 82BCh	MAR175	メモリ・アトリビュート・レジスタ 175	AF00 0000h ～ AFFF FFFFh
0184 82C0h	MAR176	メモリ・アトリビュート・レジスタ 176	B000 0000h ～ B0FF FFFFh
0184 82C4h	MAR177	メモリ・アトリビュート・レジスタ 177	B100 0000h ～ B1FF FFFFh
0184 82C8h	MAR178	メモリ・アトリビュート・レジスタ 178	B20 00000h ～ B2FF FFFFh
0184 82CCh	MAR179	メモリ・アトリビュート・レジスタ 179	B300 0000h ～ B3FF FFFFh
0184 82D0h	MAR180	メモリ・アトリビュート・レジスタ 180	B400 0000h ～ B4FF FFFFh
0184 82D4h	MAR181	メモリ・アトリビュート・レジスタ 181	B500 0000h ～ B5FF FFFFh
0184 82D8h	MAR182	メモリ・アトリビュート・レジスタ 182	B600 0000h ～ B6FF FFFFh
0184 82DCh	MAR183	メモリ・アトリビュート・レジスタ 183	B700 0000h ～ B7FF FFFFh
0184 82E0h	MAR184	メモリ・アトリビュート・レジスタ 184	B800 0000h ～ B8FF FFFFh
0184 82E4h	MAR185	メモリ・アトリビュート・レジスタ 185	B900 0000h ～ B9FF FFFFh
0184 82E8h	MAR186	メモリ・アトリビュート・レジスタ 186	BA00 0000h ～ BAFF FFFFh
0184 82ECh	MAR187	メモリ・アトリビュート・レジスタ 187	BB00 0000h ～ BBFF FFFFh
0184 82F0h	MAR188	メモリ・アトリビュート・レジスタ 188	BC00 0000h ～ BCFF FFFFh
0184 82F4h	MAR189	メモリ・アトリビュート・レジスタ 189	BD00 0000h ～ BDFF FFFFh
0184 82F8h	MAR190	メモリ・アトリビュート・レジスタ 190	BE00 0000h ～ BEFF FFFFh
0184 82FCh	MAR191	メモリ・アトリビュート・レジスタ 191	BF00 0000h ～ BFFF FFFFh
0184 8300h	MAR192	メモリ・アトリビュート・レジスタ 192	C000 0000h ～ C0FF FFFFh
0184 8304h	MAR193	メモリ・アトリビュート・レジスタ 193	C100 0000h ～ C1FF FFFFh
0184 8308h	MAR194	メモリ・アトリビュート・レジスタ 194	C200 0000h ～ C2FF FFFFh
0184 830Ch	MAR195	メモリ・アトリビュート・レジスタ 195	C300 0000h ～ C3FF FFFFh
0184 8310h	MAR196	メモリ・アトリビュート・レジスタ 196	C400 0000h ～ C4FF FFFFh
0184 8314h	MAR197	メモリ・アトリビュート・レジスタ 197	C500 0000h ～ C5FF FFFFh
0184 8318h	MAR198	メモリ・アトリビュート・レジスタ 198	C600 0000h ～ C6FF FFFFh

表 4-21. メモリ・アトリビュート・レジスタ（続き）

アドレス	略称	レジスタの説明	アトリビュート指定対象アドレス
0184 831Ch	MAR199	メモリ・アトリビュート・レジスタ 199	C700 0000h ～ C7FF FFFFh
0184 8320h	MAR200	メモリ・アトリビュート・レジスタ 200	C800 0000h ～ C8FF FFFFh
0184 8324h	MAR201	メモリ・アトリビュート・レジスタ 201	C900 0000h ～ C9FF FFFFh
0184 8328h	MAR202	メモリ・アトリビュート・レジスタ 202	CA00 0000h ～ CAFF FFFFh
0184 832Ch	MAR203	メモリ・アトリビュート・レジスタ 203	CB00 0000h ～ CBFF FFFFh
0184 8330h	MAR204	メモリ・アトリビュート・レジスタ 204	CC00 0000h ～ CCFF FFFFh
0184 8334h	MAR205	メモリ・アトリビュート・レジスタ 205	CD00 0000h ～ CDFF FFFFh
0184 8338h	MAR206	メモリ・アトリビュート・レジスタ 206	CE00 0000h ～ CEFF FFFFh
0184 833Ch	MAR207	メモリ・アトリビュート・レジスタ 207	CF00 0000h ～ CFFF FFFFh
0184 8340h	MAR208	メモリ・アトリビュート・レジスタ 208	D000 0000h ～ D0FF FFFFh
0184 8344h	MAR209	メモリ・アトリビュート・レジスタ 209	D100 0000h ～ D1FF FFFFh
0184 8348h	MAR210	メモリ・アトリビュート・レジスタ 210	D200 0000h ～ D2FF FFFFh
0184 834Ch	MAR211	メモリ・アトリビュート・レジスタ 211	D300 0000h ～ D3FF FFFFh
0184 8350h	MAR212	メモリ・アトリビュート・レジスタ 212	D400 0000h ～ D4FF FFFFh
0184 8354h	MAR213	メモリ・アトリビュート・レジスタ 213	D500 0000h ～ D5FF FFFFh
0184 8358h	MAR214	メモリ・アトリビュート・レジスタ 214	D600 0000h ～ D6FF FFFFh
0184 835Ch	MAR215	メモリ・アトリビュート・レジスタ 215	D700 0000h ～ D7FF FFFFh
0184 8360h	MAR216	メモリ・アトリビュート・レジスタ 216	D800 0000h ～ D8FF FFFFh
0184 8364h	MAR217	メモリ・アトリビュート・レジスタ 217	D900 0000h ～ D9FF FFFFh
0184 8368h	MAR218	メモリ・アトリビュート・レジスタ 218	DA00 0000h ～ DAFF FFFFh
0184 836Ch	MAR219	メモリ・アトリビュート・レジスタ 219	DB00 0000h ～ DBFF FFFFh
0184 8370h	MAR220	メモリ・アトリビュート・レジスタ 220	DC00 0000h ～ DCFF FFFFh
0184 8374h	MAR221	メモリ・アトリビュート・レジスタ 221	DD00 0000h ～ DDFF FFFFh
0184 8378h	MAR222	メモリ・アトリビュート・レジスタ 222	DE00 0000h ～ DEFF FFFFh
0184 837Ch	MAR223	メモリ・アトリビュート・レジスタ 223	DF00 0000h ～ DFFF FFFFh
0184 8380h	MAR224	メモリ・アトリビュート・レジスタ 224	E000 0000h ～ E0FF FFFFh
0184 8384h	MAR225	メモリ・アトリビュート・レジスタ 225	E10 00000h ～ E1FF FFFFh
0184 8388h	MAR226	メモリ・アトリビュート・レジスタ 226	E200 0000h ～ E2FF FFFFh
0184 838Ch	MAR227	メモリ・アトリビュート・レジスタ 227	E300 0000h ～ E3FFF FFFh
0184 8390h	MAR228	メモリ・アトリビュート・レジスタ 228	E400 0000h ～ E4FF FFFFh
0184 8394h	MAR229	メモリ・アトリビュート・レジスタ 229	E500 0000h ～ E5FF FFFFh
0184 8398h	MAR230	メモリ・アトリビュート・レジスタ 230	E600 0000h ～ E6FF FFFFh
0184 839Ch	MAR231	メモリ・アトリビュート・レジスタ 231	E700 0000h ～ E7FF FFFFh
0184 83A0h	MAR232	メモリ・アトリビュート・レジスタ 232	E800 0000h ～ E8FF FFFFh
0184 83A4h	MAR233	メモリ・アトリビュート・レジスタ 233	E900 0000h ～ E9FF FFFFh
0184 83A8h	MAR234	メモリ・アトリビュート・レジスタ 234	EA00 0000h ～ EAFF FFFFh
0184 83ACh	MAR235	メモリ・アトリビュート・レジスタ 235	EB00 0000h ～ EBFF FFFFh
0184 83B0h	MAR236	メモリ・アトリビュート・レジスタ 236	EC00 0000h ～ ECFF FFFFh
0184 83B4h	MAR237	メモリ・アトリビュート・レジスタ 237	ED00 0000h ～ EDFF FFFFh
0184 83B8h	MAR238	メモリ・アトリビュート・レジスタ 238	EE00 0000h ～ EEFF FFFFh
0184 83BCh	MAR239	メモリ・アトリビュート・レジスタ 239	EF00 0000h ～ EFFF FFFFh

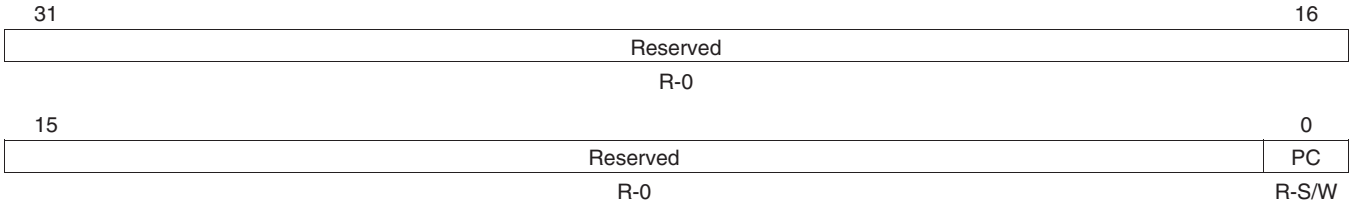
表 4-21. メモリ・アトリビュート・レジスタ（続き）

アドレス	略称	レジスタの説明	アトリビュート指定対象アドレス
0184 83C0h	MAR240	メモリ・アトリビュート・レジスタ 240	F000 0000h ～ F0FF FFFFh
0184 83C4h	MAR241	メモリ・アトリビュート・レジスタ 241	F100 0000h ～ F1FF FFFFh
0184 83C8h	MAR242	メモリ・アトリビュート・レジスタ 242	F200 0000h ～ F2FF FFFFh
0184 83CCh	MAR243	メモリ・アトリビュート・レジスタ 243	F300 0000h ～ F3FF FFFFh
0184 83D0h	MAR244	メモリ・アトリビュート・レジスタ 244	F400 0000h ～ F4FF FFFFh
0184 83D4h	MAR245	メモリ・アトリビュート・レジスタ 245	F500 0000h ～ F5FF FFFFh
0184 83D8h	MAR246	メモリ・アトリビュート・レジスタ 246	F600 0000h ～ F6FF FFFFh
0184 83DCh	MAR247	メモリ・アトリビュート・レジスタ 247	F700 0000h ～ F7FF FFFFh
0184 83E0h	MAR248	メモリ・アトリビュート・レジスタ 248	F800 0000h ～ F8FF FFFFh
0184 83E4h	MAR249	メモリ・アトリビュート・レジスタ 249	F900 0000h ～ F9FF FFFFh
0184 83E8h	MAR250	メモリ・アトリビュート・レジスタ 250	FA00 0000h ～ FAFF FFFFh
0184 83ECh	MAR251	メモリ・アトリビュート・レジスタ 251	FB00 0000h ～ FBFF FFFFh
0184 83F0h	MAR252	メモリ・アトリビュート・レジスタ 252	FC00 0000h ～ FCFF FFFFh
0184 83F4h	MAR253	メモリ・アトリビュート・レジスタ 253	FD00 0000h ～ FDFF FFFFh
0184 83F8h	MAR254	メモリ・アトリビュート・レジスタ 254	FE00 0000h ～ FEFF FFFFh
0184 83FCh	MAR255	メモリ・アトリビュート・レジスタ 255	FF00 0000h ～ FFFF FFFFh

4.4.5 メモリ・アトリビュート・レジスタ（MAR*n*）

L2 アトリビュート・レジスタ（MAR*n*）の一般的な構造を図 4-12 に示し、表 4-22 で説明します。

図 4-12. メモリ・アトリビュート・レジスタ（MAR*n*）



凡例：R = リード専用。-*n* = リセット後の値。R/SW = スーパーバイザのみがリード / ライト可能。

表 4-22. メモリ・アトリビュート・レジスタ（MAR*n*）フィールドの説明

ビット	フィールド	値	説明
31-1	Reserved	0	予約。
0	PC		許可コピー・フィールドは、影響を受けるアドレス範囲のキャッシュ可能性をイネーブル / ディスエーブルします。
		0	メモリ範囲は、キャッシュ不可能。
		1	メモリ範囲は、キャッシュ可能。

4.4.6 権限およびキャッシュ・コントロール・レジスタ

L2 メモリ・アーキテクチャは、メモリ保護サポート機能を提供します。L2 メモリ保護アーキテクチャの詳細については、4.6 節を参照してください。

表 4-23 に、役割ごとにアクセスできる L2 キャッシュ・コントロール・レジスタの概要を示します。

表 4-23. L2 キャッシュ・コントロール・レジスタのアクセス権限

レジスタ	スーパーバイザ	ユーザ
L2CFG	R/W	R
L2INV	R/W	R
L2WB	R/W	R/W
L2WBINV	R/W	R/W
L2WBAR/WC	R/W	R/W
L2WIBAR/WC	R/W	R/W
L2IBAR/WC	R/W	R/W
MARxx	R/W	R

4.5 L2 パワーダウン

C64x+ メガモジュール・アーキテクチャには、いくつかのパワーダウン機能が用意されています。パワーダウン機能によっては、ユーザからは見えないものもあります。他の機能は、ソフトウェアで制御されます。ユーザによって制御されるパワーダウン機能は、動的なグループと静的なグループの 2 つに分類できます。動的パワーダウン機能は、実行時に限られた時間だけ使用されます。その一方で、静的パワーダウン機能は、CPU がアイドル・モードになっている時間使用されます。このようなパワーダウン機能は、指定モジュールまたはパワーダウン・コントローラ（PDC）の一部のレジスタで制御されています。本章の理解を深めるために、先に第 9 章をお読みください。

L2 メモリ・アーキテクチャは、プログラム実行中に、ポート 0 およびポート 1 に接続されたメモリの一部を、動的にパワーダウンする機能を備えています。プログラムで L2 のメモリ・ページをスリープ状態にしたり、後から手動でウェイクアップすることができます。また、C64x+ は、スリープ状態のときに、プログラムでスリープ状態のページをアクセスすると、自動的にウェイクアップすることもできます。

4.5.1 L2 メモリの動的パワーダウン

L2 メモリは、別々に独立して、電源を供給できるポートごとに 2 つずつ 4 つの論理ページに分割されます。2 の累乗で表される論理ページのサイズは、デバイスによって異なります。2 の累乗で表される論理ページのサイズは、それぞれの L2 ポートに接続されている、2 の累乗で表されるメモリ・サイズの 1/2 です。2 の累乗で表すことができない、特定の L2 上のメモリ・サイズを実装したデバイスは、他のデバイスより 1 ページのサイズが大きくなります。詳細については、各デバイスのデータシートを参照してください。

動的パワーダウン機能は、レジスタ・セット全体でプログラム可能です。表 4-24 に、概要を示します。これ以降ではこのようなレジスタを簡単に説明します。詳細については 4.5.3 節を参照してください。

表 4-24. L2 メモリ・パワーダウン・レジスタの概要

アドレス	略称	レジスタの説明	参照先
0184 C040h	L2PDWAKE0	L2 パワーダウン・ウェイク・レジスタ 0	4.5.3.1 項
0184 C044h	L2PDWAKE1	L2 パワーダウン・ウェイク・レジスタ 1	4.5.3.1 項
0184 C050h	L2PDSLEEP0	L2 パワーダウン・スリープ・レジスタ 0	4.5.3.2 項
0184 C054h	L2PDSLEEP1	L2 パワーダウン・スリープ・レジスタ 1	4.5.3.2 項
0184 C060h	L2PDSTAT0	L2 パワーダウン・ステータス・レジスタ 0	4.5.3.3 項
0184 C064h	L2PDSTAT1	L2 パワーダウン・ステータス・レジスタ 1	4.5.3.3 項

4.5.1.1 手動でのスリープ制御

プログラムで、特定の論理メモリ・ページをスリープ状態にしたり、ウェイクアップしたりすることができます。これにより、プログラムで電源を供給したり、スリープ状態にしたりする L2 のメモリ領域を手動で制御できます。

4.5.1.2 ウェイクおよびスリープ・コマンド

論理 L2 メモリ・ページをスリープ状態にするには、プログラムで、適切な L2PDSLEEP レジスタのページ・スリープ・ビットに 1 をライトします。たとえば、ポート 1 上でスリープ状態にするには、プログラムで、値 0000 0002 を L2PDSLEEP1 にライトします（ビット 1 を 1 にセットします）。

同様に、論理 L2 メモリ・ページをウェイクアップするには、プログラムで、適切な L2PDWAKE レジスタのページ・ウェイク・ビットに 1 をライトします。ページをウェイクアップする手順に従うことで、これらのページにアクセスする前に複数のページをウェイクアップすることができます。そのため、このウェイクアップ・プロセスに対応するストール・ペナルティ部分は、プログラムからは見えなくなります。

L2PDWAKE および L2PDSLEEP レジスタへゼロ・ビットをライトしても、対応する論理ページには影響を与えません。

スリープ状態になったページは、そのページ内のアドレスにアクセスすることによってウェイクアップできます。別の L2PDSLEEP コマンドがそのページに対して発行されない限り、そのページはスリープ状態に戻ることはありません。

L2 コントローラは、あるページに対するアクセスを現在処理中か、そのページに対する他のリクエストがスリープ・リクエストを受け取って処理するまでのわずかの時間内に届くと、ページに対するスリープ・リクエストをドロップする場合があります。これにより、L2 はすべての未処理のリクエストを完了するまで、スリープ・リクエストを格納できません。したがって、プログラムで L2PDSTATx レジスタをチェックして、特定のスリープ・リクエストを処理していないかどうかを判別する必要があります。

4.5.1.3 パワーダウン状態のレポーティング

L2PDSTAT0 および L2PDSTAT1 メモリ・マップド・レジスタは、各 L2 メモリ・ページの現在のスリープ状態を示すレポートを作成します。これらのレジスタを使用することで、アプリケーションはスリープ状態に置かれているページを判別できます。レジスタは、L2 ポートごとに両方の論理ページに対するスリープ状態を示すレポートを作成します。

4.5.2 L2 メモリの静的パワーダウン

メガモジュール全体と L2 メモリを同時にパワーダウンさせることができます。また L2 メモリは、メガモジュール全体がパワーダウンした場合にパワーダウンすることもあります。C64x+ メガモジュールをパワーダウンさせるのに必要なソフトウェア・シーケンスは、次のとおりです。

1. PDCCMD レジスタの MEGPD フィールドを 1 にセットして、パワーダウン・モードをイネーブルします。
2. メガモジュールをウェイクアップする CPU 割り込みをイネーブルします。他の割り込みをすべてディスエーブルします。
3. IDLE 命令を実行します。

メガモジュールは、上記ステップ 2 でイネーブルされた割り込みが発生するまでパワーダウン・モードになったままです。

メガモジュールがパワーダウンしている間に、DMA アクセスが L1D、L1P、L2 のいずれかのメモリに対して行われた場合、パワーダウン・コントローラ（PDC）は 3 つのメモリ・コントローラをウェイクします。PDC は DMA アクセスが処理されると、メモリ・コントローラを再度パワーダウンします。

注： ここで説明したように、メガモジュールをパワーダウンすることは、多くの場合、静的パワーダウンといいます。この用語は、このモードを説明するために使われます。これはこの機能が長時間にわたり使われていることが多いためです。動的パワーダウンという用語が本章で使われている場合は、パワーダウン・モードが限られた時間だけ使われていることを示します。

PDCCMD レジスタおよび C64x+ メガモジュールのパワーダウン機能の詳細については、第 9 章を参照してください。

4.5.3 L2 パワーダウン・コントロール・レジスタ

表 4-25 に、L2 メモリ・パワーダウン・レジスタの概要を示します。

表 4-25. L2 メモリ・パワーダウン・レジスタの概要

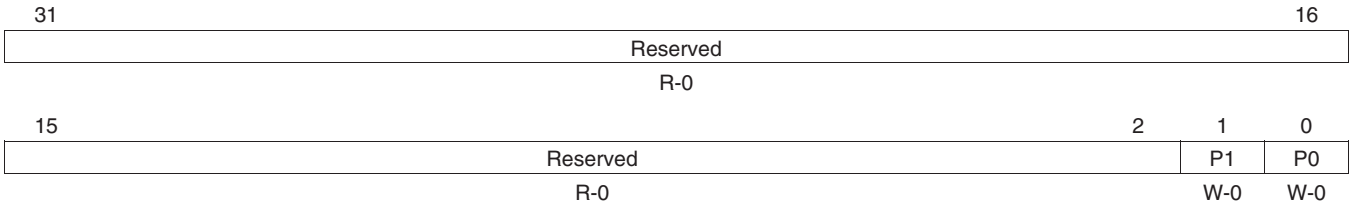
アドレス	略称	レジスタの説明	参照先
0184 C040h	L2PDWAKE0	L2 パワーダウン・ウェイク・レジスタ 0	4.5.3.1 項
0184 C044h	L2PDWAKE1	L2 パワーダウン・ウェイク・レジスタ 1	4.5.3.1 項
0184 C050h	L2PDSLEEP0	L2 パワーダウン・スリープ・レジスタ 0	4.5.3.2 項
0184 C054h	L2PDSLEEP1	L2 パワーダウン・スリープ・レジスタ 1	4.5.3.2 項
0184 C060h	L2PDSTAT0	L2 パワーダウン・ステータス・レジスタ 0	4.5.3.3 項
0184 C064h	L2PDSTAT1	L2 パワーダウン・ステータス・レジスタ 1	4.5.3.3 項

4.5.3.1 L2 パワーダウン・ウェイク・レジスタ（L2PDWAKE*n*）

レジスタをプログラムすることで、4 つの論理 L2 ページをパワーダウン状態からウェイクできます（図 4-13 を参照）。

L2 パワーダウン・ウェイク・レジスタ（L2PDWAKE*n*）を図 4-13 に示し、表 4-26 で説明します。

図 4-13. L2 パワーダウン・ウェイク・レジスタ（L2PDWAKE*n*）



凡例：R = リード専用。W = ライト専用。-*n* = リセット後の値。

表 4-26. L2 パワーダウン・ウェイク・レジスタ（L2PDWAKE*n*）フィールドの説明

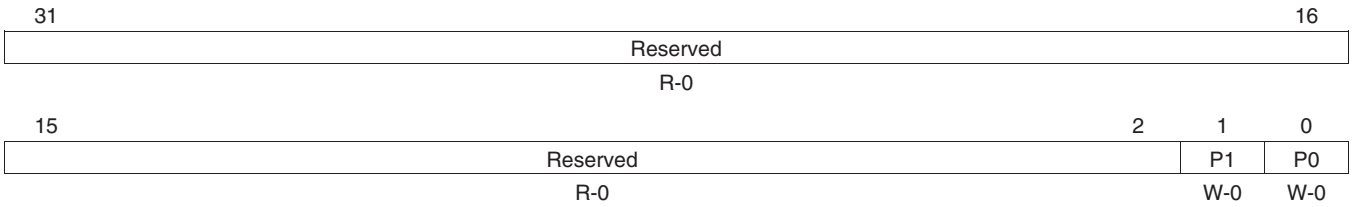
ビット	フィールド	値	説明
31-2	Reserved	0	予約。
1	P1		ページ 1、ポート 0、ポート 1 のいずれかをウェイクアップします。
		0	このフィールドにライトしても、ページ 1 には影響を与えません。
		1	このフィールドに 1 をライトすると、該当ページをパワーダウン状態からウェイクアップします。
0	P0		ページ 0、ポート 0、ポート 1 のいずれかをウェイクアップします。
		0	このフィールドに 0 をライトしても、ページ 0 には影響を与えません。
		1	このフィールドに 1 をライトすると、該当ページをパワーダウン状態からウェイクアップします。

4.5.3.2 L2 パワーダウン・スリープ・レジスタ（L2PDSLEEP*n*）

2 つのレジスタをプログラムすることで、4 つの論理 L2 ページをパワーダウン状態にすることができます（図 4-14 および図 4-13 を参照）。

L2 パワーダウン・スリープ・レジスタ（L2PDSLEEP*n*）を図 4-14 に示し、表 4-27 で説明します。

図 4-14. L2 パワーダウン・スリープ・レジスタ（L2PDSLEEP*n*）



凡例：R = リード専用。W = ライト専用。-*n* = リセット後の値。

表 4-27. L2 パワーダウン・スリープ・レジスタ（L2PDSLEEP*n*）フィールドの説明

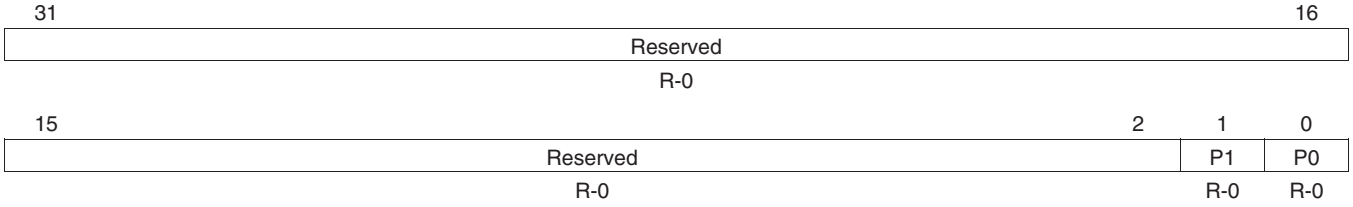
ビット	フィールド	値	説明
31-2	Reserved	0	予約。
1	P1		ページ 0、ポート 0、ポート 1 のいずれかをパワーダウン状態にします。
		0	このフィールドに 0 をライトしても、ページ 1 には影響を与えません。
		1	このフィールドに 1 をライトすると、該当ページをパワーダウン状態にします。
0	P0		ページ 0、ポート 0、ポート 0 のいずれかをパワーダウン状態にします。
		0	このフィールドに 0 をライトしても、ページ 0 には影響を与えません。
		1	このフィールドに 1 をライトすると、該当ページをパワーダウン状態にします。

4.5.3.3 L2 パワーダウン・ステータス・レジスタ（L2PDSTAT*n*）

4 つの L2 ページのパワーダウン状態が、2 つのステータス・レジスタにレポートされます。

L2 パワーダウン・ステータス・レジスタ（L2PDSTAT*n*）を図 4-15 に示し、表 4-28 で説明します。

図 4-15. L2 パワーダウン・ステータス・レジスタ（L2PDSTAT*n*）



凡例：R = リード専用。*-n* = リセット後の値。

表 4-28. L2 パワーダウン・ステータス・レジスタ（L2PDSTAT*n*）フィールドの説明

ビット	フィールド	値	説明
31-2	Reserved	0	予約。
1	P1		ページ 1、ポート n のパワーダウン状態をレポートします。
		0	ページ 1 は通常の状態。
		1	ページ 1 はパワーダウン状態。
0	P0		ページ 0、ポート n のパワーダウン状態をレポートします。
		0	ページ 0 は通常の状態。
		1	ページ 0 はパワーダウン状態。

4.5.3.4 権限および L2 パワーダウン・コントロール・レジスタ

パワーダウン・コントロール動作が受ける権限の影響は、次のように要約できます。

- スーパーバイザまたはユーザ・コードは、L2 論理ページのいずれかをパワーダウンしたり、ウェイクアップすることができます。

表 4-29 に、パワーダウン・コントロール・レジスタのアクセス権限の概要を示します。

表 4-29. L2 パワーダウン・コントロール・レジスタのアクセス権限

レジスタ	スーパーバイザ	ユーザ
L2PDWAKE0	R/W	R/W
L2PDWAKE1	R/W	R/W
L2PDSLEEP0	R/W	R/W
L2PDSLEEP1	R/W	R/W
L2PDSTAT0	R	R
L2PDSTAT1	R	R

4.6 L2 メモリ保護

L2 メモリはメモリ保護をサポートし、多くのシステムで求められる堅牢性をもたらします。メモリ保護はさまざまレベルで使用可能です。すべてのレベルがすべてのデバイスで使用できるとは限りません。詳細については、各デバイスのデータ・マニュアルを参照してください。理解が十分ではない場合には、第 8 章を参照して理解を深めてから、ここの説明をお読みください。

4.6.1 CPU、IDMA、および他のシステム・マスタによるアクセス時に行われる保護チェック

メモリ保護チェックは、メモリ保護サポート機能が組み込まれている L1P、L1D、IDMA、および他のシステム・マスタから、L2 により直接処理されるアクセスに対して行われます。

3 つのメモリ・コントローラにはすべて、C64x+ メガモジュールの割り込みコントローラへ送る 2 つの例外出力を備えているという特長があります。この例外出力の 1 つは、CPU によってトリガされる（「ローカルな」）メモリ例外が発生したことを示します。もう一方は、システム・マスタによってトリガされる（「リモートの」）例外が発生したことを示します。ほとんどのプログラムは、CPU によってトリガされる例外入力を CPU の例外入力へ送り、システム・マスタによってトリガされる入力を割り込み入力へ送る可能性があります。

L2 にヒットしたかミスしたかに関係なく、L2 は L2 に到着する CPU リード時に保護チェックを行いません。最終的に、リードはアクセス権をリクエスタに返します。その結果、L1D や L1P に対するチェックが遅れることになります。これに対し、L2 は L2 でヒットまたはミスとなるすべての CPU ライトをチェックし、その後 L2 キャッシュでラインをアロケートします。L2 は L2 でミスとなるキャッシュ不可能なライト時の権限をチェックしません。したがって、L2 は L2 で終了するすべての CPU アクセスをチェックし、最終的にアクセスを処理するコントローラ（L1P、L1D、外部ペリフェラル）に対する他のアクセスをチェックする作業を遅らせます。

すべてのシステム・マスタおよび IDMA による L2 メモリへのアクセス（リードとライト）は、常にチェックされます。システム・マスタおよび IDMA による L2 キャッシュに保持されているアドレスへのアクセスは、チェックされません。L2（または EMC）は L1D キャッシュに保持されているアドレスに対して保護チェックを行ってから、スヌープ・ライト・コマンドを L1D に発行します。

L2 コントローラでは、特定のリクエストを許可または拒否するかを決定します。これはそのリクエストに対応する権限レベル、およびそのリクエストによってアクセスされるアドレス範囲でのアクセス権限の設定に基づきます。これらのチェックに関する正しいルールは、第 8 章で説明しています。

特定のリクエストに十分なアクセス権限が設定されていない場合、L2 は例外をアサートし、そのリクエストを拒否します。許可されていないリードを行うと不正なデータが返され、ベースとなっているメモリがライトされる前に、許可されていないライトを行うと途中で打ち切られてしまいます。L2 は、CPU ライトが L2 内でキャッシュ可能な場合、L2 をミスした CPU ライトについて、アクセス権限のチェックだけを行います。また L2 内でキャッシュできない場合には、メモリ・システムを後期段階でチェックします。

4.6.2 L2 メモリ・プロテクション・レジスタ

次のレジスタは、L2 メモリを保護する動作を管理しています。MMR は 3 つの主要なカテゴリに分類されます。

- メモリ・プロテクション・ページ・アトリビュート・レジスタ（MPPA）：これらのレジスタは、保護されたページごとに対応するアクセス権限を格納します。
- メモリ・プロテクション・ロック・レジスタ（MPLK）：これらのレジスタは、ハードウェアによるメモリ保護ロック機能を実装します。ペリフェラルで他のリクエストを処理できない場合、ロックはそのペリフェラルのメモリ保護エントリに対するすべての更新をディスエーブルします。
- メモリ・プロテクション・フォールト・レジスタ（MPFxR）：メモリ保護障害を生成する各ペリフェラルには、障害の詳細を記録するために MPFAR、MPFSR、MPFCLR の各レジスタが用意されています。

4.6.2.1 L2 メモリ・プロテクション・レジスタ

表 4-30 に、L2 メモリ・プロテクション・レジスタを示します。

表 4-30. L2 メモリ・プロテクション・レジスタ

アドレス	略称	レジスタの説明	参照先
0184 A2xxh	L2MPPAxx	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ	4.6.2.2 項
0184 A100h	L2MPLK0	L2 メモリ・プロテクション・ロック・レジスタ 0	4.6.2.3.1.1 項
0184 A104h	L2MPLK1	L2 メモリ・プロテクション・ロック・レジスタ 1	4.6.2.3.1.2 項
0184 A108h	L2MPLK2	L2 メモリ・プロテクション・ロック・レジスタ 2	4.6.2.3.1.3 項
0184 A10Ch	L2MPLK3	L2 メモリ・プロテクション・ロック・レジスタ 3	4.6.2.3.1.4 項
0184 A110h	L2MPLKCMD	L2 メモリ・プロテクション・ロック・コマンド・レジスタ	4.6.2.3.1.5 項
0184 A114h	L2MPLKSTAT	L2 メモリ・プロテクション・ロック・ステータス・レジスタ	4.6.2.3.1.6 項
0184 A000h	L2MPFAR	L2 メモリ・プロテクション・フォールト・アドレス・レジスタ	4.6.2.4.1 項
0184 A004h	L2MPFSR	L2 メモリ・プロテクション・フォールト・セット・レジスタ	4.6.2.4.2 項
0184 A008h	L2MPFCR	L2 メモリ・プロテクション・フォールト・クリア・レジスタ	4.6.2.4.3 項

4.6.2.2 L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ（L2MPPAxx）

L2 は 64 ページのメモリ保護する機能を実装し、メモリ・ポートごとに 32 ページずつ備えています。L2MPPA0 ～ L2MPPA31 はポート 0 に、L2MPPA32 ～ L2MPPA63 はポート 1 にそれぞれ対応します。各ページのサイズはポートごと、デバイスごとに異なります。ページによっては、特定のデバイスでは使用できない場合があります。デバッグする目的で、使用していないページにも、すべてゼロの値をセットするようにプログラムしてください。

特定のデバイス上で使用されるページ・サイズおよびページ数を確認するには、各デバイスのデータ・マニュアルを参照してください。

L2 内の各ページには、ページに対応した 16 個のメモリ保護ビットがあります（図 4-16 を参照）。これらのメモリ保護機能を備えた 64 ページ内の保護ビットのデフォルト値は、リセット時に決められます。表 4-33 に、デフォルト設定を示します。

表 4-31 に、L2 メモリ・プロテクション・ページ・アトリビュート・レジスタを示します。

表 4-31. L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ

アドレス	略称	レジスタの説明	参照先
0184 A200h	L2MPPA0	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 0	4.6.2.2.1 項
0184 A204h	L2MPPA1	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 1	4.6.2.2.1 項
0184 A208h	L2MPPA2	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 2	4.6.2.2.1 項
0184 A20Ch	L2MPPA3	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 3	4.6.2.2.1 項
0184 A210h	L2MPPA4	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 4	4.6.2.2.1 項
0184 A214h	L2MPPA5	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 5	4.6.2.2.1 項
0184 A218h	L2MPPA6	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 6	4.6.2.2.1 項
0184 A21Ch	L2MPPA7	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 7	4.6.2.2.1 項
0184 A220h	L2MPPA8	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 8	4.6.2.2.1 項
0184 A224h	L2MPPA9	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 9	4.6.2.2.1 項
0184 A228h	L2MPPA10	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 10	4.6.2.2.1 項
0184 A22Ch	L2MPPA11	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 11	4.6.2.2.1 項
0184 A230h	L2MPPA12	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 12	4.6.2.2.1 項
0184 A234h	L2MPPA13	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 13	4.6.2.2.1 項
0184 A238h	L2MPPA14	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 14	4.6.2.2.1 項
0184 A23Ch	L2MPPA15	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 15	4.6.2.2.1 項
0184 A240h	L2MPPA16	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 16	4.6.2.2.1 項
0184 A244h	L2MPPA17	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 17	4.6.2.2.1 項
0184 A248h	L2MPPA18	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 18	4.6.2.2.1 項
0184 A24Ch	L2MPPA19	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 19	4.6.2.2.1 項
0184 A250h	L2MPPA20	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 20	4.6.2.2.1 項
0184 A254h	L2MPPA21	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 21	4.6.2.2.1 項
0184 A258h	L2MPPA22	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 22	4.6.2.2.1 項
0184 A25Ch	L2MPPA23	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 23	4.6.2.2.1 項
0184 A260h	L2MPPA24	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 24	4.6.2.2.1 項
0184 A264h	L2MPPA25	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 25	4.6.2.2.1 項
0184 A268h	L2MPPA26	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 26	4.6.2.2.1 項
0184 A26Ch	L2MPPA27	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 27	4.6.2.2.1 項
0184 A270h	L2MPPA28	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 28	4.6.2.2.1 項
0184 A274h	L2MPPA29	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 29	4.6.2.2.1 項
0184 A278h	L2MPPA30	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 30	4.6.2.2.1 項
0184 A27Ch	L2MPPA31	L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ 31	4.6.2.2.1 項

4.6.2.2.1 L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ（L2MPPA*n*）

L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ（L2MPPA*n*）を図 4-16 に示し、表 4-32 で説明します。

図 4-16. L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ（L2MPPA*n*）

31															16
Reserved															
R-0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AID5	AID4	AID3	AID2	AID1	AID0	AIDX	LOCAL	Reserved		SR	SW	SX	UR	UW	UX
R/W-config input															

凡例：R/W = リード / ライト。R = リード専用。-*n* = リセット後の値。

表 4-32. L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ（L2MPPA*n*）フィールドの説明

ビット	フィールド	値	説明
31-16	Reserved	0	予約。
15	AID5		ID = 5 からのアクセスを制御します。
		0	アクセスは拒否。
		1	アクセスは許可。
14	AID4		ID = 4 からのアクセスを制御します。
		0	アクセスは拒否。
		1	アクセスは許可。
13	AID3		ID = 3 からのアクセスを制御します。
		0	アクセスは拒否。
		1	アクセスは許可。
12	AID2		ID = 2 からのアクセスを制御します。
		0	アクセスは拒否。
		1	アクセスは許可。
11	AID1		ID = 1 からのアクセスを制御します。
		0	アクセスは拒否。
		1	アクセスは許可。
10	AID0		ID = 0 からのアクセスを制御します。
		0	アクセスは拒否。
		1	アクセスは許可。
9	AIDX		ID >= 6 からのアクセスを制御します。
		0	アクセスは拒否。
		1	アクセスは許可。
8	LOCAL		CPU からローカル・メモリ（L1/L2）へのアクセスを制御します。
		0	アクセスは拒否。
		1	アクセスは許可。
7-6	Reserved	0	予約。
5	SR		スーパーバイザによるリード・アクセス・タイプ。
		0	通常動作。
		1	スーパーバイザによるリード・リクエストを示します。

表 4-32. L2 メモリ・プロテクション・ページ・アトリビュート・レジスタ（L2MPPA*n*）フィールドの説明（続き）

ビット	フィールド	値	説明
4	SW		スーパーバイザによるライト・アクセス・タイプ。
		0	通常動作。
		1	スーパーバイザによるライト・リクエストを示します。
3	SX		スーパーバイザによる実行アクセス・タイプ。
		0	通常動作。
		1	スーパーバイザによる実行リクエストを示します。
2	UR		ユーザによるリード・アクセス・タイプ。
		0	通常動作。
		1	ユーザによるリード・リクエストを示します。
1	UW		ユーザによるライト・アクセス・タイプ。
		0	通常動作。
		1	ユーザによるライト・リクエストを示します。
0	UX		ユーザによる実行アクセス・タイプ。
		0	通常動作。
		1	ユーザによる実行リクエストを示します。

表 4-33. デフォルトのページ・アトリビュート・フィールド

許容 ID（ビット 15:8）	予約ビット（ビット 7:6）	アクセス・タイプ（ビット 5:0）
1111 1111	11	111 111

4.6.2.3 L2 メモリ・プロテクション・ロック・レジスタ

L2 は、メモリ・プロテクション・レジスタへのライト・アクセスを制御するために、64 ビットのロック・レジスタを実装しています。このようなロック・レジスタの動作については、第 8 章を参照してください。

表 4-34 に、L2 メモリ・プロテクション・ロック・レジスタを示します。

表 4-34. L2 メモリ・プロテクション・ロック・レジスタ

アドレス	略称	レジスタの説明	参照先
0184 A100h	L2MPLK0	L2 メモリ・プロテクション・ロック・レジスタ 0	4.6.2.3.1.1 項
0184 A104h	L2MPLK1	L2 メモリ・プロテクション・ロック・レジスタ 1	4.6.2.3.1.2 項
0184 A108h	L2MPLK2	L2 メモリ・プロテクション・ロック・レジスタ 2	4.6.2.3.1.3 項
0184 A10Ch	L2MPLK3	L2 メモリ・プロテクション・ロック・レジスタ 3	4.6.2.3.1.4 項
0184 A110h	L2MPLKCMD	L2 メモリ・プロテクション・ロック・コマンド・レジスタ	4.6.2.3.1.5 項
0184 A114h	L2MPLKSTAT	L2 メモリ・プロテクション・ロック・ステータス・レジスタ	4.6.2.3.1.6 項

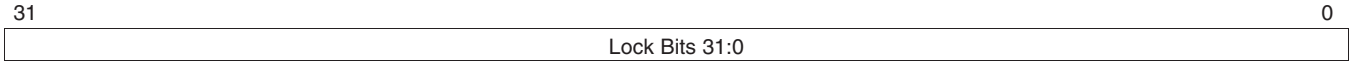
4.6.2.3.1 L2 メモリ・プロテクション・ロック・レジスタ（L2MPLK*n*）

L2 メモリ・プロテクション・ロック・レジスタ（L2MPLK*n*）を図 4-17 ～ 図 4-21 に示し、表 4-35 で説明します。

4.6.2.3.1.1 L2 メモリ・プロテクション・ロック・レジスタ 0（L2MPLK0）

L2 メモリ・プロテクション・ロック・レジスタ 0（L2MPLK0）を図 4-17 に示します。

図 4-17. L2 メモリ・プロテクション・ロック・レジスタ 0（L2MPLK0）

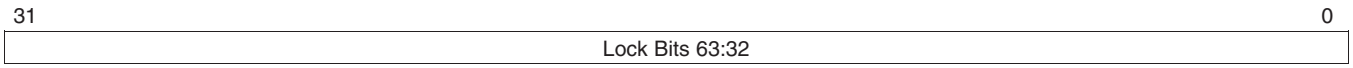


凡例：W = ライト専用。-*x* = 値は不定。詳細については、各デバイスのデータ・マニュアルを参照してください。

4.6.2.3.1.2 L2 メモリ・プロテクション・ロック・レジスタ 1（L2MPLK1）

L2 メモリ・プロテクション・ロック・レジスタ 1（L2MPLK1）を図 4-18 に示します。

図 4-18. L2 メモリ・プロテクション・ロック・レジスタ 1（L2MPLK1）

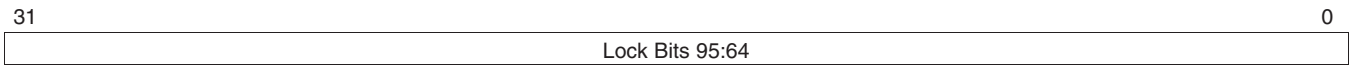


凡例：W = ライト専用。-*x* = 値は不定。詳細については、各デバイスのデータ・マニュアルを参照してください。

4.6.2.3.1.3 L2 メモリ・プロテクション・ロック・レジスタ 2（L2MPLK2）

L2 メモリ・プロテクション・ロック・レジスタ 2（L2MPLK2）を図 4-19 に示します。

図 4-19. L2 メモリ・プロテクション・ロック・レジスタ 2（L2MPLK2）

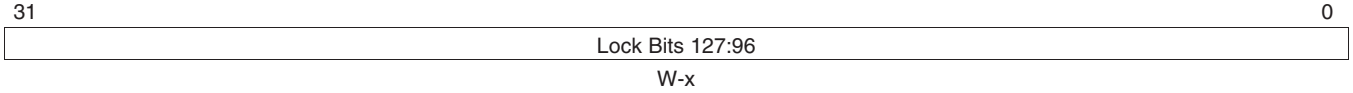


凡例：W = ライト専用。-*x* = 値は不定。詳細については、各デバイスのデータ・マニュアルを参照してください。

4.6.2.3.1.4 L2 メモリ・プロテクション・ロック・レジスタ 3（L2MPLK3）

L2 メモリ・プロテクション・ロック・レジスタ 3（L2MPLK3）を図 4-20 に示します。

図 4-20. L2 メモリ・プロテクション・ロック・レジスタ 3（L2MPLK3）

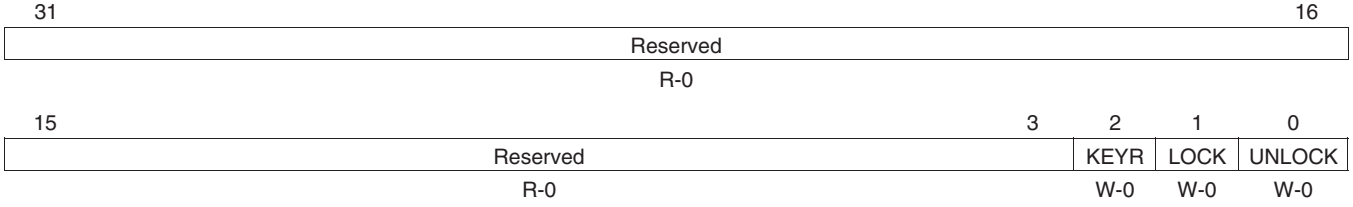


凡例：W = ライト専用。*-x* = 値は不定。詳細については、各デバイスのデータ・マニュアルを参照してください。

4.6.2.3.1.5 L2 メモリ・プロテクション・ロック・コマンド・レジスタ（L2MPLKCMD）

L2 メモリ・プロテクション・ロック・コマンド・レジスタ（L2MPLKCMD）を図 4-21 に示します。

図 4-21. L2 メモリ・プロテクション・ロック・コマンド・レジスタ（L2MPLKCMD）



凡例：R = リード専用。W = ライト専用。*-n* = リセット後の値。

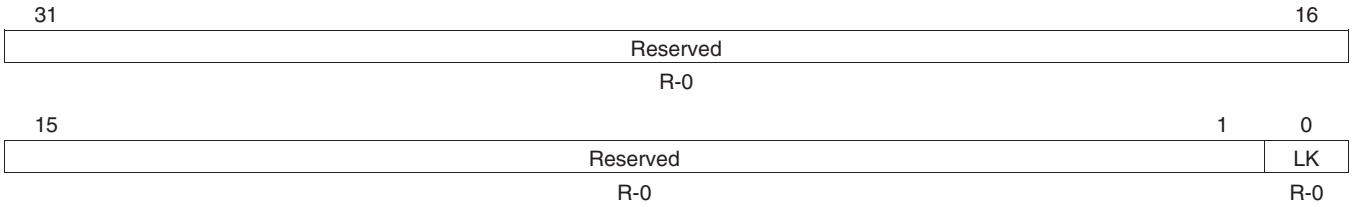
表 4-35. L2 メモリ・プロテクション・ロック・コマンド・レジスタ（L2MPLKCMD）フィールドの説明

ビット	フィールド	値	説明
31-3	Reserved	0	予約。
2	KEYR		ステータスをリセットします。
		0	影響なし。
		1	ステータスをリセットします。
1	LOCK		ロック・シーケンスを完了するためのインターフェイス。
		0	影響なし。
		1	ソフトウェアがシーケンスを正しく実行した場合に、ロックします。
0	UNLOCK		ロック解除シーケンスを完了するためのインターフェイス。
		0	影響なし。
		1	ソフトウェアがシーケンスを正しく実行した場合に、ロックを解除します。

4.6.2.3.1.6 L2 メモリ・プロテクション・ロック・ステータス・レジスタ（L2MPLKSTAT）

L2 メモリ・プロテクション・ロック・ステータス・レジスタ（L2MPLKSTAT）を図 4-22 に示し、表 4-36 で説明します。

図 4-22. L2 メモリ・プロテクション・ロック・ステータス・レジスタ（L2MPLKSTAT）



凡例：R = リード専用。*-n* = リセット後の値。

表 4-36. L2 メモリ・プロテクション・ロック・ステータス・レジスタ（L2MPLKSTAT）フィールドの説明

ビット	フィールド	値	説明
31-1	Reserved	0	予約。
0	LK		ロックの現在のステータスを示します。
		0	ロックは解除されています。
		1	ロックされています。

上記に示したように、メモリ保護アーキテクチャは、最大で 128 ビットのロック・サイズを許容します。L2 は、ロック・インターフェイスに対し 64 ビット・ロックのみを実装しています。そのため、L2MPLCK2 および L2MPLCK3 レジスタへライトされた値は無視されます。128 ビットより短いキーに関するロック・メカニズムの動作については、第 8 章を参照してください

4.6.2.4 L2 メモリ・プロテクション・フォールト・レジスタ

例外発生後にプログラムでメモリ保護障害を診断するために、L2 には障害に関する情報を格納する専用の 2 つのレジスタが実装されています。また、障害情報をクリアすることができる別のレジスタもあります。

表 4-37 に、L2 メモリ・プロテクション・フォールト・レジスタを示します。

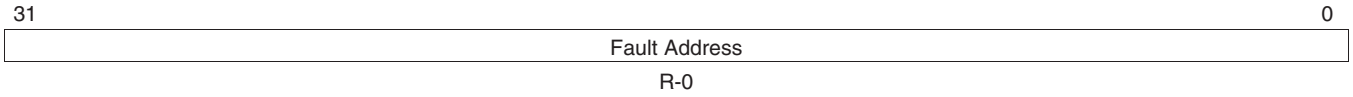
表 4-37. L2 メモリ・プロテクション・フォールト・レジスタ

アドレス	略称	レジスタの説明	参照先
0184 A000h	L2MPFAR	L2 メモリ・プロテクション・フォールト・アドレス・レジスタ	4.6.2.4.1 項
0184 A004h	L2MPFSR	L2 メモリ・プロテクション・フォールト・セット・レジスタ	4.6.2.4.2 項
0184 A008h	L2MPFCR	L2 メモリ・プロテクション・フォールト・クリア・レジスタ	4.6.2.4.3 項

4.6.2.4.1 L2 メモリ・プロテクション・フォールト・アドレス・レジスタ（L2MPFAR）

L2 メモリ・プロテクション・フォールト・アドレス・レジスタ（L2MPFAR）を図 4-23 に示し、表 4-38 で説明します。

図 4-23. L2 メモリ・プロテクション・フォールト・アドレス・レジスタ（L2MPFAR）



凡例：R = リード専用。*-n* = リセット後の値。

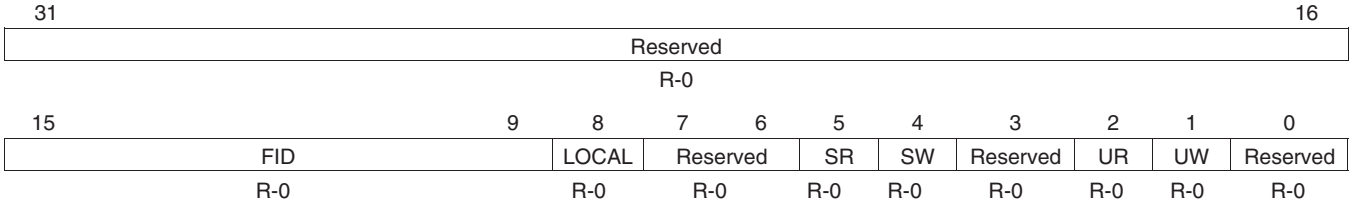
表 4-38. L2 メモリ・プロテクション・フォールト・アドレス・レジスタ（L2MPFAR）フィールドの説明

ビット	フィールド	値	説明
31-0	Fault Address	0 ～ FFFF FFFFh	障害が発生したアドレス。

4.6.2.4.2 L2 メモリ・プロテクション・フォールト・セット・レジスタ（L2MPFSR）

L2 メモリ・プロテクション・フォールト・セット・レジスタ（L2MPFSR）を図 4-24 に示し、表 4-39 で説明します。

図 4-24. L2 メモリ・プロテクション・フォールト・セット・レジスタ（L2MPFSR）



凡例：R = リード専用。*-n* = リセット後の値。

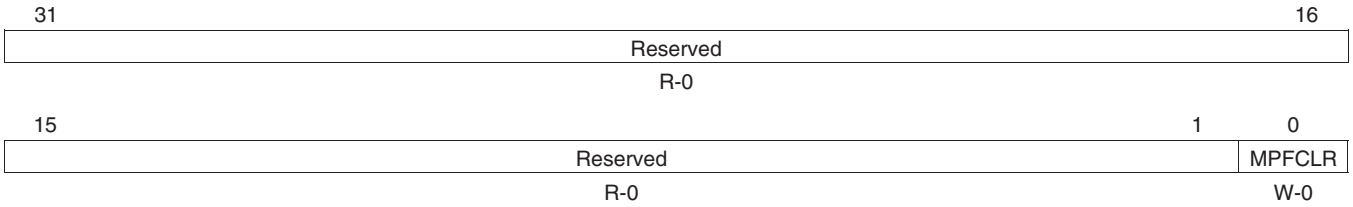
表 4-39. L2 メモリ・プロテクション・フォールト・セット・レジスタ（L2MPFSR）フィールドの説明

ビット	フィールド	値	説明
31-16	Reserved	0	予約。
15-9	FID	0 ～ 7Fh	ビット 6:0 は障害を引き起こしたリクエスタ の ID を示します。ID の幅が 7 ビットより狭い場合、残りのビットは 0 を返します。ID の幅が 7 ビットより広い場合、他のビットは切り捨てられます。LOCAL = 1 の場合、FID = 0。
8	LOCAL	0	通常動作。
		1	アクセスは「ローカル」。
7-6	Reserved	0	予約。
5	SR		スーパーバイザによるリード・アクセス・タイプ。
		0	通常動作。
		1	スーパーバイザによるリード・リクエストを示します。
4	SW		スーパーバイザによるライト・アクセス・タイプ。
		0	通常動作。
		1	スーパーバイザによるライト・リクエストを示します。
3	Reserved	0	予約。
2	UR		ユーザによるリード・アクセス・タイプ。
		0	通常動作。
		1	ユーザによるリード・リクエストを示します。
1	UW		ユーザによるライト・アクセス・タイプ。
		0	通常動作。
		1	ユーザによるライト・リクエストを示します。
0	Reserved	0	予約。

4.6.2.4.3 L2 メモリ・プロテクション・フォールト・クリア・レジスタ（L2MPFCLR）

L2 メモリ・プロテクション・フォールト・クリア・レジスタ（L2MPFCLR）を図 4-25 に示し、表 4-40 で説明します。

図 4-25. L2 メモリ・プロテクション・フォールト・クリア・レジスタ（L2MPFCLR）



凡例：R = リード専用。W = ライト専用。*-n* = リセット後の値。

表 4-40. L2 メモリ・プロテクション・フォールト・クリア・レジスタ（L2MPFCLR）フィールドの説明

ビット	フィールド	値	説明
31-1	Reserved	0	予約。
0	MPFCLR		L2MPFCR レジスタの内容をクリアするためのコマンド。
		0	影響なし。
		1	L2MPFAR および L2MPFCR レジスタの内容をクリアします。

メモリ・アクセス・プロテクション・フォールト・レジスタについては、第 8 章でその定義と意味について説明します。

L2MPFAR および L2MPFSR レジスタは、1 つの障害に関する十分な情報のみを格納します。一般的に、ハードウェアは最初の障害に関する情報を記録し、その障害のみの例外を生成します。L2 は、「ローカルな」（CPU によってトリガされる）および「リモートの」（システム・マスタ / IDMA によってトリガされる）障害という概念を使用しています。「ローカルな」障害は、「リモートの」障害を置き換え、新たな例外を生成可能です。このルールを簡潔に表すと次のようになります。

MPFSR レジスタの LOCAL フィールドの値が 0 でかつ、保留されている例外によってそのフィールドが 1 にセットされると、ハードウェアは新たな障害を記録し、新たな例外が発生したことを示す信号を送ります。

障害情報は、ソフトウェアで L2MPFCR レジスタの MPFCLR フィールドに 1 をライトしてクリアするまで、保持されます。ソフトウェアで、L2MPFCR レジスタの MPFCLR フィールドに 0 をライトしても影響ありません。L2 は L2MPFCR レジスタのビット 1 ～ 31 へライトされた値を無視します。

4.6.3 メモリ・プロテクション・レジスタへのアクセス時に行われる保護チェック

L2 は、メモリ・プロテクション・レジスタ自体にアクセス権限チェック機能を実装しています。ルールは次のとおりです。

- すべてのリクエスタは、いつでもどんな状況でも任意のメモリ・プロテクション（MP）レジスタをリードできます。ただし、（リードできない）プロテクション・ロック・レジスタ（L2MPLK0 ～ L2MPLK3）は除きます。
- スーパーバイザは、すべてのライト可能なレジスタにライトできます。

表 4-41 に、役割ごとにアクセスできる L2 メモリ・プロテクション・レジスタおよびメガモジュールで行われる保護チェックの内容について概要を示します。

表 4-41. L2 メモリ・プロテクション・レジスタのアクセス権限

レジスタ	スーパーバイザ	ユーザ
L2MPFAR	R	R
L2MPFSR	R	R
L2DMPFCR	W	/
L2DMPLK0	W	/
L2DMPLK1	W	/
L2DMPLK2	W	/
L2DMPLK3	W	/
L2DMPLKCMD	W	/
L2DMPLKSTAT	R	R
L2DMPPAxx	R/W	R

TEXAS INSTRUMENTS

第 5 章

JAJU058

内部ダイレクト・メモリ・アクセス（IDMA）コントローラ

項目 **ページ**

5.1 はじめに

ここでは、IDMA コントローラの目的を説明し、その機能について解説します。

5.1.1 内部ダイレクト・メモリ・アクセス（IDMA）コントローラの目的

IDMA コントローラの目的は、C64x+ メガモジュールに対してローカルな 2 つのメモリ・ロケーション間で高速なブロック転送を行うことです。ローカルなメモリ・ロケーションは、レベル 1 プログラム（L1P）、レベル 1 データ（L1D）、およびレベル 2（L2）メモリに含まれるロケーションまたは、外部ペリフェラル・コンフィギュレーション（CFG）に含まれるロケーションとして規定されています。IDMA コントローラは、内部 MMR 空間との間でデータを転送できません。

5.1.2 機能

IDMA コントローラを使うと、すべてのローカル・メモリ間で高速にデータを転送できます。コードとデータのセクションをメガモジュールに対してローカルなメモリ･マップド RAM にページングするための高速な方法を提供します。IDMA コントローラの主な利点は、低速なメモリ（レベル 2 - L2）と高速なメモリ（レベル 1 - L1D および L1P）間での転送を可能にするという点です。IDMA コントローラは、CPU が動作しているバックグラウンドで、転送が行われるため、キャッシュ・コントローラよりレイテンシが少なくなります。そのため、キャッシュによるストールが取り除かれます。

また、IDMA コントローラによって、メガモジュールの外部コンフィギュレーション空間（CFG）ポートからアクセスされるペリフェラル・コンフィギュレーション・レジスタを迅速にプログラムできるようになります。IDMA コントローラは、32 ワードの粒度を備えた外部コンフィギュレーション空間にアクセスでき、32 ワード・ブロック内の任意のレジスタを個別にアクセスすることができます。

要約すると、次のようになります。

- メモリ・ブロックのバースト転送（連続したデータ）用に最適化されている
- 任意のローカル・メモリ（L1P、L1D、L2（ページ 0 および 1））と外部 CFG 間でアクセス可能（ただし、ソースとデスティネーションを両方とも CFG には指定できない）。CFG は、チャネル 0 にのみアクセス可能。CFG 間での転送は不可。
- CPU へのプログラム可能な割り込みによって転送完了を示す

また IDMA コントローラは、プログラムされた埋め込み値を使用して、ブロック単位でライトを発行する場所でメモリをブロック単位で埋め込むことができます。

5.2 用語と定義

本章で使用する用語の詳細な定義については、本書の付録 B を参照してください。付録 B では、本書全体で使用している一般的な用語について説明しています。

5.3 IDMA アーキテクチャ

IDMA コントローラは、ローカル・メモリ間でデータを高速に転送することも、コンフィギュレーション・レジスタを迅速にプログラムすることもできる手段を提供します。この機能を完全にサポートするために、IDMA はチャネル 0 とチャネル 1 の 2 つのチャネルで構成されています。2 つのチャネルは、お互いに対して完全に直交で、同時動作を可能にしています。

IDMA の動作は、いくつかのレジスタによって制御されます。表 5-1 に、このようなレジスタの概要を示します。ここではこのようなレジスタを簡単に説明します。詳細については 5.4 節を参照してください。

表 5-1. IDMA レジスタの説明

レジスタ	説明
IDMA0_STAT	IDMA0 ステータス・レジスタ
IDMA0_MASK	IDMA0 マスク・レジスタ
IDMA0_SOURCE	IDMA0 ソース・アドレス・レジスタ
IDMA0_DEST	IDMA0 デスティネーション・アドレス・レジスタ
IDMA0_COUNT	IDMA0 ブロック・カウント・レジスタ
IDMA1_STAT	IDMA1 ステータス・レジスタ
IDMA1_SOURCE	IDMA1 ソース・アドレス・レジスタ
IDMA1_DEST	IDMA1 デスティネーション・アドレス・レジスタ
IDMA1_COUNT	IDMA1 ブロック・カウント・レジスタ

5.3.1 IDMA チャネル 0

IDMA チャネル 0 は、外部コンフィギュレーション空間（CFG）に配置されたコンフィギュレーション・レジスタを迅速にプログラムすることを目的としています。ローカル・メモリ（L1P、L1D、L2）から外部コンフィギュレーション空間へデータを転送します。

外部コンフィギュレーション空間には、メガモジュール外部に配置されたペリフェラル・レジスタが内蔵されています。それに対して、内部コンフィギュレーション空間には、メガモジュール内部に配置されたレジスタが内蔵されています本書で説明するレジスタは、いずれも内部コンフィギュレーション空間に属するものです。たとえば、レベル 1 データ（L1D）キャッシュを制御するために使われるレジスタは、内部コンフィギュレーション空間の一部を構成しています。内部コンフィギュレーション空間には、ロード / ストア命令を直接使用して CPU からのみアクセスできます。

IDMA チャネル 0 は、外部コンフィギュレーション空間のみアクセスできます。32 個の連続したレジスタのブロックを同時にアクセス可能です。IDMA チャネル 0 はこの機能を実現するために、ステータス、マスク、ソース・アドレス、デスティネーション・アドレス、およびブロック・カウントの 5 個のレジスタを備えています。

5.3.1.1 IDMA チャネル 0 の動作

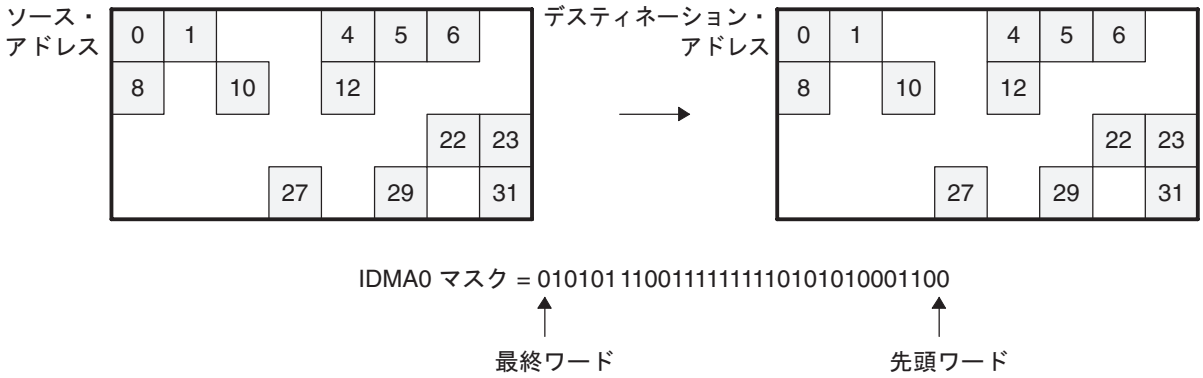
IDMA チャネル 0 に対して使われるソースおよびデスティネーション・アドレスは、適切な動作を行うために 32 バイトにアラインされている必要があります。図 5-1 に、IDMA チャネル 0 を使用したときの想定される転送を示します。

ローカル・メモリ（L1P、L1D、L2）の CFG レジスタを初期化するための値を含む 32 ワードからなるブロックを指定します。次に、IDMA チャネル 0 をプログラムして、これらの値を CFG レジスタへ転送します。

32 個の連続したロケーションをすべてプログラムすることが常に望ましいとは限らないため、マスク・レジスタが用意されています。つまり、ロケーションによっては、予約されていたり、実際のレジスタを表していなかったりする場合があります。そのため、そのようなロケーションをプログラムしてはいけません。

マスク・レジスタは、32 ビットのレジスタです。このレジスタの各ビットは、転送が行われるブロックの 32 ワードのいずれかにマップされます。たとえば、ビット 0 はワード 0 に、ビット 1 はワード 1 といったようにマップされます。マスク・ビットを 1 にセットすると、ブロックの対応するワードは転送されません。

図 5-1. IDMA チャネル 0 のトランザクション



5.3.1.2 IDMA チャネル 0 の例外

ソース・アドレスとデスティネーション・アドレスの両方が CFG に対するものである場合、IDMA チャネル 0 は、例外を生成します。その例外は、C64x+ メガモジュールの割り込みコントローラへ送られます。

IDMA コントローラ動作が停止した動作の最初のサイクル時に、例外が生成されてから、保留されていた IDMA チャネル 0 の要求が処理されます。IDMA チャネル 0 の例外は、IDMA チャネル 1 にはまったく影響を与えません。

5.3.1.3 IDMA チャネル 0 のプログラミング

CPU がそれぞれのコンフィギュレーション・レジスタにライトすると、IDMA 転送は自動的に発行されます。CPU は、IDMA 転送がトリガするチャネルのすべてのレジスタに対して、シーケンシャルにインクリメントしながらライトします。チャネル 0 では、CPU はソース・アドレス、デスティネーション・アドレス、カウント・レジスタの順にマスクに対してライトします。カウント・レジスタにライトした後に、転送リクエストが発行されます。

IDMA チャネルごとに、1 回の転送が特定の時点でアクティブになります。CPU はパラメータを更新して、後続の転送をキューに積みますが、転送はアクティブな転送が完了するまで開始されることはありません。これにより、2 つの転送（アクティブと保留）が特定の時点で CPU から処理されないようにすることが可能になります。

転送完了時に、CPU 割り込みが必要に応じてセットされます。

5.3.1.3.1 IDMA チャネル 0 の例 1

IDMA チャネル 0 を使用して、コンフィギュレーション・レジスタに対する更新を疑似コードで記述した例を例 5-1 に示します。

例 5-1. IDMA チャネル 0 を使用したコンフィギュレーション・レジスタの更新

```
IDMA0_MASK = 0x00000F0F;    //Set mask for 8 regs -- 11:8, 3:0
IDMA0_SOURCE = MMR_ADDRESS;       //Set source to config location
IDMA0_DEST = reg_ptr;             //Set destination to data memory address
IDMA0_COUNT = 0;                  //Set mask for 1 block

while (IDMA0_STATUS);             //Wait for transfer completion

... update register values ...

IDMA0_MASK = 0x00000F0F;          //Set mask for 8 regs -- 11:8, 3:0
IDMA0_SOURCE = reg_ptr;           //Set source to updated value pointer
IDMA0_DEST = MMR_ADDRESS;         //Set destination to config location
IDMA0_COUNT = 0;                  //Set mask for 1 block
```

5.3.1.3.2 IDMA チャネル 0 の例 2

EDMA は、C64x+ デバイスで一般的に使用可能なペリフェラルです。複数の QDMA リクエストを発行する例を例 5-2 に示します。QDMA に対応可能な EDMA 内のコンフィギュレーション空間には、8 ワードのロケーションが 16 個あります。各 QDMA は 8 ワードのパラメータ・エントリで指定されたように、転送リクエストを発行します。

例 5-2 に、32 個の QDMA が発行される様子を示します。ビデオ・アプリケーションで可能なように、各 QDMA のソース・アドレス、デスティネーション・アドレス、およびオプションのみが変更されます。

QDMA の詳細については、EDMA に関する資料を参照してください。

例 5-2. IDMA チャネル 0 を使用した 32 個の QDMA の更新

```
IDMA0_MASK = 0x4F4F4F4F;    //Set mask for 0, 1, 2, 3, 6 for each QDMA
IDMA0_SOURCE = &qdma_list[0];     //Set source to transfer list
IDMA0_DEST = &QDMA[0];            //Set destination to base address of QDMAs
IDMA0_COUNT = 3;                  //Set mask for 4 blocks (16 QDMAs)

IDMA0_MASK = 0x4F4F4F4F;          //Set mask for 0, 1, 2, 3, 6 for each QDMA
IDMA0_SOURCE = &qdma_list[16];    //Set source to second half of transfer list
IDMA0_DEST = &QDMA[0];            //Set destination to base address of QDMAs
IDMA0_COUNT = 3;                  //Set mask for 4 blocks (16 QDMAs)

while (IDMA0_STATUS);             //Wait for transfer completion
```

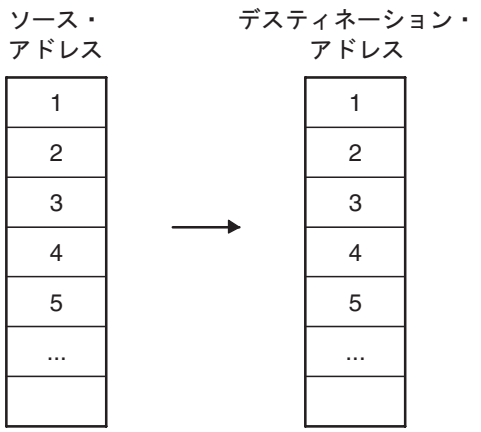
5.3.2 IDMA チャネル 1

IDMA チャネル 1 は、ローカル・メモリ間でのデータ転送を目的としています。CPU を動作させることなく、高速なメモリからの処理を調整するために、データおよびプログラムのセクションをバックグラウンドで転送します。これを可能にするために、IDMA チャネル 1 は、ステータス、ソース・アドレス、デスティネーション・アドレス、およびカウントの 4 つのレジスタを備えています。

5.3.2.1 IDMA チャネル 1 の動作

すべてのソース・アドレスとデスティネーション・アドレスは、転送全体で直線的にインクリメントします。転送サイズ（バイト数）は、IDMA チャネル 1 カウント・レジスタ（IDMA1_COUNT）の COUNT フィールドでセットされます。転送完了後に、CPU 割り込みが必要に応じてセットされます。キャッシュまたは EDMA との競合時のアービトレーションは、カウント・レジスタのオプション・フィールドにセットされたプライオリティに基づきます。図 5-2 に、IDMA チャネル 1 を使用したときの転送を示します。

図 5-2. IDMA チャネル 1 のトランザクション



5.3.2.2 IDMA チャネル 1 のプログラミング

CPU がそれぞれのコンフィギュレーション・レジスタにライトすると、IDMA 転送は自動的に発行されます。CPU は、IDMA 転送がトリガするチャネルのすべてのレジスタに対して、シーケンシャルにインクリメントしながらライトします。チャネル 1 では、CPU はソース・アドレス、デスティネーション・アドレス、カウント・レジスタへ（この順番で）ライトします。カウント・レジスタにライトした後に、転送リクエストが発行されます。

IDMA チャネルごとに、1 回の転送が任意の時点でアクティブになります。CPU は、パラメータを更新して、後続の転送をキューに積みますが、転送はアクティブな転送が完了するまで開始されることはありません。これにより、チャネルごとに 2 つの転送が任意の時点で CPU から処理されないようにすることが可能になります。

5.3.2.2.1 IDMA チャネル 1 の例

IDMA チャネル 1 を使用した新たなデータのページ・インと古いデータのページ・アウトを疑似コードで記述した例を例 5-3 に示します。

例 5-3. IDMA チャネル 1 を使用した新たなデータのページ・インと古いデータのページ・アウト

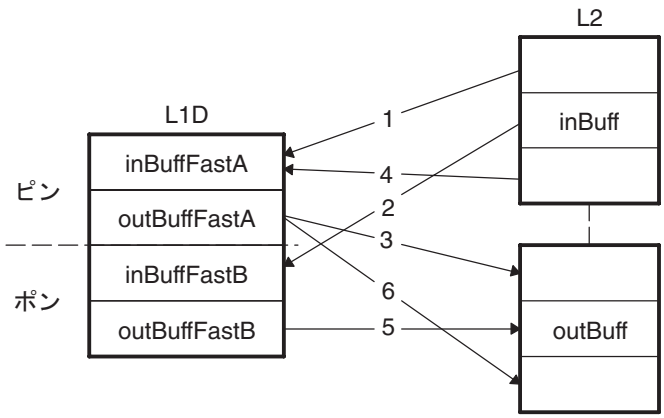
```
//Transfer ping buffers to/from L1D
//Return output buffer n - 1 to slow memory
IDMA1_SOURCE = outBuffFastA;          //Set source to fast memory output (L1D)
IDMA1_DEST   = &outBuff[n-1];         //Set destination to output buffer (L2)
IDMA1_COUNT  = 7 << IDMA_PRI_SHIFT |  //Set priority to low
               0 << IDMA_INT_SHIFT |  //Do not interrupt CPU
               buffsize;              //Set count to buffer size

//Page in input buffer n + 1 to fast memory
IDMA1_SOURCE = inBuff[n+1];           //Set source to buffer location (L2)
IDMA1_DEST   = inBuffFastA;           //Set destination to fast memory (L1D)
IDMA1_COUNT  = 7 << IDMA_PRI_SHIFT |  //Set priority to low
               1 << IDMA_INT_SHIFT |  //Interrupt CPU on completion
               buffsize;              //Set count to buffer size

... Process input buffer n in Pong -- inBuffFastB -> outBuffFastB ...
```

この例は、新たなデータを処理する上で高速メモリに転送するために、メモリの該当ロケーションで IDMA コントロールが返す出力データを使用する様子を説明しています。

図 5-3. IDMA チャネル 1 の例



5.3.2.2.2 IDMA チャネル 1 を使用したメモリ埋め込み

IDMA チャネル 1 を使用して、特殊な値をローカル・メモリ・セクションに埋め込むことができます。これを行うために、IDMA1_COUNT レジスタの FILL フィールドを 1 にセットします。FILL フィールドが 1 にセットされた場合、IDMA1 ソース・アドレス・レジスタに含まれている値は埋め込み値として使用されます。この値は、IDMA1 デスティネーション・アドレス・レジスタが指すメモリ・バッファにコピーされます。IDMA1_COUNT レジスタの COUNT フィールドは、この値がコピーされた回数を指定します。

5.4 レジスタ

表 5-2 のレジスタ・セットを使用して、IDMA コントローラをプログラムします。このようなレジスタのメモリ・アドレスについては、各デバイスのデータ・マニュアルを参照してください。

それぞれのレジスタは、CPU からリード / ライトのアクセスの両方に対してアクセス可能です。それぞれの IDMA レジスタへのアクセスは、32 ビットにアラインされている必要があります。ハーフワードおよびバイトで IDMA レジスタへライトすると、レジスタ全体がライトされます。そのため、適切な動作を行うためには、そのライト動作を回避する必要があります。アラインされないワードおよびダブル・ワードでアクセスした場合、その結果は不定です。そのため、これについても同様に回避する必要があります。

表 5-2 に、IDMA のレジスタをすべて示します。

表 5-2. 内部ダイレクト・メモリ・アクセス（IDMA）レジスタ

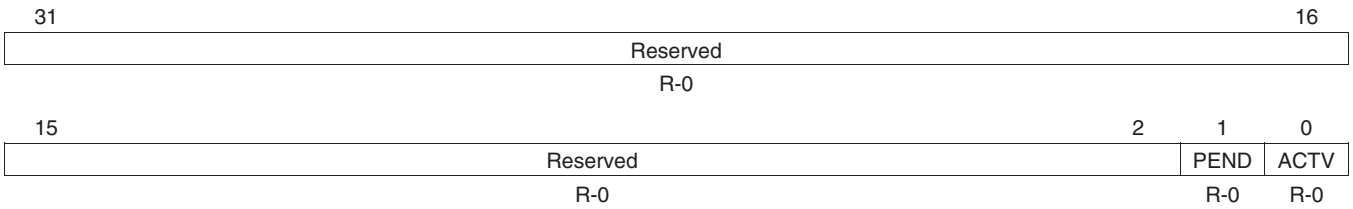
アドレス	略称	レジスタの説明	参照先
0182 0000h	IDMA0_STAT	IDMA チャネル 0 ステータス・レジスタ	5.4.1 項
0182 0004h	IDMA0_MASK	IDMA チャネル 0 マスク・レジスタ	5.4.2 項
0182 0008h	IDMA0_SOURCE	IDMA チャネル 0 ソース・アドレス・レジスタ	5.4.3 項
0182 000Ch	IDMA0_DEST	IDMA チャネル 0 デスティネーション・アドレス・レジスタ	5.4.4 項
0182 0010h	IDMA0_COUNT	IDMA チャネル 0 ブロック・カウント・レジスタ	5.4.5 項
0182 0100h	IDMA1_STAT	IDMA チャネル 1 ステータス・レジスタ	5.4.6 項
0182 0108h	IDMA1_SOURCE	IDMA チャネル 1 ソース・アドレス・レジスタ	5.4.7 項
0182 010Ch	IDMA1_DEST	IDMA チャネル 1 デスティネーション・アドレス・レジスタ	5.4.8 項
0182 0110h	IDMA1_COUNT	IDMA チャネル 1 ブロック・カウント・レジスタ	5.4.9 項

5.4.1 IDMA チャネル 0 ステータス・レジスタ（IDMA0_STAT）

IDMA チャネル 0 ステータス・レジスタ（IDMA0_STAT）は、チャネルの動作状態を示します。転送が進行中（ACTV）か、転送が保留されているか（PEND）を示す 2 つのビットがあります。

IDMA チャネル 0 ステータス・レジスタ（IDMA0_STAT）を図 5-4 に示し、表 5-3 で説明します。

図 5-4. IDMA チャネル 0 ステータス・レジスタ（IDMA0_STAT）



凡例：R = リード専用。*-n* = リセット後の値。

表 5-3. IDMA チャネル 0 ステータス・レジスタ（IDMA0_STAT）フィールドの説明

ビット	フィールド	値	説明
31-2	Reserved	0	これらの予約ビット・ロケーションは、常にゼロとしてリードされます。このフィールドに値をライトしても影響はありません。
1	PEND		転送の保留。CPU がコントロール・レジスタにライトし、かつアクティブな転送がすでに進行中（ACTV = 1）の場合、PEND ビットはセットされます。転送がアクティブの場合、PEND ビットはクリアされます。
		0	保留されている転送はありません。
		1	転送は保留されています。
0	ACTV		アクティブな転送。チャネル 0 がソース・アドレス・レジスタ（IDMA0_SOURCE）からデータのリードを開始している場合、ACTV ビットはセットされます。また、デスティネーション・アドレス・レジスタ（IDMA0_DEST）へ最後のライトが行われた後に、ACTV ビットはクリアされます。
		0	アクティブな転送はありません。
		1	アクティブな転送。

5.4.2 IDMA チャネル 0 マスク・レジスタ（IDMA0_MASK）

IDMA チャネル 0 マスク・レジスタ（IDMA0_MASK）を使用すると、転送ブロック内の不必要なレジスタをマスクすることができます。ソース / デスティネーション・アドレス・レジスタによって示される 32 ワードのメモリ・ブロック内のレジスタに対して個々の制御を可能にする 32 個のビットが備えられています。

IDMA チャネル 0 マスク・レジスタ（IDMA0_MASK）を図 5-5 に示し、表 5-4 で説明します。

図 5-5. IDMA チャネル 0 マスク・レジスタ（IDMA0_MASK）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
M31	M30	M29	M28	M27	M26	M25	M24	M23	M22	M21	M20	M19	M18	M17	M16
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
M15	M14	M13	M12	M11	M10	M9	M8	M7	M6	M5	M4	M3	M2	M1	M0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

凡例：R/W = リード / ライト。*-n* = リセット後の値。

表 5-4. IDMA チャネル 0 マスク・レジスタ（IDMA0_MASK）フィールドの説明

ビット	フィールド	値	説明
31-0	M*n*		レジスタ・マスク・ビット。
		0	レジスタへのアクセスは許可（マスクされていない）。
		1	レジスタへのアクセスはブロック（マスクされている）。

5.4.3 IDMA チャネル 0 ソース・アドレス・レジスタ（IDMA0_SOURCE）

IDMA チャネル 0 ソース・アドレス・レジスタ（IDMA0_SOURCE）は、IDMA 転送に対するソース・アドレスを示します。転送のソースは、C64x+ メガモジュールに対してローカルでなければならず、L1P、L1D、L2、CFG のいずれかに含まれている必要があります。デスティネーション・アドレスが CFG に対するアドレスの場合、転送対象のソース・アドレスは、ローカルな RAM ロケーションでなければなりません。その逆に、デスティネーション・アドレスがローカルな RAM ロケーションに対するアドレスの場合、ソース・アドレスは CFG に対するアドレスでなければなりません。また、ソース・アドレスは 32 バイトにアラインされている必要があります。

IDMA チャネル 0 ソース・アドレス・レジスタ（IDMA0_SOURCE）を図 5-6 に示し、表 5-5 で説明します。

図 5-6. IDMA チャネル 0 ソース・アドレス・レジスタ（IDMA0_SOURCE）

31	16
SOURCEADDR	
R/W-0	

15	5	4	0
SOURCEADDR		Reserved	
R/W-0		R-0	

凡例：R/W = リード / ライト。R = リード専用。*-n* = リセット後の値。

表 5-5. IDMA チャネル 0 ソース・アドレス・レジスタ（IDMA0_SOURCE）フィールドの説明

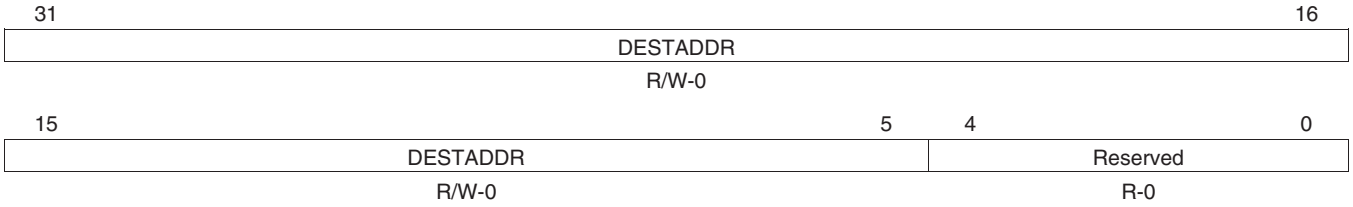
ビット	フィールド	値	説明
31-5	SOURCEADDR	0 ～ 7FF FFFFh	ソース・アドレス。C64x+ メガモジュールに対して、または有効なコンフィギュレーション・レジスタ空間に対してローカルな 32 バイトにアラインされている（ブロックにアラインされているなど）メモリ・ロケーションを指す必要があります。
4-0	Reserved	0	予約。

5.4.4 IDMA チャネル 0 デスティネーション・アドレス・レジスタ（IDMA0_DEST）

IDMA チャネル 0 デスティネーション・アドレス・レジスタ（IDMA0_DEST）は、IDMA 転送に対するデスティネーション・アドレスを示します。転送のデスティネーションは、C64x+ メガモジュールに対してローカルでなければならず、L1P、L1D、L2、CFG のいずれかに含まれている必要があります。ソース・アドレスが CFG に対するアドレスの場合、転送対象のデスティネーション・アドレスは、ローカルな RAM ロケーションでなければなりません。その逆に、ソース・アドレスがローカルな RAM ロケーションに対するアドレスの場合、デスティネーション・アドレスは CFG に対するアドレスでなければなりません。また、ソース・アドレスは 32 バイトにアラインされている必要があります。

IDMA チャネル 0 デスティネーション・アドレス・レジスタ（IDMA0_DEST）を図 5-7 に示し、表 5-6 で説明します。

図 5-7. IDMA チャネル 0 デスティネーション・アドレス・レジスタ（IDMA0_DEST）



凡例：R/W = リード / ライト。R = リード専用。-*n* = リセット後の値。

表 5-6. IDMA チャネル 0 デスティネーション・アドレス・レジスタ（IDMA0_DEST）フィールドの説明

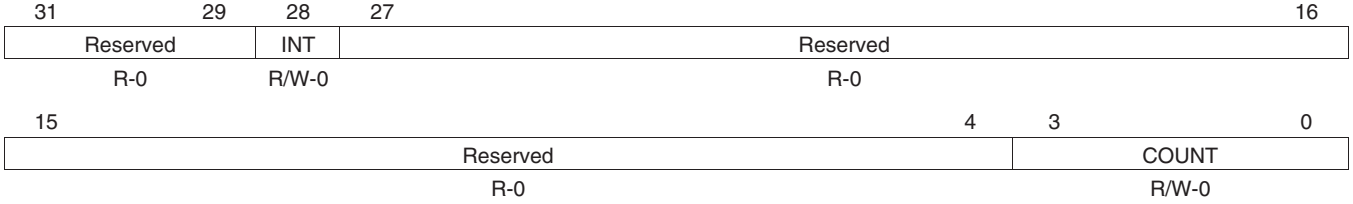
ビット	フィールド	値	説明
31-5	DESTADDR	0 ～ 7FF FFFFh	デスティネーション・アドレス。C64x+ メガモジュールに対して、または有効なコンフィギュレーション・レジスタ空間に対してローカルな 32 バイト（ウィンドウ）にアラインされているメモリ・ロケーションを指す必要があります。
4-0	Reserved	0	予約。

5.4.5 IDMA チャネル 0 カウント・レジスタ（IDMA0_COUNT）

IDMA チャネル 0 カウント・レジスタ（IDMA0_COUNT）は、データ転送時にアクセスできる 32 ワードのブロック数を示します。4 ビットからなる COUNT フィールドを使用して、最大で 16 ブロックを連続してアクセスできます。マスク・フィールドはすべてのブロックに適用され、繰り返しパターンをアクセスすることを可能にします。すべてのブロックは連続した 32 ワードの領域から構成されており、ソース・アドレスとデスティネーション・アドレスは適宜インクリメントされます。また、IDMA0_COUNT レジスタは CPU 割り込み（IDMA_INT0）をイネーブルでき、CPU に対して転送が完了したことを通知します。

IDMA チャネル 0 カウント・レジスタ（IDMA0_COUNT）を図 5-8 に示し、表 5-7 で説明します。

図 5-8. IDMA チャネル 0 カウント・レジスタ（IDMA0_COUNT）



凡例：R/W = リード / ライト。R = リード専用。-*n* = リセット後の値。

表 5-7. IDMA チャネル 0 カウント・レジスタ（IDMA0_COUNT）フィールドの説明

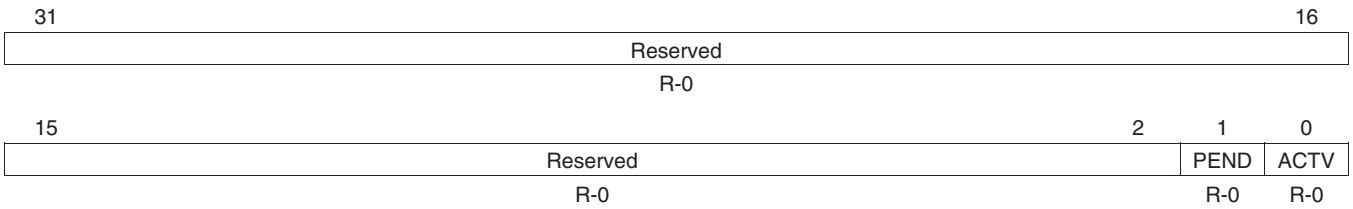
ビット	フィールド	値	説明
31-29	Reserved	0	これらの予約ビット・ロケーションは、常にゼロとしてリードされます。このフィールドに値をライトしても影響はありません。
28	INT		CPU 割り込みのイネーブル。
		0	転送完了時に CPU 割り込みを行いません。
		1	転送完了時に CPU 割り込み（IDMA_INT0）を行います。
27-4	Reserved	0	これらの予約ビット・ロケーションは、常にゼロとしてリードされます。このフィールドに値をライトしても影響はありません。
3-0	COUNT	0 ～ Fh	4 ビットのブロック・カウント。
		0	1 個の 32 ワード・ブロック間で転送します。
		1h ～ Fh	n+1 個の 32 ワード・ブロック間で転送します。

5.4.6 IDMA チャネル 1 ステータス・レジスタ（IDMA1_STAT）

IDMA チャネル 1 ステータス・レジスタ（IDMA1_STAT）は、チャネルの動作状態を示します。転送が進行中（ACTV）か、転送が保留されているか（PEND）を示す 2 つのビットがあります。

IDMA チャネル 1 ステータス・レジスタ（IDMA1_STAT）を図 5-9 に示し、表 5-8 で説明します。

図 5-9. IDMA チャネル 1 ステータス・レジスタ（IDMA1_STAT）



凡例：R = リード専用。*-n* = リセット後の値。

表 5-8. IDMA チャネル 1 ステータス・レジスタ（IDMA1_STAT）フィールドの説明

ビット	フィールド	値	説明
31-2	Reserved	0	これらの予約ビット・ロケーションは、常にゼロとしてリードされます。このフィールドに値をライトしても影響はありません。
1	PEND		転送を保留します。コントロール・レジスタが CPU にライトし、かつアクティブな転送が進行中の場合、セットされます。また、転送がアクティブの場合、クリアされます。
		0	保留されている転送はありません。
		1	転送は保留されています。
0	ACTV		アクティブな転送。チャネル 1 がソース・アドレス・レジスタ（IDMA1_SOURCE）からデータのリードを開始している場合、ACTV ビットはセットされます。また、デスティネーション・アドレス・レジスタ（IDMA1_DEST）へ最後のライトが行われた後に、ACTV ビットはクリアされます。
		0	アクティブな転送はありません。
		1	アクティブな転送。

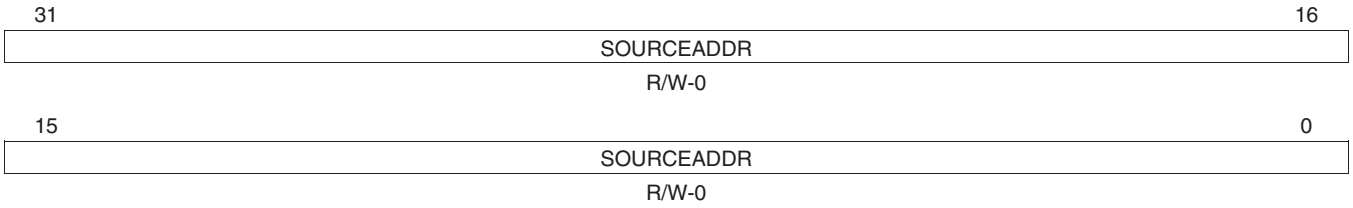
5.4.7 IDMA チャネル 1 ソース・アドレス・レジスタ（IDMA1_SOURCE）

IDMA チャネル 1 ソース・アドレス・レジスタ（IDMA1_SOURCE）は、IDMA 転送に対するソース・アドレスを示します。転送のソースは、C64x+ メガモジュールに対してローカルでなければならず、L1P、L1D、L2、CFG のいずれかに含まれている必要があります。またソース・アドレスは、EMC サイクルごとに 256 ビットの最大のスループットを取得するために、デスティネーション・アドレスとは異なるポート（L2 ポート 0 および L2 ポート 1 は同一ポートと見なされる）に対するアドレスでなければなりません。

データ転送ではなくブロック単位で埋め込みを行う場合（IDMA1_COUNT の FILL = 1）、IDMA1_SOURCE を使用して埋め込み値をプログラムします。ソース・アドレスからリードするのではなく、IDMA はプログラムされた値をデスティネーション・バッファのすべてのロケーションに転送します。

IDMA チャネル 1 ソース・アドレス・レジスタ（IDMA1_SOURCE）を図 5-10 に示し、表 5-9 で説明します。

図 5-10. IDMA チャネル 1 ソース・アドレス・レジスタ（IDMA1_SOURCE）



凡例：R/W = リード / ライト。*-n* = リセット後の値。

表 5-9. IDMA チャネル 1 ソース・アドレス・レジスタ（IDMA1_SOURCE）フィールドの説明

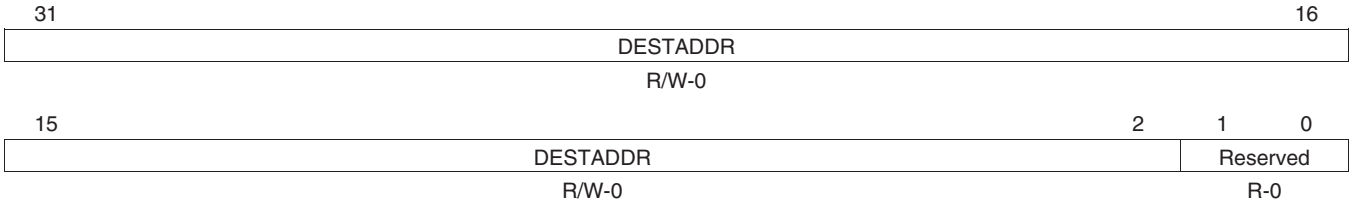
ビット	フィールド	値	説明
31-0	SOURCEADDR	0 ～ FFFF FFFFh	ソース・アドレス。C64x+ メガモジュールに対してローカルなワードにアラインされたメモリ・ロケーションを指す必要があります。ブロック単位で埋め込みを行う場合（IDMA1_COUNT の FILL = 1）、ソース・アドレスは埋め込み値です。埋め込みモード転送を行う場合、メモリ転送を行う場合に使用される SOURCEADDR フィールドのすべての 32 ビットを使用していることに注意してください。なお、2 つの LSB は 00b として実装されます。

5.4.8 IDMA チャネル 1 デスティネーション・アドレス・レジスタ（IDMA1_DEST）

IDMA チャネル 1 デスティネーション・アドレス・レジスタ（IDMA1_DEST）は、IDMA 転送に対するデスティネーション・アドレスを示します。転送のデスティネーションは、C64x+ メガモジュールに対してローカルでなければならず、L1P、L1D、L2、CFG のいずれかに含まれている必要があります。またデスティネーション・アドレスは、EMC サイクルごとに 256 ビットの最大のスループットを取得するために、ソース・アドレスとは異なるポート（L2 ポート 0 および L2 ポート 1 は同一ポートと見なされる）に対するアドレスでなければなりません。

IDMA チャネル 1 デスティネーション・アドレス・レジスタ（IDMA1_DEST）を図 5-11 に示し、表 5-10 で説明します。

図 5-11. IDMA チャネル 1 デスティネーション・アドレス・レジスタ（IDMA1_DEST）



凡例：R/W = リード / ライト。R = リード専用。-*n* = リセット後の値。

表 5-10. IDMA チャネル 1 デスティネーション・アドレス・レジスタ（IDMA1_DEST）フィールドの説明

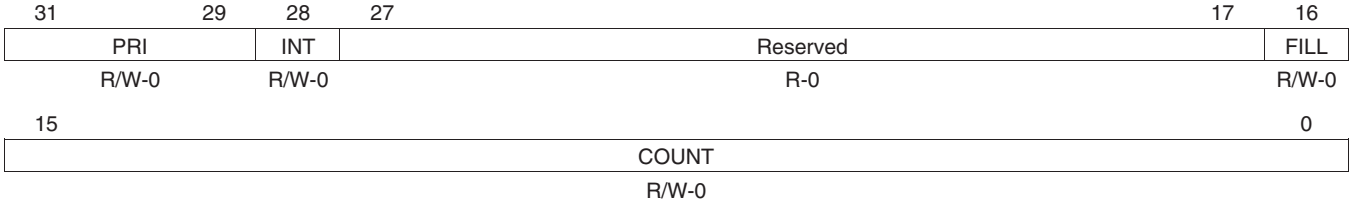
ビット	フィールド	値	説明
31-2	DESTADDR	0 ～ 3FFF FFFFh	デスティネーション・アドレス。C64x+ メガモジュールに対してローカルなワードにアラインされたメモリ・ロケーションを指す必要があります。
1-0	Reserved	0	予約。

5.4.9 IDMA チャネル 1 カウント・レジスタ（IDMA1_COUNT）

IDMA チャネル 1 カウント・レジスタ（IDMA1_COUNT）は、転送時間をバイト数で示します。また、IDMA1_COUNT レジスタは CPU 割り込み（IDMA_INT0）をイネーブルでき、指定された（CPU および他の DMA アクセスに対する）プライオリティ・レベルを示します。

IDMA チャネル 1 カウント・レジスタ（IDMA1_COUNT）を図 5-12 に示し、表 5-11 で説明します。

図 5-12. IDMA チャネル 1 カウント・レジスタ（IDMA1_COUNT）



凡例：R/W = リード / ライト。R = リード専用。-*n* = リセット後の値。

表 5-11. IDMA チャネル 1 カウント・レジスタ（IDMA1_COUNT）フィールドの説明

ビット	フィールド	値	説明
31-29	PRI	0 ～ 7h	転送のプライオリティ。競合が発生している場合に、CPU と DMA アクセス間のアービトレーションを行うために使用します。プライオリティは、0（最上位のプライオリティ）～ 7（最下位のプライオリティ）の範囲の値になることに注意してください。
28	INT		CPU 割り込みのイネーブル。
		0	転送完了時に CPU 割り込みを行いません。
		1	転送完了時に CPU 割り込み（IDMA_INT1）を行います。
27-17	Reserved	0	これらの予約ビット・ロケーションは、常にゼロとしてリードされます。このフィールドに値をライトしても影響はありません。
16	FILL		ブロック単位での埋め込み。
		0	ソース・アドレス・レジスタ（IDMA1_SOURCE）からデスティネーション・アドレス・レジスタ（IDMA1_DEST）へのブロック転送。
		1	デスティネーション・アドレス・レジスタ（IDMA1_DEST）が示すメモリ・バッファへの埋め込み値として、ソース・アドレス・レジスタ（IDMA1_SOURCE）を使用して、ブロック単位での埋め込みを行います。
15-0	COUNT	0 ～ FFh	バイト・カウント。転送時間をバイト数で指定した 16 ビット・カウント。4 バイトの倍数にする必要があります。転送回数がゼロということはデータをまったく転送していないが、INT ビットによりリクエストされた割り込みを生成したことを示します。正しい動作を行うために、2 つの LSB は常に 0 にしておく必要があります。

転送回数がゼロということは、プログラム上の考えられるオプションです。IDMA エンジンは転送をただちに「完了する」ことで、転送回数がゼロであることを処理します（つまり、実際にはデータが転送されていなかったとしても、IDMA1_COUNT レジスタの INT ビットは IDMA_INT1 を CPU にアサートします）。後続の転送が保留されている場合、ただちに開始されます。

5.5 権限レベルおよび IDMA 動作

表 5-12 に、役割ごとにアクセスできる IDMA レジスタおよびメガモジュールで行われる保護チェックの内容について概要を示します。

表 5-12. IDMA レジスタのアクセス権限

レジスタ	スーパーバイザ	ユーザ
IDMA0_STAT	R	R
IDMA0_MASK	R/W	R/W
IDMA0_SOURCE	R/W	R/W
IDMA0_DEST	R/W	R/W
IDMA0_COUNT	R/W	R/W
IDMA1_STAT	R	R
IDMA1_SOURCE	R/W	R/W
IDMA1_DEST	R/W	R/W
IDMA1_COUNT	R/W	R/W

第 6 章

JAJU058

帯域管理アーキテクチャ

本章では、TMS320C64x+ メガモジュールの帯域管理アーキテクチャについて説明します。

項目 **ページ**

6.1 はじめに

6.1.1 帯域管理の目的

帯域管理の目的は、一部のリクエスタが C64x+ メガモジュールで長期間にわたり使用可能なリソースをブロックしないことを確実にすることです。

C64x+ のメモリ保護機能と同様に、帯域管理（BWM）は（C64x+ メガモジュール全体で）グローバルに管理されていますが、C64x+ メガモジュールのリソースごとにローカルに実装されています。この目的を実現するために、帯域管理の初期化は、C64x+ メガモジュールのリソースそれぞれにあるレジスタをプログラムすることで構成されています。

6.1.2 帯域管理で保護されるリソース帯域

BWM 制御ハードウェアは、次の 4 つのリソースを管理します。

- レベル 1 プログラム（L1P）SRAM / キャッシュ
- レベル 1 データ（L1D）SRAM / キャッシュ
- レベル 2（L2）SRAM / キャッシュ
- メモリ・マップド・レジスタ・コンフィギュレーション・バス

6.1.3 帯域管理で管理されるリクエスタ

以下の項目はそれぞれ、6.1.2 項で示した C64x+ メガモジュールに対する潜在的なリクエスタです。

- CPU によって開始される転送：
  - データ・アクセス（ロード / ストアなど）
  - プログラムによるアクセス
- プログラム可能なキャッシュ・コヒーレンシ動作（ライトバックなど）：
  - ブロック・ベース
  - グローバル
- 内部 DMA（IDMA）によって開始される転送（およびその結果として生じるコヒーレンシ動作）
- 外部で開始されるスレーブ DMA（SDMA）転送（およびその結果として生じるコヒーレンシ動作）

6.1.4 用語と定義

本章で使用する用語と定義については、付録 A を参照してください。

6.2 アーキテクチャ

帯域管理方式は、重み付けプライオリティ制御による帯域割り当てとして見ることができます。

6.2.1 プライオリティ・レベルに基づく帯域割り当て

それぞれのリクエスタ（DMA、IDMA、CPU など）には、転送単位ごとにプライオリティ・レベルが割り当てられています。全部で 8 段階のプライオリティ・レベルがあります。以下にそのレベルを示します。

最上位	プライオリティ 0
	プライオリティ 1
	プライオリティ 2
	プライオリティ 3
	プライオリティ 4
	プライオリティ 5
	プライオリティ 6
	プライオリティ 7
最下位	プライオリティ 8

複数のリクエスタが単一リソースを求めて競合している場合、最上位のプライオリティが設定されたリクエスタにアクセス権を認めることで競合状態を解決します。複数の連続したサイクルで競合が発生している場合、競合カウンタによって、*n* 回のアービトレーション・サイクル 1 回ごとに、最下位のプライオリティが設定されたリクエスタがリソースにアクセスできるということが保証されます。ここで、*n* は MAXWAIT ビットでプログラム可能な値です（6.3 節を参照）。

BWM は、リソース・リクエストがブロックされるたびに競合カウンタをインクリメントすることで機能します。処理を進めるリクエストが許可されている場合、ストール・カウントは 0 にリセットされます。ストール・カウントが MAXWAIT 値に達すると、プライオリティの低いリクエスタの値は -1 にセットされ、少なくとも 1 回の転送を行うことができます（競合カウンタは、ユーザには見えません）。

6.2.2 プライオリティ・レベル：-1

前述の 9 段階のプライオリティ・レベルだけでなく、ハードウェアはプライオリティ・レベル -1 を使用して、競合カウンタの有効期限切れのためにプライオリティが高くなった転送、または最上位のプライオリティが設定された特定のリソースへの転送として固定された転送を表します。BWM アービトレーション・コントロール・レジスタに値 -1 をプログラムすることはできません。

6.2.3 プライオリティ宣言

さまざまな方法を使用して、リクエスタによるプライオリティを宣言します（表 6-1 を参照）。整合性を確保するために、BWM アービトレーション・レジスタで使用されているプライオリティ値は、対応するモジュール（IDMA など）で規定された値と等しくなるように重み付けされています。

表 6-1. プライオリティ宣言方法

リクエスタ	プライオリティの宣言場所
CPU	BWM アービトレーション・レジスタ（PRI ビット）
ユーザ定義によるキャッシュ・コヒーレンシ	固定プライオリティ
IDMA	IDMA 転送パラメータ
SDMA	外部システム・マスタ転送パラメータで指定

6.3 レジスタ

アービトレーション・レジスタと呼ばれるレジスタ・セットは、帯域管理アーキテクチャを実装しています。レジスタが実装されているブロックは、L1D、L2、および拡張メモリ・コントローラ（EMC）です。表 6-2 に、レジスタとそのベース・アドレスを示します。

表 6-2. アービトレーション・レジスタ

ブロック	略称	レジスタ名	アドレス	参照先
L1P	なし	該当なし	該当なし	該当なし
L1D	CPUARBD	CPU アービトレーション・コントロール・レジスタ	0184 1040h	6.3.1 項
	IDMAARBD	IDMA アービトレーション・コントロール・レジスタ	0184 1044h	6.3.3 項
	SDMAARBD	スレーブ DMA アービトレーション・コントロール・レジスタ	0184 1048h	6.3.4 項
	UCARBD	ユーザ・コヒーレンス・アービトレーション・コントロール・レジスタ	0184 104Ch	6.3.2 項
L2	CPUARBU	CPU アービトレーション・コントロール・レジスタ	0184 1000h	6.3.1 項
	IDMAARBU	IDMA アービトレーション・コントロール・レジスタ	0184 1004h	6.3.3 項
	SDMAARBU	スレーブ DMA アービトレーション・コントロール・レジスタ	0184 1008h	6.3.4 項
	UCARBU	ユーザ・コヒーレンス・アービトレーション・コントロール・レジスタ	0184 100Ch	6.3.2 項
EMC	CPUARBE	CPU アービトレーション・コントロール・レジスタ	0182 0200h	6.3.1 項
	IDMAARBE	IDMA アービトレーション・コントロール・レジスタ	0182 0204h	6.3.3 項
	SDMAARBE	スレーブ DMA アービトレーション・コントロール・レジスタ	0182 0208h	6.3.4 項
	MDMAARBE	マスタ DMA アービトレーション・コントロール・レジスタ	0182 020Ch	6.3.5 項

表 6-3 には、L1P のアービトレーション・レジスタはありません。実際、L1P にはプログラム可能な帯域を管理するレジスタはありません。ただし、L1P コントローラには固定帯域管理機能があります。

リソースごとにアービトレーション・レジスタ・セットがあることに注意してください。それぞれのレジスタが対応するリクエスタは異なります。

同一グループ（CPU、IDMA、SDMA、UC）に属するアービトレーション・レジスタは、等価なデフォルト値を備えています。CPUARB、IDMAARB、SDMAARB、UCARB の各レジスタを呼び出すことで、リクエスタは一般化されます（表 6-3 を参照）。

表 6-3. アービトレーション・レジスタのデフォルト値

		レジスタのビットのデフォルト値		レジスタの存在場所			
略称	レジスタ名	PRI	MAXWAIT	L1P	L1D	L2	EMC
CPUARB	CPU アービトレーション・コントロール・レジスタ	1	16	いいえ	はい	はい	はい
IDMAARB	IDMA アービトレーション･コントロール･レジスタ	該当なし	16	いいえ	はい	はい	はい
SDMAARB	スレーブ DMA アービトレーション・コントロール・レジスタ	該当なし	1	いいえ	はい	はい	はい
UCARB	ユーザ・コヒーレンス・アービトレーション・コントロール・レジスタ	該当なし	32	いいえ	はい	はい	いいえ
MDMAARB	マスタ DMA アービトレーション・コントロール・レジスタ	7	該当なし	いいえ	いいえ	いいえ	はい

CPUARB、IDMAARB、IDMAARB、UCARB のデフォルト値は、ほとんどのアプリケーションで十分なものです。これらのレジスタでは、C64x+ メガモジュールの内部でプライオリティを規定しています。MDMAARBE レジスタは、C64x+ メガモジュールの外部でデータ転送のプライオリティを規定しています。システム設計に応じて、MDMAARBE レジスタをプログラムする（6.3.5 項を参照）ことでそのプライオリティを変更することが必要になる場合があります。ほとんどの場合、MDMARBE をプログラムしてプライオリティを高く（小さな値を設定）します。

6.3.1 CPU アービトレーション・コントロール・レジスタ（CPUARBD、CPUARBU、CPUARBE）

CPU アービトレーション・コントロール・レジスタ（CPUARBD、CPUARBU、CPUARBE）は、CPU 動作の帯域管理を制御します。CPUARB レジスタを図 6-1 に示し、表 6-4 で説明します。CPU によって開始される転送は、2 つのコンポーネントで構成されています。

1. CPU は、L1P コントローラ、およびその結果として生じる L1P キャッシュ・コヒーレンシ動作（割り当て / 追い出しなど）へのプログラム・フェッチ転送を発行します。
2. CPU は、L1D コントローラに対してデータのロード / ストア転送を発行します。その結果として生じる L1D キャッシュ・コヒーレンシ動作（割り当て / 追い出し / ロング・ディスタンス・アクセス）は、L2 コントローラに対して順に発行されます。

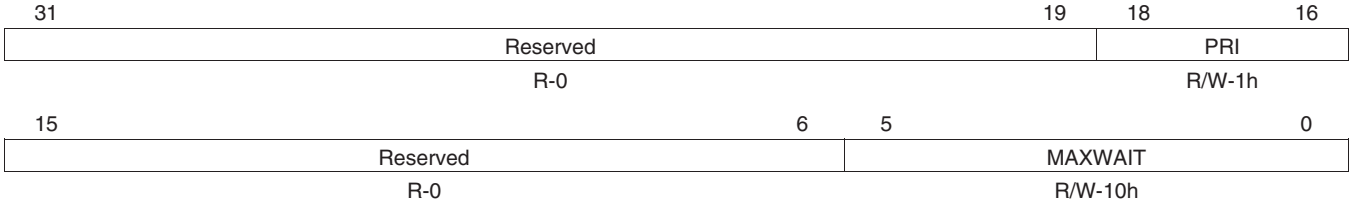
プログラムとデータの両方に対するリクエストは、CPUARB 値を使用して、最大のウェイト時間（MAXWAIT）およびプライオリティ（PRI）を規定します。CPUARB 値は、L1P または L1D に対してのみローカルな影響を与えるわけではありません。L1D/L1P キャッシュ・トランザクションに適用されるプライオリティ / 最大のウェイト時間は、ブロックごとにプログラムされます。これらの値を使用して、C64x+ メガモジュール内で対応するアクセスごとにアービトレーションを制御します。

（CPUARBD レジスタを使用して）L1D/L1P と同様に、（CPUARBU および CPUARBE レジスタをそれぞれ使用して）L2 および EMC ブロックで直接行われるメモリ・アクセスは、PRI および MAXWAIT のビット値をこれらのブロックに対してローカルに使用します。またこれらのリクエスタによって、さらなるトランザクションが生じます。

PRI のデフォルト値がセットされると、CPU トランザクションはシステム内で 2 番目に高いプライオリティが設定されます。これは、ほとんどのシステムで使用される比較的に一般的な値で、ほとんどの時間で、CPU のプライオリティが最も高くなります。しかし、（SDMA リクエストを処理するためにプライオリティが最も高い転送として通常プログラムされる）高速なシリアルポートなどリアルタイム・デッドラインが短いペリフェラルは、ほとんど時間をおかずに CPU 転送に割り込みをかけることができます。

CPU のプライオリティは、実行時にプログラム可能です。ただし、システム初期化時に CPUARB レジスタを初期化するか、デフォルト値を受け入れ、それ以降変更しないでおくこともできます。

図 6-1. CPU アービトレーション・コントロール・レジスタ（CPUARBD、CPUARBU、CPUARBE）



凡例：R/W = リード / ライト。R = リード専用。-*n* = リセット後の値。

表 6-4. CPU アービトレーション・コントロール・レジスタ（CPUARBD、CPUARBU、CPUARBE）フィールドの説明

ビット	フィールド	値	説明
31-19	Reserved	0	予約。
18-16	PRI	0 ～ 7h	プライオリティを示すフィールド。すべてのリクエスタが PRI = 8（最下位）をサポートしているとは限りません。PRI フィールドは、すべての他のリアルタイム・リクエストよりプライオリティが低いバックグラウンドで転送を行うために使用されます。
		0	プライオリティ 0（最上位）
		1h	プライオリティ 1
		2h	プライオリティ 2
		3h	プライオリティ 3
		4h	プライオリティ 4
		5h	プライオリティ 5
		6h	プライオリティ 6
		7h	プライオリティ 7（最下位）
15-6	Reserved	0	予約。
5-0	MAXWAIT	0 ～ 3Fh	EMC サイクルでの最大のウェイト時間。EMC サイクル = 2 倍の CPU サイクル。
		0	プライオリティがより高く設定されたリクエスタによって常にストールします。
		1h	最大のウェイト時間は 1 サイクル（1/2 = 50% アクセス）
		2h	最大のウェイト時間は 2 サイクル（1/3 = 33% アクセス）
		3h	予約。
		4h	最大のウェイト時間は 4 サイクル（1/5 = 20% アクセス）
		5h ～ 7h	予約。
		8h	最大のウェイト時間は 8 サイクル（1/9 = 11% アクセス）
		9h ～ Fh	予約。
		10h	最大のウェイト時間は 16 サイクル（1/17 = 6% アクセス）
		11h ～ 1Fh	予約。
		20h	最大のウェイト時間は 32 サイクル（1/33 = 3% アクセス）
		21h ～ 3Fh	予約。

6.3.2 ユーザ・コヒーレンス・アービトレーション・コントロール・レジスタ（UCARBD、UCARBU）

ユーザ・コヒーレンス・アービトレーション・コントロール・レジスタ（UCARBD、UCARBU）は、ユーザ・コヒーレンシ動作の帯域管理を制御します。これらの動作は、ユーザのプログラムで指定されるキャッシュ・ライトバックおよびキャッシュ・インバリデートのコマンドで構成されます。ユーザ・コヒーレンシ動作の詳細については、第 2 章、第 3 章、第 4 章をそれぞれ参照してください。

ユーザ・コヒーレンシ動作は、2 つのタイプに分類されます。システム内の他のリクエストに応じた固定されたプライオリティで示されます。

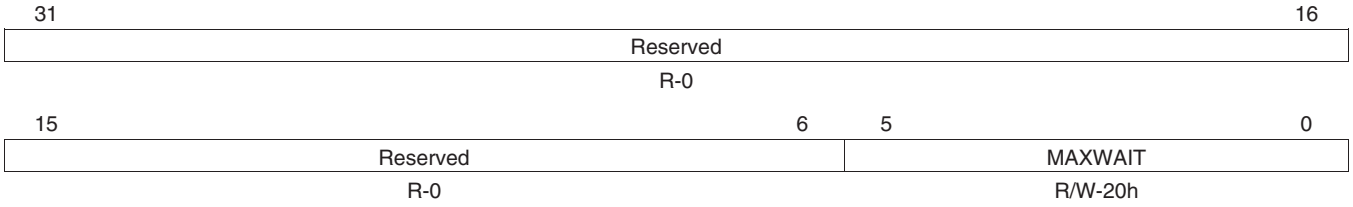
- グローバル・ユーザ・コヒーレンスは常にプライオリティが最も高くなる
- ブロック指向のコヒーレンスは常にプライオリティが最も低くなる

ユーザ・コヒーレンス・プライオリティが固定されているため、UCARB レジスタにはプライオリティを示す（PRI）ビットは含まれていません。グローバル・ユーザ・コヒーレンス動作は本質的にプライオリティが最も高く設定されているため、MAXWAIT がプログラム可能でもグローバル・キャッシュ動作には適用されず、ブロック指向のユーザ・コヒーレンス動作にのみ適用されます。ブロック指向のユーザ・コヒーレンシ動作は、L1D および L2 メモリの両方に影響を与えます。したがって、UCARB が存在するのは、L2（UCARBU）および L1D（UCARBD）のみです。

それぞれのレジスタに含まれる MAXWAIT ビット（およびそれに含まれるプライオリティ）は、DMA 転送または CPU トランザクションの結果として生じるコヒーレンシ動作のプライオリティを制御しません。

ユーザ・コヒーレンス・アービトレーション・コントロール・レジスタ（UCARBD、UCARBU）を図 6-2 に示し、表 6-5 で説明します。

図 6-2. ユーザ・コヒーレンス・アービトレーション・コントロール・レジスタ（UCARBD、UCARBU）



凡例：R/W = リード / ライト。R = リード専用。*-n* = リセット後の値。

表 6-5. ユーザ・コヒーレンス・アービトレーション・コントロール・レジスタ（UCARBD、UCARBU）フィールドの説明

ビット	フィールド	値	説明
31-6	Reserved	0	予約。
5-0	MAXWAIT	0 ～ 3Fh	EMC サイクルでの最大のウェイト時間。EMC サイクル = 2 倍の CPU サイクル。
		0	プライオリティがより高く設定されているリクエスタによって常にストールします。
		1h	最大のウェイト時間は 1 サイクル（1/2 = 50% アクセス）
		2h	最大のウェイト時間は 2 サイクル（1/3 = 33% アクセス）
		3h	予約。
		4h	最大のウェイト時間は 4 サイクル（1/5 = 20% アクセス）
		5h ～ 7h	予約。
		8h	最大のウェイト時間は 8 サイクル（1/9 = 11% アクセス）
		9h ～ Fh	予約。
		10h	最大のウェイト時間は 16 サイクル（1/17 = 6% アクセス）
		11h ～ 1Fh	予約。
		20h	最大のウェイト時間は 32 サイクル（1/33 = 3% アクセス）
		21h ～ 3Fh	予約。

6.3.3 IDMA アービトレーション・コントロール・レジスタ（IDMAARBD、IDMAARBU、IDMAARBE）

IDMA アービトレーション・コントロール・レジスタ（IDMAARBD、IDMAARBU、IDMAARBE）は、IDMA 動作を示す帯域管理を制御します。IDMA は、IDMA チャネル 0（メモリと CFG 空間の転送で使用）および IDMA チャネル 1（メモリ間の転送で使用）を使用した任意の時間内に行われるアクティブな 2 つの転送をサポートします。IDMA の動作に関する詳細については、第 5 章を参照してください。

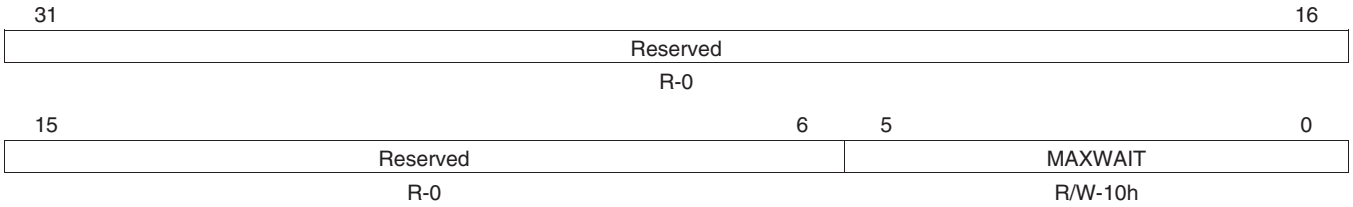
MAXWAIT フィールドを使用して、IDMA トランザクションに対する最大のウェイト時間を決定します。プライオリティ・レベルは、IDMAARB レジスタを使用してプログラムされているわけではありません。したがって、IDMAARB レジスタには PRI フィールドは含まれていません。その代わりに、プライオリティ・レベルは IDMA 転送パラメータの一部としてプログラムされています（つまり、IDMA コントロール・レジスタを直接使用します（第 5 章を参照））。要約すると、IDMA 転送プライオリティは次のようになります。

- IDMA チャネル 0 は、常にプライオリティが最も高くなる
- IDMA チャネル 1 のプライオリティは、IDMA チャネル 1 カウント・レジスタ（IDMA1_COUNT）の PRI フィールドを使用してプログラム可能

IDMA トランザクションは、L1D、L2、EMC の各リソースに影響を与えます。したがって、MAXWAIT フィールドは、L1D（IDMAARBD）、L2（IDMAARBU）、EMC（IDMAARBE）の各リソースに対して存在します。

IDMA アービトレーション・コントロール・レジスタ（IDMAARBD、IDMAARBU、IDMAARBE）を図 6-3 に示し、表 6-6 で説明します。

図 6-3. IDMA アービトレーション・コントロール・レジスタ（IDMAARBD、IDMAARBU、IDMAARBE）



凡例：R/W = リード / ライト。R = リード専用。-*n* = リセット後の値。

表 6-6. IDMA アービトレーション・コントロール・レジスタ（IDMAARBD、IDMAARBU、IDMAARBE）フィールドの説明

ビット	フィールド	値	説明
31-6	Reserved	0	予約。
5-0	MAXWAIT	0 ～ 3Fh	EMC サイクルでの最大のウェイト時間。EMC サイクル = 2 倍の CPU サイクル。
		0	プライオリティがより高く設定されているリクエスタによって常にストールします。
		1h	最大のウェイト時間は 1 サイクル（1/2 = 50% アクセス）
		2h	最大のウェイト時間は 2 サイクル（1/3 = 33% アクセス）
		3h	予約。
		4h	最大のウェイト時間は 4 サイクル（1/5 = 20% アクセス）
		5h ～ 7h	予約。
		8h	最大のウェイト時間は 8 サイクル（1/9 = 11% アクセス）
		9h ～ Fh	予約。
		10h	最大のウェイト時間は 16 サイクル（1/17 = 6% アクセス）
		11h ～ 1Fh	予約。
		20h	最大のウェイト時間は 32 サイクル（1/33 = 3% アクセス）
		21h ～ 3Fh	予約。

6.3.4 スレーブ DMA アービトレーション・コントロール・レジスタ（SDMAARBD、SDMAARBU、SDMAARBE）

スレーブ DMA アービトレーション・コントロール・レジスタ（SDMAARBD、SDMAARBU、SDMAARBE）は、スレーブ DMA（SDMA）動作を示す帯域管理を制御します。

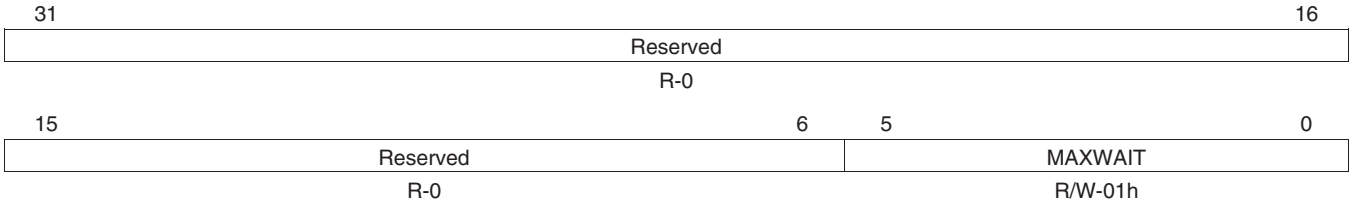
SDMA は、任意の時点で時間内に行われるアクティブな複数の転送をサポートできます。MAXWAIT フィールドは、すべてのスレーブ DMA トランザクションに対して最大のウェイト時間を制御します。プライオリティ・レベルは、SDMAARB レジスタを使用してプログラムされているわけではありません。したがって、SDMAARB レジスタには PRI フィールドは含まれていません。その代わりに、システム・マスタは、プライオリティ・レベルを指定します。C64x+ メガモジュールの外部のプライオリティ設定は DMA / チップ / ペリフェラル固有のため、プライオリティ・アロケーションに関する詳細については、各デバイスのデータシートなどを参照してください。

注： 外部で生成されるすべての DMA トランザクション（C64x+ メガモジュール外部から受信）に対する SDMA プライオリティは、その結果として生じるすべてのキャッシュ・コヒーレンス動作（スヌープ、スヌープ・ライト）を含む C64x+ メガモジュールから伝播されます。

SDMA トランザクションは、L1D、L2、EMC の各リソースに影響を与えます。したがって、MAXWAIT フィールドが、L1D（SDMAARBD）、L2（SDMAARBD）、EMC（SDMAARBE）の各リソースに対して存在します。

スレーブ DMA アービトレーション・コントロール・レジスタ（SDMAARBD、SDMAARBU、SDMARBE）を図 6-4 に示し、表 6-7 で説明します。

図 6-4. スレーブ DMA アービトレーション・コントロール・レジスタ（SDMAARBD、SDMAARBU、SDMARBE）



凡例：R/W = リード / ライト。R = リード専用。-*n* = リセット後の値。

表 6-7. スレーブ DMA アービトレーション・コントロール・レジスタ（SDMAARBD、SDMAARBU、SDMARBE）フィールドの説明

ビット	フィールド	値	説明
31-6	Reserved	0	予約。
5-0	MAXWAIT	0 ～ 3Fh	EMC サイクルでの最大のウェイト時間。EMC サイクル = 2 倍の CPU サイクル。
		0	プライオリティがより高く設定されているリクエスタによって常にストールします。
		1h	最大のウェイト時間は 1 サイクル（1/2 = 50% アクセス）
		2h	最大のウェイト時間は 2 サイクル（1/3 = 33% アクセス）
		3h	予約。
		4h	最大のウェイト時間は 4 サイクル（1/5 = 20% アクセス）
		5h-7h	予約。
		8h	最大のウェイト時間は 8 サイクル（1/9 = 11% アクセス）
		9h ～ Fh	予約。
		10h	最大のウェイト時間は 16 サイクル（1/17 = 6% アクセス）
		11h ～ 1Fh	予約。
		20h	最大のウェイト時間は 32 サイクル（1/33 = 3% アクセス）
		21h ～ 3Fh	予約。

6.3.5 マスタ DMA アービトレーション・コントロール・レジスタ（MDMAARBE）

マスタ DMA アービトレーション・コントロール・レジスタ（MDMAARBE）は、マスタ DMA（MDMA）動作の帯域管理を制御します。

プライオリティを示す（PRI）フィールドは、次のように発行されるプライオリティを制御します。

- MDMA トランザクション：キャッシュ・ミスまたは CFG 空間以外へのロング・ディスタンス・アクセスの結果
- コンフィギュレーション・バス・トランザクション：CFG 空間へのロング・ディスタンス・アクセスまたは CFG 空間への IDMA 転送

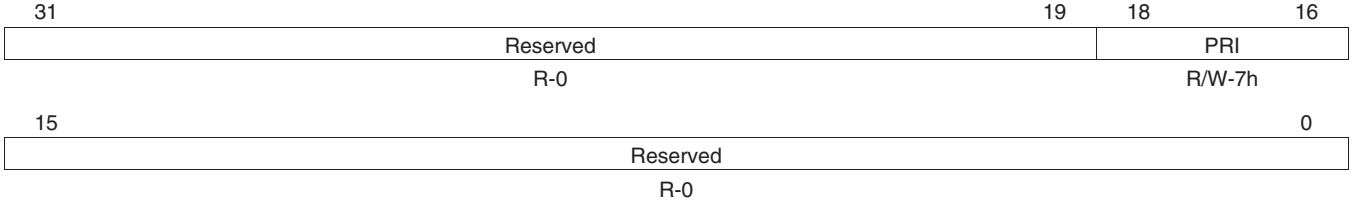
MDMAARBE プライオリティは、トランザクションには 2 つの部分があるという点でシステム内の他のプログラム可能なプライオリティとは異なります。以下に説明するように、各部分は別々に処理されます。

- 内部の転送の半数に対するアービトレーションは、他の章で規定したように、開始するリソース（CPU（L1P/L1D、ユーザ・コヒーレンス（L2）、IDMA など）によって異なります。
- 外部の転送の半数に対するアービトレーションは、DMA によって異なります。これは PRI フィールドで規定しています。PRI フィールドの値は、リソースに対する内部アービトレーションにはまったく影響しません。このプライオリティ値は、DMA マスタ・インターフェイスまたはコンフィギュレーション・インターフェイスを使用して開始されるすべてのトランザクションに対して使用されるだけです。

注： DMA アービトレーション・コントロール・レジスタ（MDMAARBE）によって生じる内部アービトレーションは存在しないため、MAXWAIT フィールドはありません。

マスタ DMA アービトレーション・コントロール・レジスタ（MDMAARBE）を図 6-5 に示し、表 6-8 で説明します。

図 6-5. マスタ DMA アービトレーション・コントロール・レジスタ（MDMAARBE）



凡例：R/W = リード / ライト。R = リード専用。-*n* = リセット後の値。

表 6-8. マスタ DMA アービトレーション・コントロール・レジスタ（MDMAARBE）フィールドの説明

ビット	フィールド	値	説明
31-19	Reserved	0	予約。
18-16	PRI	0 ～ 7h	プライオリティ・フィールド：すべてのリクエスタが PRI = 8（最下位）をサポートしているとは限りません。このフィールドは、すべての他のリアルタイム・リクエストよりプライオリティが低いバックグラウンドで転送を行うために使用されます。
		0	プライオリティ 0（最上位）
		1h	プライオリティ 1
		2h	プライオリティ 2
		3h	プライオリティ 3
		4h	プライオリティ 4
		5h	プライオリティ 5
		6h	プライオリティ 6
		7h	プライオリティ 7（最下位）
15-0	Reserved	0	予約。

6.4 権限およびバンドウィズ・マネージメント・レジスタ

表 6-9 に、役割（スーパーバイザまたはユーザ）ごとにアクセスできるバンドウィズ・マネージメント・レジスタをまとめました。

表 6-9. バンドウィズ・マネージメント・レジスタのアクセス権限

レジスタ	スーパーバイザ	ユーザ
CPUARBD	R/W	R
IDMAARBD	R/W	R
SDMAARBD	R/W	R
UCARBD	R/W	R
CPUARBU	R/W	R
IDMAARBU	R/W	R
SDMAARBU	R/W	R
UCARBU	R/W	R
CPUARBE	R/W	R
IDMAARBE	R/W	R
SDMAARBE	R/W	R
MDMAARBE	R/W	R

TEXAS INSTRUMENTS

第 7 章

JAJU058

割り込みコントローラ

本章では、割り込みコントローラについて説明します。

項目 **ページ**

7.1 はじめに

ここでは、割り込みコントローラの目的を説明し、その機能について解説します。

7.1.1 C64x+ 割り込みコントローラ（INTC）の目的

C64x+ システムは、幅広い種類のシステム・イベントを処理しています。割り込みコントローラは、必要なイベントを選択し、そのイベントを適切な CPU 割り込みおよび例外入力へ送る方法を用意しています。

ユーザはこれらの同じシステム・イベントの多くを使用して、EDMA などの他のペリフェラルをドライブしますが、メガモジュールの割り込みコントローラは CPU の管理専用です。

7.1.2 機能

注： ノンマスカブル割り込みは、すべての C6000 デバイスでサポートされているとは限りません。詳細については、各デバイスのデータ・マニュアルを参照してください。

割り込みコントローラは、システム・イベントを CPU の割り込みおよび例外入力に接続します。割り込みコントローラは、最大で 128 個のシステム・イベントをサポートします。

128 個のシステム・イベントは、割り込みコントローラへの入力として機能します。内部で生成された（メガモジュール内の）イベントとチップ・レベル・イベントの両方から構成されます。イベント・リストについては、7.3 節を参照してください。また、INTC レジスタは、これらの 128 個のシステム・イベントだけでなく、ノンマスカブル・リセット・イベントを受け取り、そのイベントを CPU に直接送ります。

割り込みコントローラは、次のイベント入力から C64x+ CPU に 17 個の信号を出力します。

- 1 つのマスカブル・ハードウェア例外（EXCEP）
- 12 個のマスカブル・ハードウェア割り込み（INT4 ～ INT15）
- 割り込みまたは例外（NMI）として使用可能な 1 つのノンマスカブル信号
- 1 つのリセット信号（RESET）

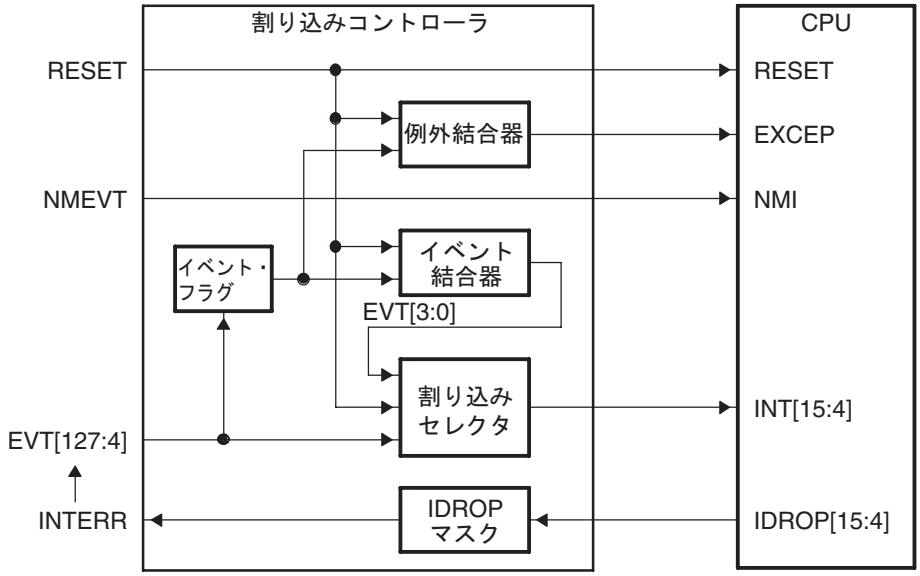
これらの CPU 割り込み / 例外信号の詳細については、『TMS320C64x/C64x+ DSP CPU and Instruction Set Reference Guide』（資料番号 SPRU732）を参照してください。

割り込みコントローラには、割り込みや例外へイベントを容易に送ることができるようにする次のモジュールが組み込まれています。

- 割り込みセレクタ：一部のシステム・イベントを 12 個のマスカブル割り込みへ送ります。
- イベント結合器：大量のシステム・イベントを 4 つのイベントに削減します。
- 例外結合器：一部のシステム・イベントを単一ハードウェア例外入力のためにグループにまとめます。

7.1.3 機能ブロック図

図 7-1. C64x+ メガモジュールに搭載されている割り込みコントローラのブロック図



7.1.4 用語と定義

本章で特に重要な用語は、次のとおりです。

システム・イベント：何らかの動作が発生し応答を必要としていることを CPU に通知することを目的とした内部または外部で生成する信号。

割り込み：外部または内部のハードウェア信号（イベント）が存在するため、通常のプログラム・フローをリダイレクトする手段を提供します。

例外は、プログラム・フローをリダイレクトするという点で割り込みとよく似ていますが、通常、例外はシステム内のエラー条件に関連付けられています。

本章で使用する用語の詳細な定義については、本書の付録 A および付録 B を参照してください。付録 A では、本書全体で使用している一般的な用語について説明し、付録 B ではメモリ / キャッシュ・アーキテクチャ関連の用語を定義しています。

7.2 割り込みコントローラのアーキテクチャ

C64x+ メガモジュールの割り込みコントローラは、システム・イベントを柔軟に管理できるように設計されています。この機能は、表 7-1 に示すレジスタ・セットを使用して実装されています。これらのレジスタについては、本章全体で説明します。これらのレジスタの詳細な説明については、7.5 節を参照してください。

表 7-1. インタラプト・コントローラ・レジスタ

レジスタ	説明	タイプ
EVTFLAG [3:0]	イベント・フラグ・レジスタ	ステータス
EVTCLR [3:0]	イベント・クリア・レジスタ	コマンド
EVTSET [3:0]	イベント・セット・レジスタ	コマンド
EVTMASK [3:0]	イベント・マスク・レジスタ	制御
MEVTFLAG [3:0]	マスクド・イベント・フラグ・レジスタ	ステータス
EXPMASK [3:0]	エクセプション・マスク・レジスタ	制御
MEXPFLAG [3:0]	マスクド・エクセプション・フラグ・レジスタ	ステータス
INTMUX [3:1]	インタラプト・マルチプレキシング・レジスタ	制御
INTXSTAT	インタラプト・エクセプション・ステータス・レジスタ	ステータス
INTXCLR	インタラプト・エクセプション・クリア・レジスタ	コマンド
INTDMASK	ドロップト・インタラプト・マスク・レジスタ	制御

7.2.1 イベント・レジスタ

割り込みコントローラには、コントローラ自身が受け取ったシステム・イベントのステータスを管理するレジスタ・セットがあります。レジスタは、次のグループにまとめることができます。

- イベント・フラグ・レジスタ（EVTFLAGx）
- クリア・フラグ・レジスタ（EVTCLRx）
- セット・フラグ・レジスタ（EVTSETx）

イベント・フラグ・レジスタは、割り込みコントローラが受け取ったすべてのシステム・イベントをキャプチャします。124 個のシステム・イベント入力をカバーするために、4 個の 32 ビット・レジスタがあります。それぞれのシステム・イベントは、イベント・フラグ・レジスタのいずれかの固有のフラグ・ビット（EFxx）にマップされます。

一般的なイベント・フラグ・レジスタの構造を図 7-2 に示します。

図 7-2. イベント・フラグ・レジスタの構造

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EF	EF	EF	EF	EF	EF	EF	EF	EF	EF	EF	EF	EF	EF	EF	EF
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EF	EF	EF	EF	EF	EF	EF	EF	EF	EF	EF	EF	EF	EF	EF	EF
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

凡例：R = リード専用。-*n* = リセット後の値。

124 個のシステム・イベントはすべて、4 個の 32 ビット・レジスタ EVTFLAGx のビットに個別にマップされます。これにより、EVTFLAG0 (EF03:EF00) の LSB 側の 4 つのビットはシステム・イベントに対応しないままになります。これらの 4 つのビットは予約されていて、常にゼロが入ります。つまり、これらのフィールドに対応するシステム・イベント入力はありません。その代わりに、イベント 00 ～ 03 に対応するシステム・イベントがイベント結合器により（割り込みコントローラに対して）内部で生成され、割り込みセレクタに送られます（図 7-1 を参照）。

イベント・フラグ（EFxx）は、ラッチされるレジスタ・ビットです。つまり、何らかのイベントを受信した場合、値 1 を保持します。EVTFLAGx レジスタはリード専用で、ライト専用のイベント・クリア・レジスタ EVTCLR[3:0] によってクリアされる必要があります。

イベント・クリア・レジスタを使用して、イベント・フラグ・レジスタをクリアします。4 個の 32 ビットのイベント・クリア・レジスタがあります。これらのレジスタのフィールドは、イベント・フラグ・レジスタのフィールドに 1 対 1 でマップします。イベント・クリア・レジスタの特定のフィールドに 1 をライトすると、対応するイベント・フラグ・レジスタのフィールドはクリアされます。

イベント・クリア・レジスタの構造を図 7-3 に示します。

図 7-3. イベント・クリア・レジスタの構造

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EC	EC	EC	EC	EC	EC	EC	EC	EC	EC	EC	EC	EC	EC	EC	EC
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EC	EC	EC	EC	EC	EC	EC	EC	EC	EC	EC	EC	EC	EC	EC	EC
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

凡例：W = ライト専用。-*n* = リセット後の値。

イベント・セット・レジスタは、概念的にイベント・クリア・レジスタによく似ています。イベント・セット・レジスタを使用して、イベント・フラグ・レジスタ内のいずれかのビットを手動でセットします（たとえば、割り込みサービス・ルーチンのテスト時に、イベント・セット・レジスタを使用して割り込みを生成することが有効な場合があります）。4 個の 32 ビットのイベント・セット・レジスタがあり、そのフィールドはイベント・フラグ・レジスタのフィールドに 1 対 1 でマップします。イベント・セット・レジスタの特定のフィールドに 1 をライトすると、対応するイベント・フラグ・レジスタのフィールドは 1 にセットされます。

イベント・セット・レジスタの構造を図 7-4 に示します。

図 7-4. イベント・セット・レジスタの構造

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ES	ES	ES	ES	ES	ES	ES	ES	ES	ES	ES	ES	ES	ES	ES	ES
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ES	ES	ES	ES	ES	ES	ES	ES	ES	ES	ES	ES	ES	ES	ES	ES
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

凡例：W = ライト専用。-*n* = リセット後の値。

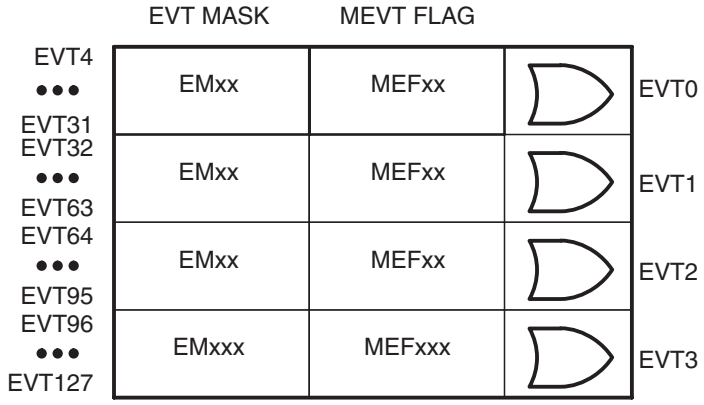
割り込みコントローラは、イベント・フラグ・レジスタに直接ライトするのではなく、イベント・クリア・レジスタおよびイベント・セット・レジスタを使用して潜在的な競合状態を回避します。このようなレジスタが他にない場合、フラグ・ビットをリード後にライトしてクリアする場合、CPU は誤ってセットされたイベント・フラグをクリアしてしまうこともあります。

新たなイベントを同じサイクルで受け取ると、EVTCLRx レジスタによってクリアが指定されれば、新たなイベント入力はイベントの取りこぼしを回避するために優先されます。

7.2.2 イベント結合器

イベント結合器（図 7-5）を使用すると、複数のシステム・イベントを組み合わせて単一イベントにすることができます。組み合わされたイベントは、割り込みセレクタへ送られます。これにより、CPU には使用可能な割り込みが 12 個しかありませんが、すべてのシステム・イベントを処理することができます。

図 7-5. イベント結合器



イベント結合器の基本概念は、システム・イベント・フラグのサブセット上で OR 演算を行うことです（表 7-2 を参照）。OR 演算を行った結果は、新たに「組み合わされた」イベントとして提供されます。

イベント結合器は、124 個のシステム・イベントを 4 つのグループに分類します。最初のグループにはイベント 4 ～ 31、2 番目のグループにはイベント 32 ～ 63、3 番目のグループにはイベント 64 ～ 95、4 番目のグループにはイベント 96 ～ 127 がそれぞれ含まれます。各グループ内のイベントを組み合わせて、新たに「組み合わされた」イベントを提供することができます。これらの新しいイベントは、それぞれ EVT0、EVT1、EVT2、EVT3 と表します。これらのイベントは、128 個のイベント全体を組み合わせたときの元の 124 個のシステム・イベントとともに割り込みセレクタに送られます。

各グループには、イベント・マスク・レジスタがあります。

イベント・マスク・レジスタの一般的な構造を図 7-6 に示します。

図 7-6. イベント・マスク・レジスタの構造

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EM	EM	EM	EM	EM	EM	EM	EM	EM	EM	EM	EM	EM	EM	EM	EM
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EM	EM	EM	EM	EM	EM	EM	EM	EM	EM	EM	EM	EM	EM	EM	EM
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

凡例：R/W = リード / ライト。*-n* = リセット後の値。

イベント・マスク・レジスタ内のイベント・マスク・ビットは、受け取ったシステム・イベントを組み合わせて、イネーブルまたはマスクするように動作します。レジスタの内容は、デフォルトではゼロです。そのため、すべてのシステム・イベントはマスクされず、組み合わされて、対応する EVTx になります。イベント・ソースをマスク・アウトする（つまり、イベントの組み合わせをディスエーブルする）には、対応するマスク・ビットを 1 にセットする必要があります。イベント 0 ～ 3 のイベント・マスクは予約されていて、常にマスクされていることに注意してください。

例 7-1.

```
Assume an application requires the events 124-127 to be combined.
       In order to accomplish this, EVTMASK3 will need to be programmed as follows:
                          EVTMASK3 = 00001111111111111111111111111111
```

イベント結合器は、プログラム可能なイベントの組み合わせに基づき組み合わされた出力イベントを生成するだけでなく、マスクされたイベント・フラグ・レジスタを示します。

マスクド・イベント・フラグ・レジスタの構造を図 7-7 に示します。

図 7-7. マスクド・イベント・フラグ・レジスタの 32 個のビット構造

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

凡例：R = リード専用。*-n* = リセット後の値。

マスクド・イベント・フラグ・レジスタの内容は、イベント・マスク・レジスタでイネーブルされたイベントの場合、イベント・フラグ・レジスタの内容と同じです。マスクド・イベント・フラグ・レジスタをリードすると、CPU には対応する組み合わされたイベント（EVT [3:0]）に関連するイベント・フラグだけが見えます。これは、組み合わされたイベントを処理する割り込みルーチンで有効です。

例 7-2.

```
Assuming the following configuration:
EVTFLAG3 = 01101010010011001110001110010101
EVTMASK3 = 00001111111111111111111111111111

The Masked Event Flag register 3 will be:
MEVTFLAG3 = 01100000000000000000000000000000
```

組み合わされた割り込みを処理する手順は、次のとおりです。

1. 組み合わされたイベント EVTx に対応する MEVTFLAGx レジスタをリードする
2. 最初にペンディングされた（フラグが付けられた）イベントをチェックする
3. この MEVTFLAGx 値を EVTCLRx レジスタへライトする
4. ステップ 2 で示されたイベントを処理する
5. MEVTFLAGx レジスタ = 0 となるまで、ステップ 1 ～ 4 を繰り返す

この手順では、EVTx で組み合わされたイベントの評価およびクリアのみを行います。さらに、EVTMASKx レジスタでマスクされた一部のイベントは、EVTFLAGx レジスタでセットされていても、クリアされません（かつ、イベントをクリアする必要もありません）（これにより、そのイベントを使用して、例外を生成することができます）。

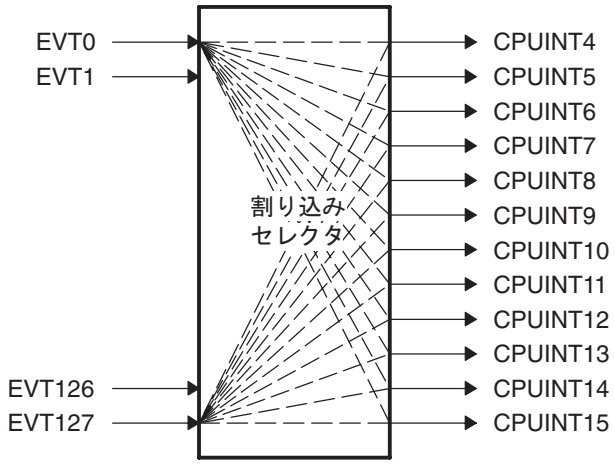
注： CPU は割り込みサービス・ルーチン内で復帰する前に、保留されているイベントがなくなるまでステップ 1 ～ 4 を繰り返します。これにより、割り込みサービス・ルーチン実行中に受け取る一部のイベントが確実にキャプチャされます（EVTCLRy [x] レジスタ内でそのフラグがクリアされるのと同時に、イベント EVTx を受け取ると、フラグはクリアされないということも忘れないでください）。

7.2.3 割り込みセレクタ

7.2.3.1 割り込みセレクタの動作

CPU には、12 個のマスカブル割り込み（CPUINT4 ～ CPUINT15）があります。割り込みセレクタを使用すると、128 個のシステム・イベントのいずれかを 12 個の CPU 割り込み入力のいずれかに送ることができます（図 7-8 を参照）。

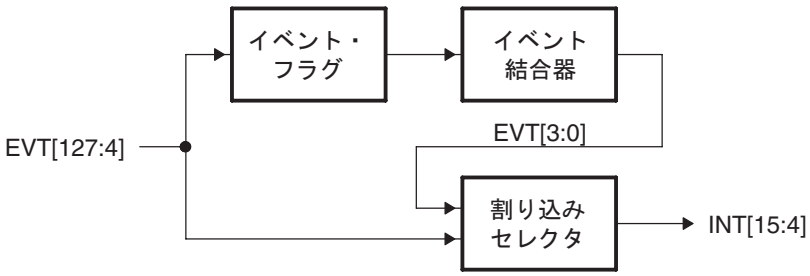
図 7-8. 割り込みセレクタのブロック図



128 個のシステム・イベントは、イベント入力またはイベント結合器によって生成されたイベントの組み合わせです。イベント結合器のロジックで、複数のイベント入力を 4 つのイベント出力グループにまとめることができます。その後、これらの出力は、割り込みセレクタに渡され、追加のシステム・イベントとして扱われます（EVT0 ～ EVT3）。

割り込みセレクタだけでなく、イベント結合器も使用すると、柔軟性の高い割り込みルーティング方式が可能になります。これにより、INTC モジュールは大量のシステム割り込みをメガモジュール内で処理できる柔軟性がもたらされます。また大量のシステム割り込みを CPU 内で同時に処理できるようにもなり、その結果、増え続ける割り込みを効率的に処理できます。

図 7-9. CPU による割り込みルーティングを示す図



割り込みセレクタにはインタラプト・マルチプレキシング・レジスタ INTMUX [3:1] があり、12 個の使用可能な CPU 割り込みごとに割り込みソースをプログラムすることができます。割り込みセレクタに渡されるイベントにはそれぞれ、これらのレジスタをプログラムするために使用されるイベント番号があります。

CPU 割り込み（CPUINT4 ～ CPUINT15）の順序によって、保留されている割り込みのプライオリティが決まります。割り込みサービス・ルーチンはアトミック（入れ子にはできない）のため、CPU 割り込みのプライオリティは保留されている割り込みにのみ適用されます。CPU の割り込み機能の詳細については、『TMS320C64x/C64x+ DSP CPU and Instruction Set Reference Guide』（資料番号 SPRU732）を参照してください。

7.2.3.2 割り込みエラー・イベント

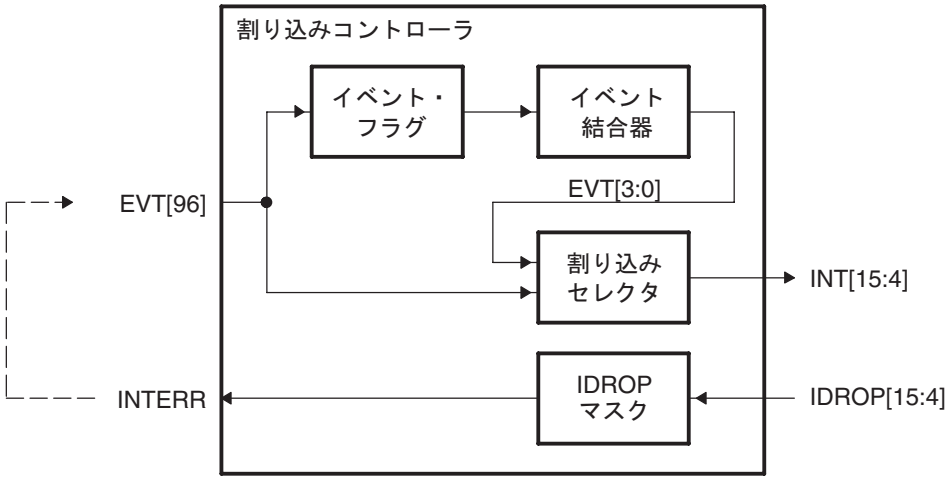
割り込みコントローラを備えた C64x+ CPU は、割り込みがドロップされたことを検出するたびに、システム・イベント（EVT96）を生成できます。このイベントが生成されるのは、対応する CPU の割り込みフラグ・ビットがすでにセットされている間に、CPU 割り込みを受け取ったときです。このエラー・イベントで、割り込みが長時間に渡りディスエーブルになっているかどうか、割り込みのできないコード・セクションが長すぎるかどうかなど、考えられるコードの問題をプログラマに示すことができます。

割り込みドロップ検出ロジックが CPU 内にあるため、単一システム・イベントから供給される割り込みだけが検出できます。組み合わされたイベントに基づいた割り込みがドロップされると、該当のグループに含まれる 1 つ以上の割り込みによってエラーが発生したことだけがわかります。

CPU が割り込みをドロップしたエラー状態を検出すると、その情報は割り込みコントローラのインタラプト・エクセプション・ステータス・レジスタ（INTXSTAT）に渡され、そこでドロップした割り込み番号が記録され、システム・イベントがアサートされます。このレジスタについては、7.5.3.2 項を参照してください。

例外生成に関連する信号を含むブロック図を図 7-10 に示します。

図 7-10. 割り込み例外イベントを示すブロック図



INTERR イベントが割り込みコントローラから出力され、内部でシステム・イベント EVT96 に送られます（図 7-10 を参照）。

INTXERR はドロップされた CPU ID を 1 つだけ保持できるため、最初にドロップされた割り込みを検出したことだけが INTERR（EVT96）からレポートされます。割り込み例外ステータスは、エクセプション・クリア・レジスタ（INTXCLR）によってクリアされます。このレジスタはクリア・ビットだけで構成されています。INTXCLR レジスタの CLEAR フィールドに 1 をライトすると、INTXSTAT レジスタを 0 にリセットします。ステータスがハードウェアでクリアされてから、新しく生成された IDROPx イベントだけが検出できます。

ドロップされた割り込みエラー・イベントを処理する場合、サービス・ルーチンは次のようになります。

1. INTXSTAT レジスタをリードします。
2. エラー状態をチェックします。
3. INTXCLR レジスタでエラーをクリアします。

1 つ以上の CPU 割り込みが発生してもドロップされた割り込みエラーを生成しないようにするには、ドロップト・インタラプト・マスク・レジスタ（INTDMASK）をプログラムしてそのエラーを無視します。

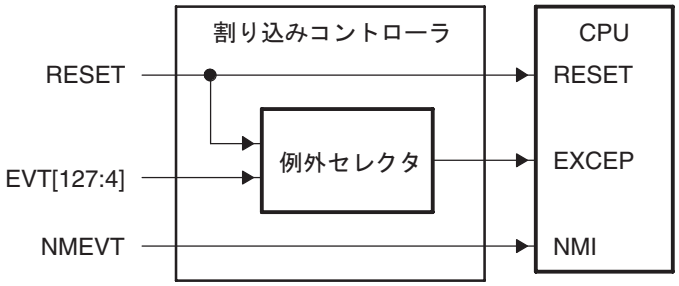
7.2.4 例外結合器

C64x+ CPU には、システム・レベルでマスカブル例外に対応した単一イベント入力があります。この入力は、EXCEP で表されます。例外結合器を使用すると、複数のシステム・イベントを組み合わせて単一の例外イベントにすることができます（図 7-12）。これにより、1 つの CPU 例外入力のみが使用可能な場合でも、CPU はすべてのシステム・イベントを処理することができます。

例外結合器を使用すると、システム設計者は EXCEP 値を判別するために OR 演算を行うシステム・イベント・フラグのサブセットを選択することができます。

例外結合器によって、システム例外をルーティングする様子を示すブロック図を 図 7-11 に示します。

図 7-11. システム例外をルーティングする様子を示す図



注： リセットや NMI もこの図に示されています。実際、例外が C64x+ CPU 内でイネーブルの場合、NMI 信号はノンマスカブル例外入力として使用されます。これらの 2 つの信号をさまざまな他の CPU 例外とともに CPU と組み合わせます。CPU 例外の詳細については、『TMS320C64x/C64x+ DSP CPU and Instruction Set Reference Guide』（資料番号 SPRU732）を参照してください。

システム・イベントのサブセットのみが CPU に対して例外を生成できるようにするために、例外結合器には 4 個のマスク・レジスタからなるセット EXPMASK[3:0] があります。これは、望ましくないイベントをディスエーブルにするために使われます。CPU に対して 1 つの例外入力だけがあるため、すべてのマスク・レジスタは、最大で 128 個のイベントを組み合わせて単一の EXCEP 出力にするために協調動作します。これにより、CPU はすべてのシステム例外を処理できます。

エクセプション・マスク・レジスタの一般的な構造を図 7-12 に示します。

図 7-12. エクセプション・マスク・レジスタの構造

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
XM	XM	XM	XM	XM	XM	XM	XM	XM	XM	XM	XM	XM	XM	XM	XM
R/W-FFFFh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XM	XM	XM	XM	XM	XM	XM	XM	XM	XM	XM	XM	XM	XM	XM	XM
R/W-FFFFh															

凡例：R/W = リード / ライト。*-n* = リセット後の値。

EXPMASKx レジスタのデフォルト値は、すべて 1 です。これは、すべてのイベントがマスクされていることを表します。したがって、このレジスタをプログラムしていない限り、システム・イベントにより例外が生成されることはありません。

イベント結合器（7.2.2 項を参照）と同様、例外結合器はエクセプション・マスク・レジスタと組み合わせて使用するマスクされた例外フラグ（MEXPFLAGx）のセットを提供します。マスクド・エクセプション・フラグ・レジスタは、マスクされたイベント・フラグ・レジスタ（7.2.1 項）を示します。マスクド・エクセプション・フラグ・レジスタをリードすると、CPU には CPU の EXCEP 入力に関連したイベント・フラグだけが見えます。

マスクド・エクセプション・レジスタの一般的な構造を図 7-13 に示します。

図 7-13. マスクド・エクセプション・フラグ・レジスタの構造

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MXF	MXF	MXF	MXF	MXF	MXF	MXF	MXF	MXF	MXF	MXF	MXF	MXF	MXF	MXF	MXF
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MXF	MXF	MXF	MXF	MXF	MXF	MXF	MXF	MXF	MXF	MXF	MXF	MXF	MXF	MXF	MXF
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

凡例：R = リード専用。*-n* = リセット後の値。

CPU は例外を受け取ると、例外の原因を判別し、適切なイベントに応答するために、例外サービス・ルーチンを実行します。例外処理時に、サービス・ルーチンは最初に例外が CPU 内部でノンマスカブル例外によって、あるいは EXCEP 信号によって生成されたのかを判別する必要があります。

EXCEP が例外の原因であるということが判明した場合、サービス・ルーチンではマスクド・エクセプション・フラグ・レジスタ（MEXPFLAG [3:0]）をリードし、例外のトリガになったマスクされていないイベントを決定します。

組み合わされた割り込みを処理する手順は、次のとおりです。

1. MEXPFLAG [3:0] レジスタをリードします。
2. 保留されているイベントを処理したかチェックします。
3. MEXPFLAG [3:0] 値を EVTCLR [3:0] レジスタへライトします。MEXPFLAGx 値を EVTCLRx レジスタと組み合わせて使用すると、EXCEP を生成するように組み合わされたこれらのイベントのみがクリアされます。EXPMASKx でマスクされた一部のイベントは、EVTFLAGx レジスタでセットされていてもクリアする必要はありません。これにより、そのイベントを使用して、組み合わされた割り込みイベントを生成することができます。
4. CPU は例外サービス・ルーチンから復帰する前に、保留されているイベントがなくなるまでステップ 1 ～ 3 を繰り返します。これにより、例外サービス・ルーチン実行中に受け取るすべてのイベントが確実にキャプチャされます。

注： CPU が新たな例外に対応することが必要な場合、ステップ 4 は重要です。このことが該当する理由を示す 2 つの事実があります。

- マスクされていないイベント・フラグ入力がアクティブの場合、例外結合器の出力はアクティブです。
- CPU は、例外リクエストを 0 から 1 への遷移として認識します。

したがって、すべてのマスクされないイベント・フラグは、CPU が新たな Low から High への遷移と認識できるようになる前に、クリアしておく必要があります。

7.3 C64x+ メガモジュールのイベント

C64x+ メガモジュールのさまざまなコンポーネントが生成する多くのイベントがあります。これらのイベントは、割り込みコントローラへ送られ、アサート時には CPU によって処理されます。これらのイベントをそのイベント・マッピングとともに表 7-2 に示します。

注： 使用可能なイベントとして示されたイベント（4 ～ 8、10、15 ～ 95）は、メガモジュールに対するチップ・レベル・イベント用です。したがって、それぞれの C64x+ デバイスは、これらのイベント入力を必要に応じて使用できます。これらの使用可能なイベントの使用方法の詳細については、各デバイスのデータ・マニュアルを参照してください。

表 7-2. システム・イベントのマッピング

EVT 番号	イベント	マッピング元	説明
0	EVT0	INT コントローラ	イベント結合器 0 の出力（イベント 1 ～ 31）。
1	EVT1	INT コントローラ	イベント結合器 1 の出力（イベント 32 ～ 63）。
2	EVT2	INT コントローラ	イベント結合器 2 の出力（イベント 64 ～ 95）。
3	EVT3	INT コントローラ	イベント結合器 3 の出力（イベント 96 ～ 127）。
4-8	使用可能なイベント。		
9	予約。		
10	使用可能なイベント。		
11-12	予約。		
13	IDMAINT0	EMC	IDMA チャネル 0 割り込み。
14	IDMAINT1	EMC	IDMA チャネル 1 割り込み。
15-95	使用可能なイベント。		
96	INTERR	INT コントローラ	ドロップされた CPU 割り込みイベント。
97	EMC_IDMAERR	EMC	無効な IDMA パラメータ。
98-117	予約。		
118	PDC_INT	PDC	PDC スリープ割り込み。
119	SYS_CMPA	SYS	CPU メモリ保護障害。
120	L1P_CMPA	L1P	CPU メモリ保護障害。
121	L1P_DMPA	L1P	DMA メモリ保護障害。
122	L1D_CMPA	L1D	CPU メモリ保護障害。
123	L1D_DMPA	L1D	DMA メモリ保護障害。
124	L2_CMPA	L2	CPU メモリ保護障害。
125	L2_DMPA	L2	DMA メモリ保護障害。
126	EMC_CMPA	EMC	CPU メモリ保護障害。
127	EMC_BUSERR	EMC	バス・エラー割り込み。

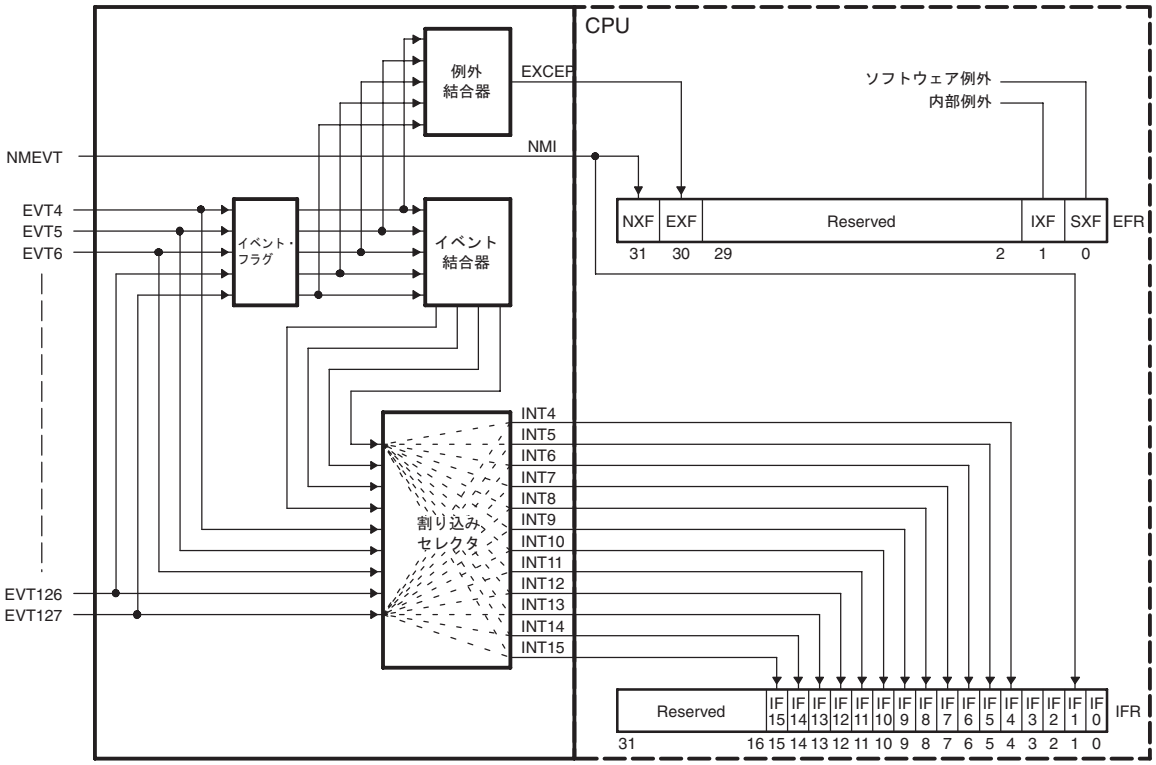
7.4 割り込みコントローラ - CPU との相互作用

7.4.1 CPU - 割り込みコントローラのインターフェイス

例外結合器および割り込みセレクタが生成する割り込みコントローラの出力先は、C64x+ CPU になります。

12 個の割り込み信号は、CPU のインタラプト・フラグ・レジスタ（IFR）に示されます（図 7-14 を参照）。

図 7-14. CPU によるイベント・ルーティングを示す図



CPU が割り込みを認識できるようにするために、割り込みをイネーブルします。CPU は、個々の割り込みをインタラプト・イネーブル・レジスタ（IER）およびインタラプト・タスク・レジスタのグローバル割り込みイネーブル・フィールド（ITSR.GIE）を使用してイネーブルする必要があります。

また、例外信号（EXCEP）が CPU のエクセプション・フラグ・レジスタ（EFR）（図 7-15）に記録されることにも注意してください。エクセプション・フラグ・レジスタ（EFR）が認識される前に、例外をイネーブルしておく必要があります。システム設計を容易にし、かつ下位互換性を確保するために、例外の認識は、デバイス・リセット後にディスエーブルされます。ITSR レジスタ（ITSR）のグローバル例外イネーブル・フィールド（GEE）をセットすることで、例外を有効にすることができます。モード（例外と割り込み）が変更中には NMI を受け取らないことを確実にするために、割り込みをイネーブルする前に、例外をイネーブルしておく必要があります。

システム例外が CPU でディスエーブルの場合、ノンマスカブル割り込み（NMI）は割り込みとして動作します。また、システム例外を受け取った場合、フラグが IFR レジスタの BIT1 フィールドにポストされます。ただし、システム例外が CPU でイネーブルの場合、このフラグはセットされません。むしろ、例外ソースが NMI、EXCEP、割り込み例外、ソフトウェア例外（SWE/SWENR）のいずれかを示すために、例外ソースはエクセプション・フラグ・レジスタ（EFR）で識別されます。

すべての NMI 処理は、NMI 割り込みベクタが割り込みとして使用されているか、例外を表しているかに関係なく、そのベクタを共有します。SWENR が SWE 命令ではなく例外を生成する場合に、CPU は、NMI ベクタに対立するベクタとして REP レジスタを使用します。

詳細については、『TMS320C64x/C64x+ DSP CPU and Instruction Set Reference Guide』（資料番号 SPRU732）を参照してください。

7.4.2 CPU による割り込みイベントの処理

CPU が単一イベントの割り込みを処理する（システム・イベントが割り込みセレクタ内で直接指定される）場合、割り込みコントローラのイベント・フラグ・レジスタ（EVTFLAGx）をリードまたはクリアする必要はありません。

ただし、組み合わされたシステム・イベントを処理する場合、イベント・フラグを割り込みサービス・ルーチンまたは例外サービス・ルーチン内で使用する必要があります。これらのフラグを使用して、割り込みまたは例外を開始したイベントを判別します。つまり、CPU のインタラプト・フラグ・レジスタ（またはエクセプション・フラグ・レジスタ）は CPU に対して組み合わされたイベントが発生したことを指示します。次に、サービス・ルーチンはイベント・フラグ・レジスタを使用して正確な原因を判別します。

また、後続のイベントを受け取るために、サービス・ルーチン内で、適切なイベント・フラグ・レジスタのビットをソフトウェアでクリアする必要があることに注意することも大切です。イベント・フラグがクリアされないと、新たなシステム・イベントは認識されません。新たなシステム・イベントをドロップされた割り込みとして認識することすらできません。これは、CPU によってドロップされた割り込みロジックは、（割り込みコントローラのイベント入力ではなく）CPU 割り込み入力に適用されるためです。イベントは割り込みコントローラで組み合わせるため、CPU からは見ることができません。

多くのシステムでは、サービス・ルーチンをリードして、イベント・フラグ・レジスタ（EVTFLAGx）全体をクリアしようとする可能性があります。この方法は一部のシステムでは適切に機能することもありますが、一部のイベント・フラグはシステムのどのコードからもポーリングされないということに注意する必要があります。特殊なイベントをポーリングする（そのイベントで CPU に割り込むのではなく、システム内の一部のコードを使用してときどきリードする）必要がある場合、イベント・フラグ・ビットをすべて無差別にクリアすると、期待しない結果がもたらされます。

7.5 レジスタ

C64x+ メガモジュールのインタラプト・コントローラ・レジスタを表 7-3 に示します。

表 7-3. インタラプト・コントローラ・レジスタ

アドレス	略称	レジスタの説明	参照先
0180 0000h ～ 0180 000Ch	EVTFLAG0	イベント・フラグ・レジスタ 0	7.5.1.1 項
	EVTFLAG1	イベント・フラグ・レジスタ 1	7.5.1.1 項
	EVTFLAG2	イベント・フラグ・レジスタ 2	7.5.1.1 項
	EVTFLAG3	イベント・フラグ・レジスタ 3	7.5.1.1 項
0180 0020h ～ 0180 002Ch	EVTSET0	イベント・セット・レジスタ 0	7.5.1.2 項
	EVTSET1	イベント・セット・レジスタ 1	7.5.1.2 項
	EVTSET2	イベント・セット・レジスタ 2	7.5.1.2 項
	EVTSET3	イベント・セット・レジスタ 3	7.5.1.2 項
0180 0040h ～ 0180 004Ch	EVTCLR0	イベント・クリア・レジスタ 0	7.5.1.3 項
	EVTCLR1	イベント・クリア・レジスタ 1	7.5.1.3 項
	EVTCLR2	イベント・クリア・レジスタ 2	7.5.1.3 項
	EVTCLR3	イベント・クリア・レジスタ 3	7.5.1.3 項
0180 0080h ～ 0180 008Ch	EVTMASK0	イベント・マスク・レジスタ 0	7.5.2.1 項
	EVTMASK1	イベント・マスク・レジスタ 1	7.5.2.1 項
	EVTMASK2	イベント・マスク・レジスタ 2	7.5.2.1 項
	EVTMASK3	イベント・マスク・レジスタ 3	7.5.2.1 項
0180 00A0h ～ 0180 00ACh	MEVTFLAG0	マスクド・イベント・フラグ・レジスタ 0	7.5.2.2 項
	MEVTFLAG1	マスクド・イベント・フラグ・レジスタ 1	7.5.2.2 項
	MEVTFLAG2	マスクド・イベント・フラグ・レジスタ 2	7.5.2.2 項
	MEVTFLAG3	マスクド・イベント・フラグ・レジスタ 3	7.5.2.2 項
0180 0104h ～ 0180 010Ch	INTMUX1	インタラプト・マルチプレキシング・レジスタ 1	7.5.3.1 項
	INTMUX2	インタラプト・マルチプレキシング・レジスタ 2	7.5.3.1 項
	INTMUX3	インタラプト・マルチプレキシング・レジスタ 3	7.5.3.1 項
0180 0180h	INTXSTAT	インタラプト・エクセプション・ステータス・レジスタ	7.5.3.2 項
0180 0184h	INTXCLR	インタラプト・エクセプション・クリア・レジスタ	7.5.3.3 項
0180 0188h	INTDMASK	ドロップト・インタラプト・マスク・レジスタ	7.5.3.4 項
0180 00C0h ～ 0180 00CCh	EXPMASK0	エクセプション・マスク・レジスタ 0	7.5.4.1 項
	EXPMASK1	エクセプション・マスク・レジスタ 1	7.5.4.1 項
	EXPMASK2	エクセプション・マスク・レジスタ 2	7.5.4.1 項
	EXPMASK3	エクセプション・マスク・レジスタ 3	7.5.4.1 項
0180 00E0h ～ 0180 00ECh	MEXPFLAG0	マスクド・エクセプション・フラグ・レジスタ 0	7.5.4.2 項
	MEXPFLAG1	マスクド・エクセプション・フラグ・レジスタ 1	7.5.4.2 項
	MEXPFLAG2	マスクド・エクセプション・フラグ・レジスタ 2	7.5.4.2 項
	MEXPFLAG3	マスクド・エクセプション・フラグ・レジスタ 3	7.5.4.2 項

7.5.1 イベント・レジスタ

割り込みコントローラには、コントローラ自身が受け取ったシステム・イベントを管理するためにステータス・レジスタとコントロール・レジスタのセットがあります。これには、128 個のシステム・イベントをすべてカバーするセット、フラグ、クリアの各レジスタが含まれています。

注： イベント・フラグ・ビット 0 ～ 3 は予約されていて、常に 0 が入ります。イベント・フラグ・レジスタに送られるこれらのフィールドに対応するイベントはありません。

7.5.1.1 イベント・フラグ・レジスタ（EVTFLAG*n*）

イベント・フラグ・レジスタ（EVTFLAG*n*）のイベント・フラグは、受け取った 128 個のシステム・イベントのいずれかに対して値 1 を保持します。また、このレジスタはリード専用です。ライト専用のイベント・クリア・レジスタ（EVTCLR*n*）を使用して、このレジスタをクリアします。イベント・セット・レジスタ（EVTSET*n*）を使用して手動で EVTFLAG*n* 内のいずかのビット（マスクされたビットを含む）をセットします。イベント・フラグ・レジスタ（EVTFLAG*n*）を図 7-15 ～ 図 7-18 に示し、表 7-4 で説明します。

図 7-15. イベント・フラグ・レジスタ 0（EVTFLAG0）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EF31	EF30	EF29	EF28	EF27	EF26	EF25	EF24	EF23	EF22	EF21	EF20	EF19	EF18	EF17	EF16
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EF15	EF14	EF13	EF12	EF11	EF10	EF9	EF8	EF7	EF6	EF5	EF4	EF3	EF2	EF1	EF0
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

凡例：R = リード専用。-*n* = リセット後の値。

図 7-16. イベント・フラグ・レジスタ 1（EVTFLAG1）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EF63	EF62	EF61	EF60	EF59	EF58	EF57	EF56	EF55	EF54	EF53	EF52	EF51	EF50	EF49	EF48
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EF47	EF46	EF45	EF44	EF43	EF42	EF41	EF40	EF39	EF38	EF37	EF36	EF35	EF34	EF33	EF32
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

凡例：R = リード専用。-*n* = リセット後の値。

図 7-17. イベント・フラグ・レジスタ 2（EVTFLAG2）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EF95	EF94	EF93	EF92	EF91	EF90	EF89	EF88	EF87	EF86	EF85	EF84	EF83	EF82	EF81	EF80
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EF79	EF78	EF77	EF76	EF75	EF74	EF73	EF72	EF71	EF70	EF69	EF68	EF67	EF66	EF65	EF64
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

凡例：R = リード専用。-*n* = リセット後の値。

図 7-18. イベント・フラグ・レジスタ 3（EVTFLAG3）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EF127	EF126	EF125	EF124	EF123	EF122	EF121	EF120	EF119	EF118	EF117	EF116	EF115	EF114	EF113	EF112
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EF111	EF110	EF109	EF108	EF107	EF106	EF105	EF104	EF103	EF102	EF101	EF100	EF99	EF98	EF97	EF96
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

凡例：R = リード専用。*-n* = リセット後の値。

表 7-4. イベント・フラグ・レジスタ（EVTFLAG*n*）フィールドの説明

ビット	フィールド	値	説明
31-0	EFyyy		イベント EVTyyy の状態をキャプチャします。
		0	EVTyyy は発生しませんでした。
		1	EVTyyy が発生しました。

7.5.1.2 イベント・セット・レジスタ（EVTSET*n*）

イベント・セット・レジスタ（EVTSET*n*）を使用して手動でイベント・セット・レジスタ（EVTFLAG*n*）内のいずかのビットをセットします。

イベント・セット・レジスタ（EVTSET*n*）を図 7-19 ～ 図 7-22 に示し、表 7-5 で説明します。

図 7-19. イベント・セット・レジスタ 0（EVTSET0）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ES31	ES30	ES29	ES28	ES27	ES26	ES25	ES24	ES23	ES22	ES21	ES20	ES19	ES18	ES17	ES16
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ES15	ES14	ES13	ES12	ES11	ES10	ES9	ES8	ES7	ES6	ES5	ES4	ES3	ES2	ES1	ES0
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

凡例：W = ライト専用。-*n* = リセット後の値。

図 7-20. イベント・セット・レジスタ 1（EVTSET1）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ES63	ES62	ES61	ES60	ES59	ES58	ES57	ES56	ES55	ES54	ES53	ES52	ES51	ES50	ES49	ES48
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ES47	ES46	ES45	ES44	ES43	ES42	ES41	ES40	ES39	ES38	ES37	ES36	ES35	ES34	ES33	ES32
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

凡例：W = ライト専用。-*n* = リセット後の値。

図 7-21. イベント・セット・レジスタ 2（EVTSET2）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ES95	ES94	ES93	ES92	ES91	ES90	ES89	ES88	ES87	ES86	ES85	ES84	ES83	ES82	ES81	ES80
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ES79	ES78	ES77	ES76	ES75	ES74	ES73	ES72	ES71	ES70	ES69	ES68	ES67	ES66	ES65	ES64
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

凡例：W = ライト専用。-*n* = リセット後の値。

図 7-22. イベント・セット・レジスタ 3（EVTSET3）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ES127	ES126	ES125	ES124	ES123	ES122	ES121	ES120	ES119	ES118	ES117	ES116	ES115	ES114	ES113	ES112
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ES111	ES110	ES109	ES108	ES107	ES106	ES105	ES104	ES103	ES102	ES101	ES100	ES99	ES98	ES97	ES96
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

凡例：W = ライト専用。-*n* = リセット後の値。

表 7-5. イベント・セット・レジスタ（EVTSET*n*）フィールドの説明

ビット	フィールド	値	説明
31-0	ESyyy		イベント・フラグ・レジスタ（EVTFLAG*n*）のフィールド EFyyy をセットします。
		0	影響なし。
		1	EFyyy = 1 をセットします。

7.5.1.3 イベント・クリア・レジスタ（EVTCLR*n*）

イベント・クリア・レジスタ（EVTCLR*n*）を使用して、イベント・フラグ・レジスタ（EVTFLAG*n*）のイベント・フラグをクリアします。

イベント・クリア・レジスタ（EVTCLR*n*）を図 7-23 ～ 図 7-26 に示し、表 7-6 で説明します。

図 7-23. イベント・クリア・レジスタ 0（EVTCLR0）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EC31	EC30	EC29	EC28	EC27	EC26	EC25	EC24	EC23	EC22	EC21	EC20	EC19	EC18	EC17	EC16
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EC15	EC14	EC13	EC12	EC11	EC10	EC9	EC8	EC7	EC6	EC5	EC4	EC3	EC2	EC1	EC0
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

凡例：W = ライト専用。-*n* = リセット後の値。

図 7-24. イベント・クリア・レジスタ 1（EVTCLR1）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EC63	EC62	EC61	EC60	EC59	EC58	EC57	EC56	EC55	EC54	EC53	EC52	EC51	EC50	EC49	EC48
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EC47	EC46	EC45	EC44	EC43	EC42	EC41	EC40	EC39	EC38	EC37	EC36	EC35	EC34	EC33	EC32
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

凡例：W = ライト専用。-*n* = リセット後の値。

図 7-25. イベント・クリア・レジスタ 2（EVTCLR2）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EC95	EC94	EC93	EC92	EC91	EC90	EC89	EC88	EC87	EC86	EC85	EC84	EC83	EC82	EC81	EC80
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EC79	EC78	EC77	EC76	EC75	EC74	EC73	EC72	EC71	EC70	EC69	EC68	EC67	EC66	EC65	EC64
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

凡例：W = ライト専用。-*n* = リセット後の値。

図 7-26. イベント・クリア・レジスタ 3（EVTCLR3）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EC127	EC126	EC125	EC124	EC123	EC122	EC121	EC120	EC119	EC118	EC117	EC116	EC115	EC114	EC113	EC112
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EC111	EC110	EC109	EC108	EC107	EC106	EC105	EC104	EC103	EC102	EC101	EC100	EC99	EC98	EC97	EC96
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

凡例：W = ライト専用。-*n* = リセット後の値。

表 7-6. イベント・クリア・レジスタ（EVTCLR*n*）フィールドの説明

ビット	フィールド	値	説明
31-0	ECyyy		イベント・フラグ・レジスタ（EVTFLAG*n*）のフィールド EFyyy をクリアします。
		0	影響なし。
		1	EFyyy = 0 をセットします。

7.5.2 イベント・コンバイナ・レジスタ

イベント結合器をプログラムするために、イベント・マスク・レジスタ（EVTMASK [3:0]）のセットがあります。これらのレジスタは、最大で 32 個のイベントを組み合わせて、割り込みコントローラが使用できる単一イベントにすることができます。EVTMASK [3:0] レジスタ内のイベント・マスク・ビットは、受け取ったシステム・イベントをマスク（またはイネーブル）するように動作します。割り込みセレクタ（EVT [3:0]）に渡される 4 つのイベント信号があります。

イベント・マスク・レジスタを以下に示します（ビット EM [3:0] は使用しません）。

7.5.2.1 イベント・マスク・レジスタ（EVTMASK*n*）

イベント結合器をプログラムするために、イベント・マスク・レジスタ（EVTMASK [3:0]）のセットがあります。これらのレジスタは、最大で 32 個のイベントを組み合わせて、1 つの CPU 割り込みまたは AET イベントとして使用する単一のイベント出力にすることができます。EVTMASK*n* レジスタ内のイベント・マスク・ビットは、イベント出力時に受け取ったシステム・イベントを組み合わせるイネーブラとして動作します。イベントおよび AET イベント・セレクタ（EVT [3:0]）への 4 つのイベント出力があります。

イベント・マスク・レジスタ（EVTMASK*n*）を図 7-27 ～ 図 7-30 に示し、表 7-7 で説明します。

図 7-27. イベント・マスク・レジスタ 0（EVTMASK0）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EM31	EM30	EM29	EM28	EM27	EM26	EM25	EM24	EM23	EM22	EM21	EM20	EM19	EM18	EM17	EM16
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EM15	EM14	EM13	EM12	EM11	EM10	EM9	EM8	EM7	EM6	EM5	EM4	EM3	EM2	EM1	EM0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-1	R-1	R-1	R-1

凡例：R/W = リード / ライト。R = リード専用。-*n* = リセット後の値。

図 7-28. イベント・マスク・レジスタ 1（EVTMASK1）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EM63	EM62	EM61	EM60	EM59	EM58	EM57	EM56	EM55	EM54	EM53	EM52	EM51	EM50	EM49	EM48
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EM47	EM46	EM45	EM44	EM43	EM42	EM41	EM40	EM39	EM38	EM37	EM36	EM35	EM34	EM33	EM32
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

凡例：R/W = リード / ライト。-*n* = リセット後の値。

図 7-29. イベント・マスク・レジスタ 2（EVTMASK2）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EM95	EM94	EM93	EM92	EM91	EM90	EM89	EM88	EM87	EM86	EM85	EM84	EM83	EM82	EM81	EM80
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EM79	EM78	EM77	EM76	EM75	EM74	EM73	EM72	EM71	EM70	EM69	EM68	EM67	EM66	EM65	EM64
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

凡例：R/W = リード / ライト。-*n* = リセット後の値。

図 7-30. イベント・マスク・レジスタ 3（EVTMASK3）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EM127	EM126	EM125	EM124	EM123	EM122	EM121	EM120	EM119	EM118	EM117	EM116	EM115	EM114	EM113	EM112
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EM111	EM110	EM109	EM108	EM107	EM106	EM105	EM104	EM103	EM102	EM101	EM100	EM99	EM98	EM97	EM96
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

凡例：R/W = リード / ライト。-*n* = リセット後の値。

表 7-7. イベント・マスク・レジスタ（EVTMASK*n*）フィールドの説明

ビット	フィールド	値	説明
31-0	EMyyy		イベント結合器への入力としてのイベント EVTyyy の使用をディスエーブルします。
		0	EVTyyy を組み合わせます。
		1	EVTyyy の組み合わせをディスエーブルします。

7.5.2.2 マスクド・イベント・フラグ・レジスタ（MEVTFLAG*n*）

イベント結合器には、4 個のマスクド・イベント・フラグ・レジスタのセットがあります（マスクされたイベント・フラグ・レジスタ）。

マスクド・イベント・フラグ・レジスタ（MEVTFLAG*n*）を図 7-31 ～ 図 7-34 に示し、表 7-8 で説明します。

図 7-31. マスクド・イベント・フラグ・レジスタ 0（MEVTFLAG0）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

凡例：R = リード専用。-*n* = リセット後の値。

図 7-32. マスクド・イベント・フラグ・レジスタ 1（MEVTFLAG1）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

凡例：R = リード専用。-*n* = リセット後の値。

図 7-33. マスクド・イベント・フラグ・レジスタ 2（MEVTFLAG2）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

凡例：R = リード専用。-*n* = リセット後の値。

図 7-34. マスクド・イベント・フラグ・レジスタ 3（MEVTFLAG3）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF	MEF
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

凡例：R = リード専用。*-n* = リセット後の値。

表 7-8. マスクド・イベント・フラグ・レジスタ（MEVTFLAG*n*）フィールドの説明

ビット	フィールド	値	説明
31-0	MEFyyy	0 〜 FFFF FFFFh	イベント・マスク・レジスタ（EVTMASK*n*）で EMyyy = 0 の場合、EFyyy の内容を示します。 (EMyyy = 0) の場合 MEFyyy = Efyyy それ以外の場合 MEFyyy = 0

7.5.3 CPU インタラプト・セレクタ・レジスタ

7.5.3.1 インタラプト・マルチプレキシング・レジスタ（INTMUX*n*）

割り込みセレクタにはインタラプト・マルチプレキシング・レジスタがあり、12 個の使用可能な CPU 割り込みごとにソースをプログラムすることができます。

インタラプト・マルチプレキシング・レジスタ（INTMUX*n*）を図 7-35 ～ 図 7-37 に示し、表 7-9 で説明します。

図 7-35. インタラプト・マルチプレキシング・レジスタ 1（INTMUX1）

31	30 24	23	22 16
Reserved	INTSEL7	Reserved	INTSEL6
R-0	R/W-7h	R-0	R/W-6h

15	14 8	7	6 0
Reserved	INTSEL5	Reserved	INTSEL4
R-0	R/W-5h	R-0	R/W-4h

凡例：R/W = リード / ライト。R = リード専用。-*n* = リセット後の値。

図 7-36. インタラプト・マルチプレキシング・レジスタ 2（INTMUX2）

31	30 24	23	22 16
Reserved	INTSEL11	Reserved	INTSEL10
R-0	R/W-Bh	R-0	R/W-Ah

15	14 8	7	6 0
Reserved	INTSEL9	Reserved	INTSEL8
R-0	R/W-9h	R-0	R/W-8h

凡例：R/W = リード / ライト。R = リード専用。-*n* = リセット後の値。

図 7-37. インタラプト・マルチプレキシング・レジスタ 3（INTMUX3）

31	30 24	23	22 16
Reserved	INTSEL15	Reserved	INTSEL14
R-0	R/W-Fh	R-0	R/W-Eh

15	14 8	7	6 0
Reserved	INTSEL13	Reserved	INTSEL12
R-0	R/W-Dh	R-0	R/W-Ch

凡例：R/W = リード / ライト。R = リード専用。-*n* = リセット後の値。

表 7-9. インタラプト・マルチプレキシング・レジスタ（INTMUX*n*）フィールドの説明

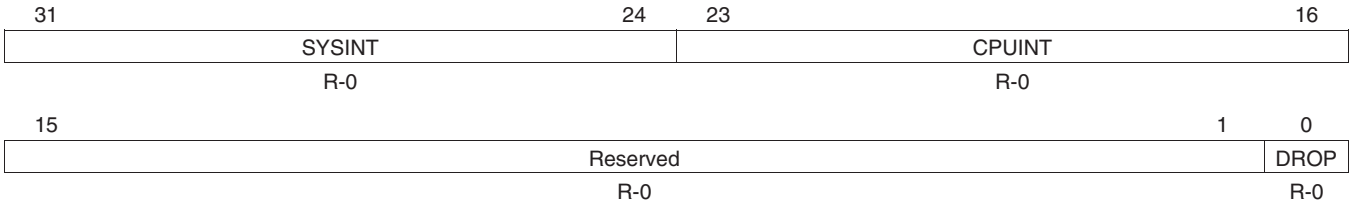
フィールド	値	説明
INTSELnn	0 ～ 7Fh	CPUINTnn にマップされるイベント番号が含まれます。

7.5.3.2 インタラプト・エクセプション・ステータス・レジスタ（INTXSTAT）

インタラプト・エクセプション・ステータス・レジスタ（INTXSTAT）は、生成された例外の原因となるイベントを判別するための情報を提供します。INTXSTAT レジスタは、CPU 割り込みおよびドロップされたイベントのシステム・イベント番号を保持します。

インタラプト・エクセプション・ステータス・レジスタ（INTXSTAT）を図 7-38 に示し、表 7-10 で説明します。

図 7-38. インタラプト・エクセプション・ステータス・レジスタ（INTXSTAT）



凡例：R = リード専用。*-n* = リセット後の値。

表 7-10. インタラプト・エクセプション・ステータス・レジスタ（INTXSTAT）フィールドの説明

ビット	フィールド	値	説明
31-24	SYSINT	0 〜 FFh	システム・イベント番号。
		0 〜 7Fh	EVT0 〜 EVT128
		80h 〜 FFh	予約。
23-16	CPUINT	0 〜 FFh	CPU 割り込み番号。
		0 〜 Fh	CPUINT0 〜 CPUINT15
		10h 〜 FFh	予約。
15-1	Reserved	0	予約。
0	DROP		ドロップされたイベント・フラグ。
		0	ドロップされたイベントはありません。
		1	CPU がイベントをドロップしました。

7.5.3.3 インタラプト・エクセプション・クリア・レジスタ（INTXCLR）

割り込み例外ステータスは、エクセプション・クリア・レジスタによってクリアされます。なお、下記に示すように基本的に 1 つのクリア・ビットで行われます。ステータスがクリアされると、新たな IDROPx イベントのみがハードウェアで検出可能になります。

インタラプト・エクセプション・クリア・レジスタ（INTXCLR）を図 7-39 に示し、表 7-11 で説明します。

図 7-39. インタラプト・エクセプション・クリア・レジスタ（INTXCLR）

31	1
Reserved	
R-0	
0	
CLEAR	
W-0	

凡例：R = リード専用。W = ライト専用。*-n* = リセット後の値。

表 7-11. インタラプト・エクセプション・クリア・レジスタ（INTXCLR）フィールドの説明

ビット	フィールド	値	説明
31-1	Reserved	0	予約。
0	CLEAR		割り込み例外ステータスをクリアします。
		0	影響なし。
		1	割り込み例外ステータスをクリアします。

7.5.3.4 ドロップト・インタラプト・マスク・レジスタ（INTDMASK）

INTERR イベントを生成するドロップされた割り込みは、マスク・レジスタでフィルタリングできます。ドロップ検出ハードウェアで無視されるこれらの CPU 割り込みは、ドロップト・インタラプト・マスク・レジスタ（INTDMASK）でマスクできます。

ドロップト・インタラプト・マスク・レジスタ（INTDMASK）を図 7-40 に示し、表 7-12 で説明します。

図 7-40. ドロップト・インタラプト・マスク・レジスタ（INTDMASK）

31													16
Reserved													
R-0													
15	14	13	12	11	10	9	8	7	6	5	4	3	0
IDM15	IDM14	IDM13	IDM12	IDM11	IDM10	IDM9	IDM8	IDM7	IDM6	IDM5	IDM4	Reserved	
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	

凡例：R = リード専用。*-n* = リセット後の値。

表 7-12. ドロップト・インタラプト・マスク・レジスタ（INTDMASK）フィールドの説明

ビット	フィールド	値	説明
31-16	Reserved	0	予約。
15-4	IDMnn		ドロップ検出ハードウェアによる CPUINTnn の検出をディスエーブルします。
		0	影響なし。
		1	ドロップ検出ハードウェアによる CPUINTnn の検出は無視されます。
3-0	Reserved	0	予約。

7.5.4 CPU エクセプション・レジスタ

7.5.4.1 CPU エクセプション・コンバイナ・マスク・レジスタ（EXPMASK*n*）

例外結合器は、イベント結合器と同様に、EXCEP をトリガしたイベントのゲート制御に使用されるマスク・レジスタがあります。

注： イベント 0 ～ 3 の例外マスクは予約されていて、常にマスクされています。

エクセプション・コンバイナ・マスク・レジスタ（EXPMASK*n*）を図 7-41 ～ 図 7-44 に示し、表 7-13 で説明します。

図 7-41. エクセプション・コンバイナ・マスク・レジスタ 0（EXPMASK0）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
XM31	XM30	XM29	XM28	XM27	XM26	XM25	XM24	XM23	XM22	XM21	XM20	XM19	XM18	XM17	XM16
R/W-FFFFh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XM15	XM14	XM13	XM12	XM11	XM10	XM9	XM8	XM7	XM6	XM5	XM4	XM3	XM2	XM1	XM0
R/W-FFFFh															

凡例：R/W = リード / ライト。-*n* = リセット後の値。

図 7-42. エクセプション・コンバイナ・マスク・レジスタ 1（EXPMASK1）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
XM63	XM62	XM61	XM60	XM59	XM58	XM57	XM56	XM55	XM54	XM53	XM52	XM51	XM50	XM49	XM48
R/W-FFFFh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XM47	XM46	XM45	XM44	XM43	XM42	XM41	XM40	XM39	XM38	XM37	XM36	XM35	XM34	XM33	XM32
R/W-FFFFh															

凡例：R/W = リード / ライト。-*n* = リセット後の値。

図 7-43. エクセプション・コンバイナ・マスク・レジスタ 2（EXPMASK2）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
XM95	XM94	XM93	XM92	XM91	XM90	XM89	XM88	XM87	XM86	XM85	XM84	XM83	XM82	XM81	XM80
R/W-FFFFh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XM79	XM78	XM77	XM76	XM75	XM74	XM73	XM72	XM71	XM70	XM69	XM68	XM67	XM66	XM65	XM64
R/W-FFFFh															

凡例：R/W = リード / ライト。-*n* = リセット後の値。

図 7-44. エクセプション・コンバイナ・マスク・レジスタ 3（EXPMASK3）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
XM127	XM126	XM125	XM124	XM123	XM122	XM121	XM120	XM119	XM118	XM117	XM116	XM115	XM114	XM113	XM112
R/W-FFFFh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XM111	XM110	XM109	XM108	XM107	XM106	XM105	XM104	XM103	XM102	XM101	XM100	XM99	XM98	XM97	XM96
R/W-FFFFh															

凡例：R/W = リード / ライト。*-n* = リセット後の値。

表 7-13. エクセプション・コンバイナ・マスク・レジスタ（EXPMASK*n*）フィールドの説明

ビット	フィールド	値	説明
31-0	XMyyy		例外結合器でのイベント EVTyyy の使用をイネーブルします。
		0	EVTyyy を組み合わせます。
		1	EVTyyy の組み合わせをディスエーブルします。

7.5.4.2 マスクド・エクセプション・フラグ・レジスタ（MEXPFLAG*n*）

例外結合器には、4 個のマスクド・エクセプション・フラグ・レジスタがあります（マスクされたエクセプション・フラグ・レジスタ）。マスクド・エクセプション・フラグ・レジスタ（MEXPFLAG*n*）を図 7-45 ～ 図 7-48 に示し、表 7-14 で説明します。

図 7-45. マスクド・エクセプション・フラグ・レジスタ 0（MEXPFLAG0）

31	30	29	28	27	26	25	24
MXF31	MXF30	MXF29	MXF28	MXF27	MXF26	MXF25	MXF24
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
23	22	21	20	19	18	17	16
MXF23	MXF22	MXF21	MXF20	MXF19	MXF18	MXF17	MXF16
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
15	14	13	12	11	10	9	8
MXF15	MXF14	MXF13	MXF12	MXF11	MXF10	MXF9	MXF8
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
7	6	5	4	3	2	1	0
MXF7	MXF6	MXF5	MXF4	MXF3	MXF2	MXF1	MXF0
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

凡例：R = リード専用。-*n* = リセット後の値。

図 7-46. マスクド・エクセプション・フラグ・レジスタ 1（MEXPFLAG1）

31	30	29	28	27	26	25	24
MXF63	MXF62	MXF61	MXF60	MXF59	MXF58	MXF57	MXF56
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
23	22	21	20	19	18	17	16
MXF55	MXF54	MXF53	MXF52	MXF51	MXF50	MXF49	MXF48
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
15	14	13	12	11	10	9	8
MXF47	MXF46	MXF45	MXF44	MXF43	MXF42	MXF41	MXF40
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
7	6	5	4	3	2	1	0
MXF39	MXF38	MXF37	MXF36	MXF35	MXF34	MXF33	MXF32
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

凡例：R = リード専用。-*n* = リセット後の値。

図 7-47. マスクド・エクセプション・フラグ・レジスタ 2（MEXPFLAG2）

31	30	29	28	27	26	25	24
MXF95	MXF94	MXF93	MXF92	MXF91	MXF90	MXF89	MXF88
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
23	22	21	20	19	18	17	16
MXF87	MXF86	MXF85	MXF84	MXF83	MXF82	MXF81	MXF80
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
15	14	13	12	11	10	9	8
MXF79	MXF78	MXF77	MXF76	MXF75	MXF74	MXF73	MXF72
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
7	6	5	4	3	2	1	0
MXF71	MXF70	MXF69	MXF68	MXF67	MXF66	MXF65	MXF64
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

凡例：R = リード専用。*-n* = リセット後の値。

図 7-48. マスクド・エクセプション・フラグ・レジスタ 3（MEXPFLAG3）

31	30	29	28	27	26	25	24
MXF127	MXF126	MXF125	MXF124	MXF123	MXF122	MXF121	MXF120
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
23	22	21	20	19	18	17	16
MXF119	MXF118	MXF117	MXF116	MXF115	MXF114	MXF113	MXF112
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
15	14	13	12	11	10	9	8
MXF111	MXF110	MXF109	MXF108	MXF107	MXF106	MXF105	MXF104
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
7	6	5	4	3	2	1	0
MXF103	MXF102	MXF101	MXF100	MXF99	MXF98	MXF97	MXF96
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

凡例：R = リード専用。*-n* = リセット後の値。

表 7-14. マスクド・エクセプション・フラグ・レジスタ（MEXPFLAG*n*）フィールドの説明

ビット	フィールド	値	説明
31-0	MXFyyy	0 ～ FFFF FFFFh	エクセプション・マスク・レジスタ（EXPMASK*n*）で XMyyy = 0 の場合、EFyyy の内容を示します。 (XMyyy = 0) の場合 MXFyyy = Efyyy それ以外の場合 MXFyyy = 0

7.5.5 権限およびインタラプト・コントローラ・レジスタ

C64x+ アーキテクチャは、メモリ保護サポート機能を提供します。

表 7-15 に、役割ごとにアクセスできるインタラプト・コントローラ・レジスタの概要を示します。

表 7-15. インタラプト・コントローラ・レジスタのアクセス権限

レジスタ	スーパーバイザ	ユーザ
EVTFLAGx	R	R
EVTCLRx	W	/
EVTSETx	W	/
EVTMASKx	R/W	R
MEVTFLAGx	R	R
EXPMASKx	R/W	R
MEXPFLAGx	R	R
INTMUXx	R/W	R
INTSTAT	R	R
INTXCLR	W	/
INTDMASK	R/W	R

第 8 章

JAJU058

メモリ保護

項目 **ページ**

8.1 はじめに

8.1.1 メモリ保護の目的

メモリ保護を行うと、システムに多くの利益がもたらされます。メモリ保護機能は次のとおりです。

- オペレーティング・システムのデータ構造を正常に動作しないコードから保護します。
- 不正なメモリ・アクセスに関する豊富な情報を提供することで、デバッグを支援します。
- オペレーティング・システムがスーパーバイザ・モードとユーザ・モードのアクセス間の境界を明確に指定できるため、システムの堅牢性が大幅に高まります。

C64x+ メガモジュールのメモリ保護アーキテクチャは、CPU 特権レベルとメモリ・システムのアクセス権限構造を組み合わせることでこれらの利点をもたらしています。

8.1.2 特権レベル

スレッドの特権は、スレッド自身が備えているアクセス権限レベルを決定します。

CPU 上で動作しているコードは、1 つまたは 2 つの特権モード、つまりスーパーバイザ・モードまたはユーザ・モードで実行されています。スーパーバイザ・コードは、ユーザ・コードより信頼性が高いものと見なされています。スーパーバイザ・スレッドの例には、オペレーティング・システムのカーネルおよびハードウェアのデバイス・ドライバが挙げられます。ユーザ・スレッドの例には、ボコーダおよび最終アプリケーションがあります。

スーパーバイザ・モードは、原則的に、ペリフェラル・レジスタおよびメモリ保護コンフィギュレーションへのアクセスが許可されています。ユーザ・モードは、原則的に、OS が明確に指定したメモリ空間のみ使用が許可されています。

CPU アクセスは、内部 DMA および他のアクセスと同様に、それらに対応した特権レベルを備えています。CPU の特権レベルは、前述のように決定されます。CPU によって開始される内部 DMA アクセスは、開始された時点での CPU 特権レベルを継承します。

8.2 用語と定義

本章で使用する用語の詳細な定義については、本書の付録 A を参照してください。付録 A では、本書全体で使用している一般的な用語について説明しています。

8.3 メモリ保護アーキテクチャ

8.3.1 メモリ保護ページ

C64x+ メモリ保護アーキテクチャは、DSP 内部メモリ（L1P、L1D、L2）を複数の*ページ*に分割します。各ページには、それに対応したアクセス権限のセットがあります。8.3.2 項とそれ以下のレベルで、そのアクセス権限について説明します。

通常、メモリ・ページのサイズは 2 の累乗で表します。L1 および L2 のメモリ・ページのサイズは、デバイスごとに異なります。詳細については、各デバイスのデータシートを参照してください。

8.3.2 アクセス権限の構造

メモリ保護アーキテクチャは、16 ビットのアクセス権限エントリにある 2 つのアクセス権限フィールドで、ページごとのアクセス権限の構造を指定します。図 8-1 に、アクセス権限エントリの構造を示します。

図 8-1. アクセス権限フィールド

31															16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Allowed IDs								Reserved		Access Types					
AID5	AID4	AID3	AID2	AID1	AID0	AIDX	LOCAL			SR	SW	SX	UR	UW	UX

8.3.2.1 リクエスタ ID ベースのアクセス制御

デバイスのリクエスタにはそれぞれ、関連付けられた、権限を確認するための N ビット・コードがあります。この ID は、該当リクエスタに代わって行われるすべてのメモリ・アクセスと IDMA に付いています。つまり、リクエスタが IDMA レジスタに直接ライトすることで IDMA 転送をトリガする場合、IDMA エンジンは転送と同時に該当 ID を発行します。それぞれの CPU およびすべてのマスタ動作するペリフェラル（RapidIO、HPI、EMAC）には、ID があります。同じデバイス内にある複数のシステム・マスタは ID を共有できます。それぞれのメモリ保護エントリには、関連付けられた、特定のページにアクセスできるリクエスタを示す、許可 ID フィールドがあります。メモリ保護ハードウェアは、すべての使用可能なリクエスタの ID を、メモリ保護エントリ内の許可 ID フィールドのビットにマップします。許可 ID フィールドは、さまざまな CPU、CPU 以外のリクエスタ、およびローカル・メモリへの特定の CPU アクセスを区別します。

- AID0 ～ AID5 は、番号が小さい ID を許可 ID ビットにマップします。
- もう 1 つの許可 ID ビットである AIDX は、それ以上の番号の PrivID によって行われるアクセスをキャプチャします。
- LOCAL ビットは、ローカルな L1 および L2 への CPU アクセスを特別に扱います。

許可 ID ビット・フィールドを図 8-2 に示し、表 8-1 で説明します。

図 8-2. 許可 ID ビット・フィールド

15	14	13	12	11	10	9	8
AID5	AID4	AID3	AID2	AID1	AID0	AIDX	LOCAL

許可 ID ビット・フィールドは、8 ビットです。

AID ビットを 1 にセットすると、対応する ID へのアクセスが許可されます。AID ビットを 0 にクリアすると、対応するリクエスタへのアクセスが拒否されます。

表 8-1. 許可 ID ビット・フィールドの説明

ビット	フィールド	説明
15	AID5	ID = 5 からのアクセスを許可します。
14	AID4	ID = 4 からのアクセスを許可します。
13	AID3	ID = 3 からのアクセスを許可します。
12	AID2	ID = 2 からのアクセスを許可します。
11	AID1	ID = 1 からのアクセスを許可します。
10	AID0	ID = 0 からのアクセスを許可します。
9	AIDX	ID >= 6 からのアクセスを許可します。
8	LOCAL	CPU からローカル・メモリ（L1/L2 のみ）へのアクセスを許可します。

ビット AID0 ～ AID5 に対応する前述の ID 割り当ては、CPU のローカル L1 および L2 メモリ以外のすべての IDMA および CPU メモリ・アクセスに適用されます。LOCAL ビットは、ローカル L1 および L2 メモリへの CPU アクセスを管理します。AIDX ビットは、それに対応した専用の AID ビットを備えていない ID にマップされます。

8.3.2.2 リクエスト・タイプ・ベースのアクセス権限

メモリ保護モデルは、リード、ライト、実行の 3 つの基本機能アクセス・タイプを指定します。リードおよびライトは、データ・アクセス、つまり CPU のロード / ストア・ユニットからまたは IDMA エンジン始まるアクセスを意味します。実行は、プログラム・フェッチに関連付けられたアクセスを意味します。

メモリ保護モデルを使用すると、ユーザ・モードとスーパーバイザ・モードの双方各々に、リード、ライト、および実行のアクセス権限を制御できます。これらの 6 つのアクセス制限ビットを表 8-2 に示します。

表 8-2. リクエスト・タイプ・アクセス制御

ビット	フィールド	説明
5	SR	スーパーバイザがリードできます。
4	SW	スーパーバイザがライトできます。
3	SX	スーパーバイザが実行できます。
2	UR	ユーザがリードできます。
1	UW	ユーザがライトできます。
0	UX	ユーザが実行できます。

各ビットでは、1 はアクセス・タイプを許可し、0 は拒否します。そのため、UX = 1 はユーザ・モードで特定のページから実行できるということを意味します。メモリ保護アーキテクチャを使用すると、ここに示した 6 個のビットをすべて別々に使用できます。64 個のエンコーディングが完全に許可されていますが、プログラムですべてのエンコーディングを使用できるとは限りません。

8.3.3 不正なアクセスと例外

不正なアクセスが検出されると、メモリ保護ハードウェアは 2 つの独立した処置を行います。

- アクセスが行われないようにします。
- エラーをオペレーティング・システムへレポートします。

不正なアクセスは、関係するページまたはレジスタに指定されているアクセス制限を超える制限を要求するメモリ・アクセスです。これ以降では、不正なアクセスが行われたときのメモリ保護動作について説明します。

8.3.3.1 不正なアクセスの処理

不正なアクセスが行われると、メモリ保護によってリクエスタがアクセスできないようにし、不正なアクセスによって保護されるメモリの状態が変更されないことを確実にします。

8.3.3.2 例外の生成

不正なアクセスを検出すると、メモリ保護ハードウェアはエラーをオペレーティング・システムにレポートします。

8.4 メモリ・プロテクション・アーキテクチャ・レジスタ

メモリ保護アーキテクチャは、メモリ・マップド・レジスタ（MMR）のレジスタ・セットをいくつか規定しています。MPA を実装した各ハードウェア・ブロックは、レジスタ・セットの一部としてこれらの MMR を実装します。結果的に、これらの MMR は、そのコンフィギュレーション・レジスタ・アドレス空間にあります。

MMR を実装したペリフェラルは、これらの MMR へのアクセスを管理します。

MMR は 3 つの主要なカテゴリに分類されます。

- メモリ・プロテクション・ページ・アトリビュート・レジスタ（MPPA）：これらのレジスタは、保護されたページごとに対応するアクセス権限を格納します。MPPA レジスタの定義は、8.4.1 項に示します。
- メモリ・プロテクション・フォールト・レジスタ（MPFxR）：メモリ保護障害を生成する各ペリフェラルには、障害の詳細を記録するために MPFAR、MPFSR、MPFCR の各レジスタが用意されています。これらのレジスタの定義は、8.4.2 項および 8.4.2.1 項に示します。
- メモリ・プロテクション・ロック・レジスタ（MPLK）：ペリフェラルで他のリクエストを処理できない場合、ロックをかけてそのペリフェラルのメモリ保護エントリに対するすべての更新はディスエーブルされます。MPLK レジスタの定義は、8.4.3 項に示します。それぞれのメモリで独自のメモリ・プロテクション・レジスタを実装しているため、メモリ・マップの詳細については、各デバイスのデータシートを参照してください。

表 8-3 に、メモリ保護アーキテクチャのメモリ・マップド・レジスタを示します。これらのレジスタのメモリ・アドレスについては、各デバイスのデータ・マニュアルを参照してください。

表 8-3. メモリ・プロテクション・アーキテクチャ・レジスタ

略称	レジスタの説明	参照先
MPPA	メモリ・プロテクション・ページ・アトリビュート・レジスタ	8.4.1 項
MPFAR	メモリ・プロテクション・フォールト・アドレス・レジスタ	8.4.2 項
MPFSR	メモリ・プロテクション・フォールト・ステータス・レジスタ	8.4.2 項
MPFCR	メモリ・プロテクション・フォールト・コマンド・レジスタ	8.4.2 項
MPLK	メモリ・プロテクション・ロック・レジスタ	8.4.3 項

8.4.1 メモリ・プロテクション・ページ・アトリビュート・レジスタ（MPPA）

設定可能なメモリ保護ページという概念を実装した各メモリには、メモリ・プロテクション・ページ・アトリビュート・レジスタ（MPPA）があります。1 つの MPPA レジスタで、ペリフェラルが実装したそれぞれのページをカバーします。通常、これらのレジスタは、メモリの MMR メモリ・マップ内の連続したブロックにあります。

それぞれの MPPA レジスタは、メモリ・マップでは 32 ビットを占有しますが、その 16 ビットのみが使用されます。8.3.2 項に、MPPA レジスタ・フィールドのレイアウトとその定義を示します。

MPPA レジスタのリセット値は、デバイスによって異なります。

8.4.2 メモリ・プロテクション・フォールト・レジスタ（MPFAR、MPFSR、MPFCR）

メモリ保護アーキテクチャを実装し、かつ例外を生成するすべてのメモリには、メモリ保護違反の詳細をレポートするために、メモリ・プロテクション・フォールト・レジスタのセットがあります。

C64x+ MPA が指定する 3 つのレジスタは、メモリ・プロテクション・フォールト・アドレス・レジスタ（MPFAR）、メモリ・プロテクション・フォールト・ステータス・レジスタ（MPFSR）、メモリ・プロテクション・フォールト・コマンド・レジスタ（MPFCR）です。

メモリ保護アーキテクチャを実装していても、例外を生成できないメモリには、これらのレジスタは実装されていません。

8.4.2.1 メモリ・アクセス・プロテクション・フォールト・レジスタ

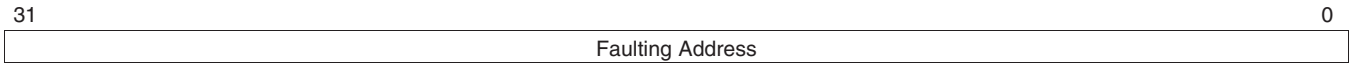
メモリ保護ハードウェアの特定の部分が特権違反を検出した場合、例外のトリガ処理の一部として違反に関する基本的な情報がキャプチャされます。原則的に、障害を引き起こしたアドレスおよびアクセス・タイプがキャプチャされます。

ハードウェアは、メモリのメモリ・プロテクション・フォールト・アドレス・レジスタ（MPFAR）に障害を引き起こしたアドレスを記録します。ハードウェアは、メモリのメモリ・プロテクション・フォールト・ステータス・レジスタ（MPFSR）に障害に関するその他の情報を記録します。ソフトウェアで、メモリ・プロテクション・フォールト・コマンド・レジスタ（MPFCR）にライトして、障害をクリアすることができます。

8.4.2.1.1 メモリ・プロテクション・フォールト・アドレス・レジスタ（MPFAR）

メモリ・プロテクション・フォールト・アドレス・レジスタ（MPFAR）を図 8-3 に示し、表 8-4 で説明します。

図 8-3. メモリ・プロテクション・フォールト・アドレス・レジスタ（MPFAR）



凡例：R = リード専用。*-n* = リセット後の値。

表 8-4. メモリ・プロテクション・フォールト・アドレス・レジスタ（MPFAR）フィールドの説明

ビット	フィールド	値	説明
31-0	Faulting Address	0 ～ FFFF FFFFh	障害が発生したアドレス。

8.4.2.1.2 メモリ・プロテクション・フォールト・ステータス・レジスタ（MPFSR）

メモリ・プロテクション・フォールト・ステータス・レジスタ（MPFSR）を図 8-4 に示し、表 8-5 で説明します。

図 8-4. メモリ・プロテクション・フォールト・ステータス・レジスタ（MPFSR）

31									16
Reserved									
R-0									

15	9	8	7	6	5	4	3	2	1	0
FID		LOCAL	Reserved		SR	SW	SX	UR	UW	UX
R-0		R-0	R-0		R-0	R-0	R-0	R-0	R-0	R-0

凡例：R = リード専用。*-n* = リセット後の値。

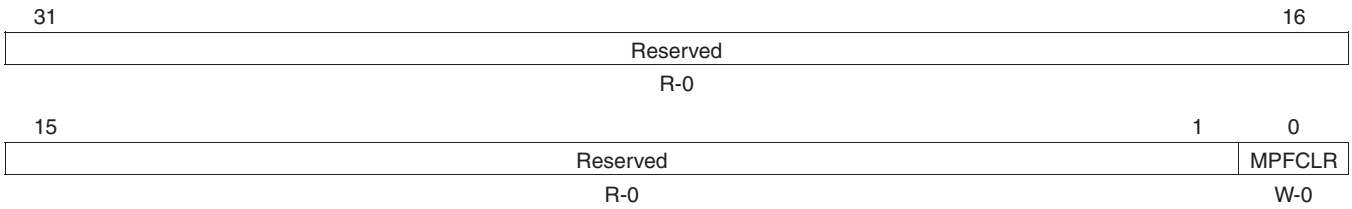
表 8-5. メモリ・プロテクション・フォールト・ステータス・レジスタ（MPFSR）フィールドの説明

ビット	フィールド	値	説明
31-16	Reserved	0	予約。
15-9	FID	1Fh	ビット 6:0 は障害を引き起こしたリクエスタ の ID を示します。ID の幅が 7 ビットより狭い場合、残りのビットは 0 を返します。ID の幅が 7 ビットより広い場合、他のビットは切り捨てられます。LOCAL = 1 の場合、FID = 0。
8	LOCAL	0-1	アクセスは「ローカル」。
7-6	Reserved	0	予約。
5	SR	0-1	セットされている場合、スーパーバイザによるリード・リクエストを示します。
4	SW	0-1	セットされている場合、スーパーバイザによるライト・リクエストを示します。
3	SX	0-1	セットされている場合、スーパーバイザによるプログラム・フェッチ・リクエストを示します。
2	UR	0-1	セットされている場合、ユーザによるリード・リクエストを示します。
1	UW	0-1	セットされている場合、ユーザによるライト・リクエストを示します。
0	UX	0-1	セットされている場合、ユーザによるプログラム・フェッチ・リクエストを示します。

8.4.2.1.3 メモリ・プロテクション・フォールト・コマンド・レジスタ（MPFCR）

メモリ・プロテクション・フォールト・コマンド・レジスタ（MPFCR）を図 8-5 に示し、表 8-6 で説明します。

図 8-5. メモリ・プロテクション・フォールト・コマンド・レジスタ（MPFCR）



凡例：R = リード専用。W = ライト専用。*-n* = リセット後の値。

表 8-6. メモリ・プロテクション・フォールト・コマンド・レジスタ（MPFCR）フィールドの説明

ビット	フィールド	値	説明
31-1	Reserved	0	予約。
0	MPFCLR		L1DMPFAR レジスタをクリアするためのコマンド。
		0	影響なし。
		1	L1DMPFAR および L1DMPFCR レジスタをクリアします。

メモリ・プロテクション・ページ・アトリビュート・レジスタとよく似たフォーマットで、MPFAR レジスタは保護違反を起こしたアドレスを記録し、MPFSR レジスタはアクセス・タイプを記録します。MPFCLR レジスタには、MPFAR および MPFCLR レジスタをクリアするために 1 つのコマンド・ビットが含まれています。

キャッシュは、通常の機能アクセスと区別される 2 つの特殊なアクセス・タイプ（ライン・フィルとライトバック）を生成します。保護ハードウェアは、特殊なパターンをアクセス・タイプ・フィールドにエンコードすることで、キャッシュ・トランザクション時の障害を示します。

- 障害を起こしたライン・フィルは、SR = SW = = UR = UW = UX = 1 をセットします。
- 障害を起こしたビクティム・ライトバックは、SW = UW = 1 をセットします。

ソフトウェアでこの方法を使用して、メモリ保護障害をデコードするには次のようにします。

- LOCAL フィールドがセットされている場合、リクエストは自身のメモリに対するローカルな CPU リクエストです。それ以外の場合、障害を起こしたリクエスタの ID はフォールト・ステータス・レジスタのビット 9 ～ 15 にあります。
- アクセス・タイプ・フィールド（SR、SW、SX、UR、UW、UX）の値は、デフォルトのアクセス・タイプを示します（表 8-7 を参照）。

表 8-7. MPFSR レジスタのアクセス・タイプ・フィールドの意味

SR	SW	SX	UR	UW	UX	意味
1	0	0	0	0	0	スーパーバイザのリードによる障害。
0	**1**	0	0	0	0	スーパーバイザのライトによる障害。
0	0	**1**	0	0	0	スーパーバイザのプログラム・フェッチによる障害。
0	0	0	**1**	0	0	ユーザのリードによる障害。
0	0	0	0	**1**	0	ユーザのライトによる障害。
0	0	0	0	0	**1**	ユーザのプログラム・フェッチによる障害。
1	**1**	**1**	**1**	**1**	**1**	キャッシュ・ライン・フィルによる障害。
0	**1**	0	0	**1**	0	キャッシュ・ビクティム・ライトバックによる障害。
その他						予約 -- これはエンドポイントで指定される場合があります。

別のマスタがメガモジュールの内容をキャッシュする場合、メガモジュールはライン・フィルおよびビクティム・ライトバック・エンコーディングを生成し、かつそのときに障害を検出します。

それぞれのメモリ保護ブロックは、自身のメモリ保護障害情報をキャプチャします。そのため、それぞれのメモリ保護例外を起こす可能性のあるソースには、対応する MPFAR/MPFSR/MPFCR レジスタ・セットがあります。

MPFAR および MPFSR レジスタは、1 つの障害に関する情報をのみ格納します。障害を起こした結果、例外が発生します。ソフトウェアで MPFCR にライトすることでクリアするまで、障害情報は保持されます。

MPFCR レジスタの MPFCLR（ビット 0）に 1 をライトすることで、スーパーバイザは記録された障害をクリアします。このビットに 1 をライトすると、MPFAR と MPFSR レジスタを両方ともクリアします。MPFAR および MPFSR レジスタは、ライトには対応しません。スーパーバイザが障害をクリアすると、ハードウェアでは次の保護違反を記録し、例外発生時にそのことを示す信号を送ります。MPFCR レジスタのそれ以外のビットに 1 をライトしても、メモリ・プロテクション・レジスタには影響を与えません。MPFCR レジスタの MPFCLR フィールドに 0 をライトしても影響はありません。

さまざまな独立したメモリ保護ブロックは、お互いに直接連携しているわけではありません。一部の動作（キャッシュ・ライン・フィルなど）によって、エンドポイントおよびキャッシュ階層で例外が発生することもあります。したがって、CPU が最初の例外にも応答する前に、1 回の不正なメモリ・アクセスによって、別々のブロックで複数の例外が発生する場合があります。それでも、CPU がメモリの MPFAR および MPFSR レジスタをクリアするまで、個々のメモリによって、2 つ以上の例外が発生することはありません。

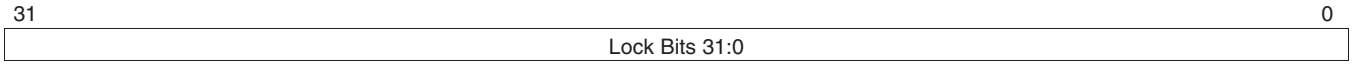
8.4.3 メモリ・プロテクション・ロック・レジスタ（MPLK*n*）

セキュリティの別のレイヤーとして、メモリ保護アーキテクチャは、ハードウェアによる「保護ロック」を規定しています。ハードウェアによるロックは、特定のメモリのプロテクション・レジスタに対する、それ以外のすべてのアクセス制御にわたる別のレイヤーになります。

ハードウェアによるロック機能を保護エントリに実装したデバイスには、6 個のレジスタが実装されています（図 8-6 ～ 図 8-11）。

メモリ・プロテクション・ロック・レジスタ（MPLK*n*）を図 8-6 ～ 図 8-10 に示し、表 8-8 で説明します。

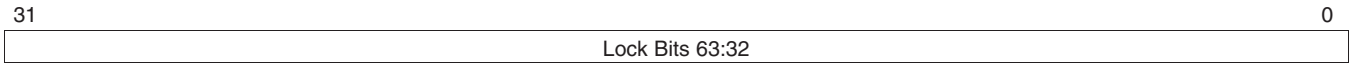
図 8-6. メモリ・プロテクション・ロック・レジスタ（MPLK0）



W-x

凡例：W = ライト専用。-*x* = 値は不定。詳細については、各デバイスのデータ・マニュアルを参照してください。

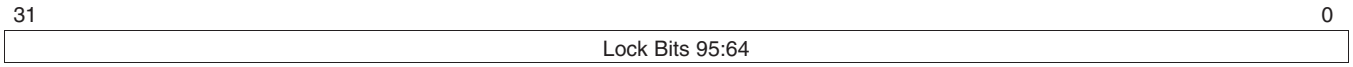
図 8-7. メモリ・プロテクション・ロック・レジスタ（MPLK1）



W-x

凡例：W = ライト専用。-*x* = 値は不定。詳細については、各デバイスのデータ・マニュアルを参照してください。

図 8-8. メモリ・プロテクション・ロック・レジスタ（MPLK2）



W-x

凡例：W = ライト専用。-*x* = 値は不定。詳細については、各デバイスのデータ・マニュアルを参照してください。

図 8-9. メモリ・プロテクション・ロック・レジスタ（MPLK3）

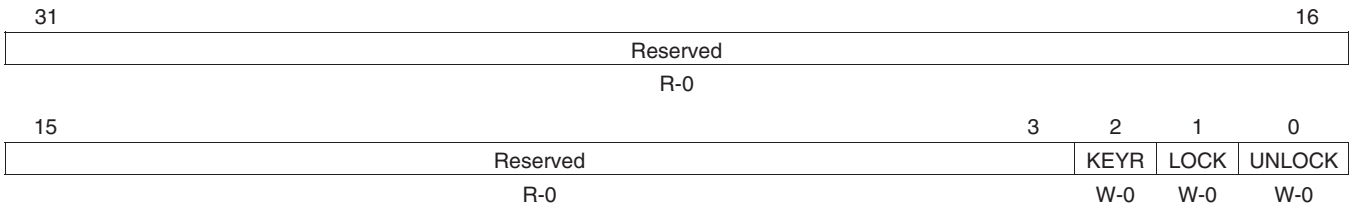
W-x

凡例：W = ライト専用。-*x* = 値は不定。詳細については、各デバイスのデータ・マニュアルを参照してください。

8.4.3.1 メモリ・プロテクション・ロック・コマンド・レジスタ（MPLKCMD）

メモリ・プロテクション・ロック・コマンド・レジスタ（MPLKCMD）を図 8-10 に示し、表 8-8 で説明します。

図 8-10. メモリ・プロテクション・ロック・コマンド・レジスタ（MPLKCMD）



凡例：R = リード専用。W = ライト専用。*-n* = リセット後の値。

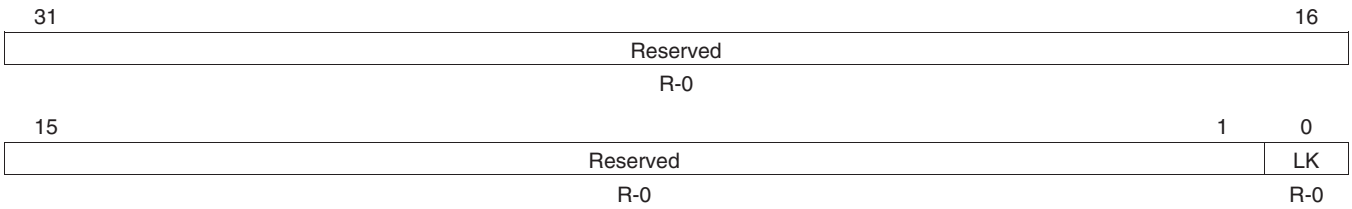
表 8-8. メモリ・プロテクション・ロック・コマンド・レジスタ（MPLKCMD）フィールドの説明

ビット	フィールド	値	説明
31-3	Reserved	0	予約。
2	KEYR		ステータスをリセットします。
		0	影響なし。
		1	ステータスをリセットします。
1	LOCK		ロック・シーケンスを完了するためのインターフェイス。
		0	影響なし。
		1	ソフトウェアがシーケンスを正しく実行した場合に、ロックします。
0	UNLOCK		ロック解除シーケンスを完了するためのインターフェイス。
		0	影響なし。
		1	ソフトウェアがシーケンスを正しく実行した場合に、ロックを解除します。

8.4.3.2 メモリ・プロテクション・ロック・ステータス・レジスタ（MPLKSTAT）

メモリ・プロテクション・ロック・ステータス・レジスタ（MPLKSTAT）を図 8-11 に示し、表 8-9 で説明します。

図 8-11. メモリ・プロテクション・ロック・ステータス・レジスタ（MPLKSTAT）



凡例：R = リード専用。*-n* = リセット後の値。

表 8-9. メモリ・プロテクション・ロック・ステータス・レジスタ（MPLKSTAT）フィールドの説明

ビット	フィールド	値	説明
31-1	Reserved	0	予約。
0	LK		ロックの現在のステータスを示します。
		0	ロックは解除されています。
		1	ロックされています。

ロックには、ロックまたはロック解除の 2 つのステートのいずれかが設定されています。リセットすると、MPLKSTAT レジスタの LK フィールドを使用してロックのステートはロック解除になります。

ロックが現在解除されている間、ソフトウェアでロックすることができます。ロックするには、アプリケーションで次の手順を正確に実行します。

1. MPLKCMD レジスタの KEYR フィールドに 1 をライトします。これにより、MPLK0 ～ MPLK3 レジスタ内部の一部のステータスがリセットされます。
2. MPLK0 ～ MPLK3 レジスタにキーをライトします。4 つのレジスタすべてに、一度だけライトする必要があります。ライトする順番は、問いません。
3. MPLKCMD レジスタの LOCK フィールドに 1 をライトします。これで、ロックされます。

この順番に従ってプログラムを記述すると、メモリ保護ハードウェアによってロックされます。このハードウェアでロックする手順は、次のとおりです。

- MPLKSTAT レジスタの LK フィールドを 1 にセットします。
- ライトされたキー（またはそのサブセット）を「ロック解除」キーとして設定します。
- このメモリに対して、MPPA および MPCFG レジスタへの以後のライトをブロックします。

ハードウェアは不正なロック・シーケンスを検出した場合、例外が発生したことを示す信号を送ります。ハードウェアは､障害が発生した時点でライトされた MPLK レジスタのアドレスを MPFAR レジスタの例外発生アドレスとしてレポートします。

レジスタが現在ロックされている場合、ソフトウェアでペリフェラルのプロテクション・レジスタのロックを解除するロック設定シーケンスによく似たシーケンスを実行します。

1. MPLKCMD レジスタの KEYR フィールドに 1 をライトします。これにより、MPLK0 ～ MPLK3 レジスタ内部の一部のステータスがリセットされます。
2. MPLK0 ～ MPLK3 レジスタにロック解除キーをライトします。ハードウェアは、ライトされた値と保存されていたキー値を比較します。ソフトウェアで、4 つのレジスタすべてに、一度だけライトする必要があります。ライトする順番は、問いません。
3. MPLKCMD レジスタの UNLOCK フィールドに 1 をライトします。ステップ 2 でライトされたキーが保存されていたキーと一致した場合、ハードウェアはロックを解除します。ステップ 2 でライトされたキーが一致しなかった場合、ハードウェアは例外が発生したことを示す信号を送ります。ハードウェアは、MPLKCMD レジスタのアドレスとして障害の発生したアドレスをレポートします。

8.4.4 128 ビットより短いキー

デバイスによっては、メモリに 128 ビットより短いキーが実装されている場合があります。この場合、ハードウェアが 128 ビットをすべて考慮していない場合でも、ロックを操作するアプリケーションではロック時およびロック解除時に 128 ビット・キーをすべてライトしてしまいます。

8.5 メモリ・プロテクション・レジスタへのアクセス時に行われるアクセス権限チェック

メモリ保護アーキテクチャを実装したメモリには、メモリ・プロテクション・レジスタでアクセス権限をチェックする機能が実装されています。表 8-10 に、これらのチェックの概要を示します。

表 8-10. MP レジスタへの許可アクセス

レジスタ・セット	スーパーバイザ		ユーザ	
	リード	ライト	リード	ライト
MPPAx	常時	ロック解除	常時	しない
MPFAR、MPFSR	常時	しない	常時	しない
MPFCR	しない	常時	しない	常時
MPLK0 ～ MPLK3	しない	ロック / ロック解除中	しない	しない
MPLKSTAT	常時	しない	常時	しない
MPLKCMD	しない	ロック / ロック解除の開始時 / 終了時	しない	しない

第 9 章

JAJU058

パワーダウン・コントローラ

項目 **ページ**

9.1 はじめに

ここでは、パワーダウン・コントローラの目的を説明し、その機能について解説します。

9.1.1 C64x+ メガモジュールのパワーダウン・マネージメント

C64x+ メガモジュールは、C64x+ メガモジュールのさまざまな部分をパワーダウンできる機能をサポートしています。C64x+ メガモジュールのパワーダウン・コントローラを使用すると、C64x+ メガモジュール全体をパワーダウンできます。これらの機能を使用すると、低価格のシステム電源要件に対応したシステムを設計できます。

注： C64x+ メガモジュール外部にあるペリフェラルが、パワーダウン機能を備えることもできます。ただし、その話題は本書の対象外になるため、本章では説明しません。

9.1.2 パワーダウン機能の概要

表 9-1 に、C64x+ メガモジュールで使用可能なパワーダウン機能を示します。また、その適用方法とタイミングについて簡単に説明します。

表 9-1. C64x+ メガモジュールのパワーダウン機能

パワーダウン機能	適用方法 / タイミング
L1P メモリ	SPLOOP 命令実行時
L2 メモリ	L2 コントロール・レジスタを使用してソフトウェアでプログラム可能
キャッシュ・コントロール・ハードウェア	キャッシュがディスエーブルの場合
CPU	IDLE 命令発行時
C64x+ メガモジュール全体	PDC および IDLE でイネーブルされる

9.2 パワーダウン機能

9.2.1 L1P メモリ

L1P メモリは、SPLOOP バッファからの命令実行時に動的にパワーダウンします。この機能は自動的にイネーブルされ、ユーザには見えません。SPLOOP 命令が完了すると、CPU は L1P メモリからのフェッチを再開し、RAM はウェイクします。つまり、L1P はアクセスがない場合、パワーダウンします。SPLOOP 命令の詳細については、『TMS320C64x/C64x+ DSP CPU and Instruction Set Reference Guide』（資料番号 SPRU732）を参照してください。

注： この L1P は、C64x+ メガモジュール全体がパワーダウンする場合にもパワーダウンします（9.2.5 項を参照）。

9.2.2 L2 メモリ

L2 メモリは、ソフトウェア制御の下で実行時にパワーダウンできる機能をサポートしています。この機能を使用すると、L2 の不必要な部分を一時的にパワーダウンできます。

パワー・マネージメントから見ると、L2 メモリは、4 つの論理ページ（L2 のポートごとに 2 つずつ）に分類できます。L2 パワーダウン・スリープ・レジスタ（L2PDSLEEP）の適切なフィールドをプログラムすると、それぞれページを個別にパワーダウンすることができます。同様に、L2 パワーダウン・ウェイク・レジスタ（L2PDWAKEn）をプログラムすると、それぞれのページをパワーアップすることができます。また、L2 はページがアクセスされると、そのページをウェイクします。

パワーダウンおよびウェイクアップの手順の詳細については、4.5 節を参照してください。

注： この L2 は、C64x+ メガモジュール全体がパワーダウンする場合にもパワーダウンします（9.2.5 項を参照）。

9.2.3 キャッシュ・パワーダウン・モード

L1D、L1P、L2 のいずれかのキャッシュがディスエーブルの場合、いずれもパワーダウン・モード状態になります。

注： この 3 台のキャッシュ・コントローラは、C64x+ メガモジュール全体がパワーダウンする場合にパワーダウンします（9.2.5 項を参照）。

9.2.4 CPU のパワーダウン

技術的に本書の対象範囲外の話題ですが、CPU は IDLE 命令が発行されると、パワーダウンします。CPU は割り込みによってウェイクします。IDLE 命令の詳細については、『TMS320C64x/C64x+ DSP CPU and Instruction Set Reference Guide』（資料番号 SPRU732）を参照してください。

また IDLE 命令 は、C64x+ メガモジュール全体をパワーダウンする手順の一部としても使用されます（9.2.5 項を参照）。

9.2.5 C64x+ メガモジュールのパワーダウン機能

注： ここで説明したように、メガモジュールをパワーダウンすることは、多くの場合、*静的パワーダウン*といいます。この用語は、このパワーダウン・モードを説明するために使われます。これはこの機能が長時間にわたり使われていることが多いためです。*動的パワーダウン*という用語が本章で使われている場合は、パワーダウン・モードが限られた時間だけ使われていることを示します。

C64x+ メガモジュール全体をパワーダウンする手順は、次のとおりです。前に指定したオプションを除き、C64x+ メガモジュールの一部だけをパワーダウンすることはできません。C64x+ メガモジュールをパワーダウンするには、完全にソフトウェア制御の下で、パワーダウン・コントローラ・コマンド・レジスタ（PDCCMD）のメガモジュール・パワーダウン・ビット（MEGPD）をプログラムします。

C64x+ メガモジュールをパワーダウンさせるのに必要なソフトウェア・シーケンスは、次のとおりです。

1. PDCCMD レジスタの MEGPD フィールドを 1 にセットして、パワーダウンをイネーブルします。
2. C64x+ メガモジュールをウェイクアップする CPU 割り込みをイネーブルします。他の割り込みをすべてディスエーブルします。
3. IDLE 命令を実行します。

C64x+ メガモジュールは、上記ステップ 2 でイネーブルされた割り込みが発生するまでパワーダウン・モードになったままです。

C64x+ メガモジュールがパワーダウンしている間に、DMA アクセスが L1D、L1P、L2 のいずれかのメモリに対して行われた場合、パワーダウン・コントローラ（PDC）は 3 つのメモリ・コントローラをウェイクします。PDC は DMA アクセスが処理されると、再度メモリ・コントローラをパワーダウンします。

9.2.6 その他のパワーダウン

9.2.6.1 外部からリクエストされたパワーダウン

システムによっては、C64x+ メガモジュールが外部で駆動されたパワーダウン・リクエストに対応することが望ましい場合があります。これは、9.2.5 項で説明している手順を使用することで、CPU 制御下でのみ可能です。

通常、外部のパワーダウン・リクエストは、外部ハードウェア割り込み / 例外を使用することで実現します。割り込みサービス・ルーチンでは、C64x+ メガモジュールのパワーダウン手順に従って、外部リクエストを処理します。

9.2.6.2 C62x/C64x/C67x DSP のパワーダウン・モード

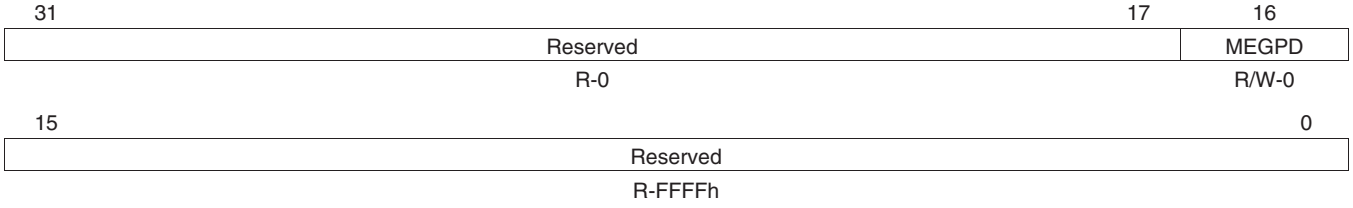
レガシーなデバイスにあるパワーダウン・モード（CPU のコントロール・ステータス・レジスタ（CSR）によってセットされる）は、C64x+ メガモジュール内ではサポートされません。これらの信号は C64x+ メガモジュール境界に送られるため、一部の C64x+ デバイスは CSR レジスタの PWRD ビットを使用する別のパワーダウン機能を利用している場合があります。このような機能の詳細については、各デバイスのデータ・マニュアルを参照してください。

9.3 パワーダウン・コントローラ・コマンド・レジスタ（PDCCMD）

パワーダウン・コントローラ・コマンド・レジスタ（PDCCMD）を使用して、（アドレス 0181 0000h にある）PDC をプログラムします。PDCCMD レジスタの MEGPD ビットを 1 にセットすると、C64x+ メガモジュールのグローバル・スタティック・パワーダウン・モードはイネーブルされます。MEGPD レジスタを 1 にセットすると、C64x+ メガモジュールのグローバル・スタティック・パワーダウン・モードは、CPU がアイドル状態に入ったときにアクティブになります。CPU がスーパーバイザ・モードの場合のみ、PDCCMD はライトできます。PDCCMD は、スーパーバイザ / ユーザのモードに関係なくリードできます。

パワーダウン・コントローラ・コマンド・レジスタ（PDCCMD）を図 9-1 に示し、表 9-2 で説明します。

図 9-1. パワーダウン・コントローラ・コマンド・レジスタ（PDCCMD）



凡例：R/W = リード / ライト。R = リード専用。-*n* = リセット後の値。

表 9-2. パワーダウン・コントローラ・コマンド・レジスタ（PDCCMD）フィールドの説明

ビット	フィールド	値	説明
31-17	Reserved	0	予約。
16	MEGPD		IDLE 時にパワーダウン。
		0	通常動作。CPU が IDLE の場合、CPU または C64x+ メガモジュールをパワーダウンしません。
		1	スリープ・モード。CPU が IDLE 状態に入ると、CPU と C64x+ メガモジュールをパワーダウンします。
15-0	Reserved	1	予約。これらのビットは、常に 1 としてリードされます。

表 9-3 に、パワーダウン・コントローラ・コマンド・レジスタのアクセス権限について示します。

表 9-3. PDC コマンド・レジスタのアクセス権限

レジスタ	スーパーバイザ	ユーザ
PDCCMD	R/W	R

第 10 章

JAJU058

その他

項目 **ページ**

10.1 はじめに

表 10-1 に、その他のメモリ・マップド・レジスタを示します。

表 10-1. その他のレジスタ

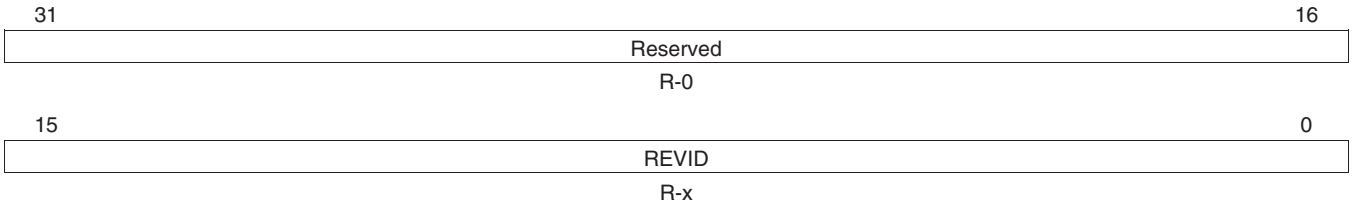
アドレス	略称	レジスタの説明	参照先
0181 2000h	MM_REVID	メガモジュール・リビジョン ID レジスタ	10.2 節
0182 0400h	BUSERR	バス・エラー・ステータス・レジスタ	10.3 節
0182 0404h	BUSERRCLR	バス・エラー・クリア・レジスタ	10.4 節

10.2 メガモジュール・リビジョン ID レジスタ（MM_REVID）

C64x+ メガモジュール・リビジョン ID レジスタ（MM_REVID）は、メガモジュールのリビジョンに関する情報を提供します。

メガモジュール・リビジョン ID レジスタ（MM_REVID）を図 10-1 に示し、表 10-2 で説明します。

図 10-1. メガモジュール・リビジョン ID レジスタ（MM_REVID）



凡例：R = リード専用。*-n* = リセット後の値。*-x* = 値は不定。詳細については、各デバイスのデータ・マニュアルを参照してください。

表 10-2. メガモジュール・リビジョン ID レジスタ（MM_REVID）フィールドの説明

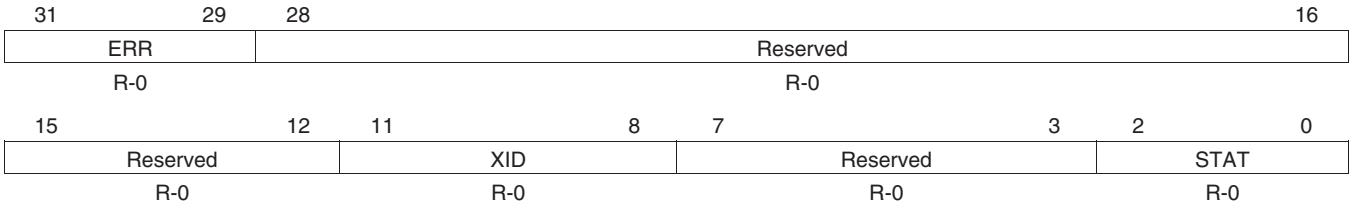
ビット	フィールド	値	説明
31-16	Reserved	0	予約されたビット位置。
15-0	REVID		C64x+ メガモジュールのバージョンを示すリビジョンは、デバイスに実装されています。C64x+ メガモジュールのリビジョンは、シリコンのリビジョンによって異なります。詳細については、各デバイスのデータ・マニュアルを参照してください。

10.3 バス・エラー・レジスタ（BUSERR）

バス・エラー・レジスタ（BUSERR）は、MDMA バスまたは CFG バスで行われる外部トランザクションに対してエラーを示す信号を送ります。

バス・エラー・レジスタ（BUSERR）を図 10-2 に示し、表 10-3 で説明します。

図 10-2. バス・エラー・レジスタ（BUSERR）



凡例：R = リード専用。*-n* = リセット後の値。

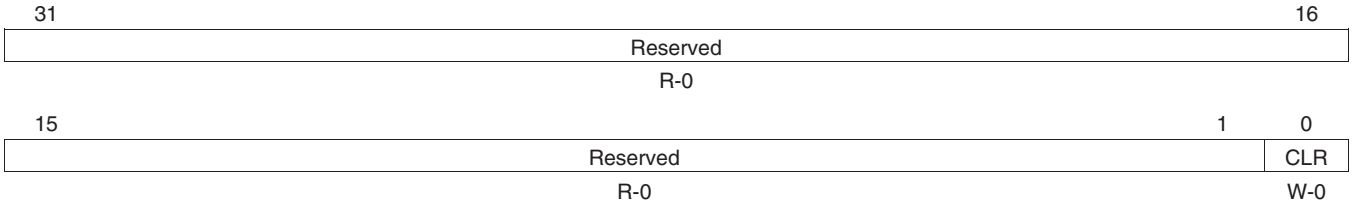
表 10-3. バス・エラー・レジスタ（BUSERR）フィールドの説明

ビット	フィールド	値	説明
31-29	ERR	0 〜 7h	エラーの検出。
		0	エラーなし。
		1h	MDMA リード時に検出されるエラー・ステータス。
		2h	MDMA ライト時に検出されるエラー・ステータス。
		3h	CFG リード時に検出されるエラー・ステータス。
		4h	CFG ライト時に検出されるエラー・ステータス。
		5h 〜 7h	予約。
28-12	Reserved	0	予約。
11-8	XID	0 〜 Fh	トランザクション ID。 リード・エラーまたはライト・エラーの検出時に、トランザクション ID（RID または WID）を格納します。
7-3	Reserved	0	予約。
2-0	STAT	0 〜 7h	トランザクション・ステータス。
		0	成功（エラーの原因とはならずラッチされる）、または認識されない RID/WID（エラーの原因となりラッチされる）。
		1h	アドレッシング・エラー。
		2h	特権エラー。
		3h	タイムアウト・エラー。
		4h	データ・エラー。
		5h 〜 6h	予約。
		7h	排他的 - 動作障害。

10.4 バス・エラー・ディテール・レジスタ（BUSERRCLR）

バス・エラー・ディテール・レジスタ（BUSERRCLR）の内容と説明を図 10-3 に示し、表 10-4 で説明します。

図 10-3. バス・エラー・ディテール・レジスタ（BUSERRCLR）



凡例：R = リード専用。W = ライト専用。*-n* = リセット後の値。

表 10-4. バス・エラー・ディテール・レジスタ（BUSERRCLR）フィールドの説明

ビット	フィールド	値	説明
31-1	Reserved	0	予約。
0	CLR		レジスタをクリアします。
		0	ライトしても影響はありません。
		1	CLR レジスタに 1 をライトすると、BUSERR レジスタのすべてのビットをクリアします。エラーが検出されると、他のエラーが検出され格納される前に、エラー・レジスタをクリアする必要があります。

一般的な用語と定義

表 A-1 に、本書で使われている一般的な用語を示します。

表 A-1. 一般的な用語と定義

用語	定義
C64x+	新製品 C6000 アーキテクチャを表す一般名称。
C64x+ CPU	CPU ハードウェア（機能ユニットおよびレジスタ）を表します。
C64x+ メガモジュール	C64x+ CPU に加え、サポート・ハードウェア（メモリ、帯域管理、割り込み、メモリ保護、パワーダウン・サポートを行うための）を含みます。
CFG	C64x+ メガモジュール外部のメモリ・マップド・レジスタを含む外部コンフィギュレーション空間。
EMC	拡張メモリ・コントローラ。EMC は、C64x+ メガモジュールから外部環境へデータを転送します。EMC は、外部 DMA と通信するポートを実装しています。EMC クロックは、CPU クロックの 1/2 です。
IDMA	内部 DMA。C64x+ メガモジュールに対してローカルな DMA エンジンです。C64x+ メガモジュールに対してローカルなメモリ（L1P、L1D、L2）と外部コンフィギュレーション空間の間でデータ転送を行うことができます。
L1D	レベル 1 データ・メモリの一般名称。この用語は、メモリ自体またはメモリ・コントローラを表す場合があります。
L1P	レベル 1 プログラム・メモリの一般名称。この用語は、メモリ自体またはメモリ・コントローラを表す場合があります。
L2	レベル 2 メモリの一般名称。この用語は、メモリ自体またはメモリ・コントローラを表す場合があります。

キャッシュに関する用語と定義

表 B-1 に、本書で使われる C64x+ メモリ・アーキテクチャに関連するキャッシュ関連の用語を示します。

表 B-1. キャッシュ関連の用語と定義

用語	定義
DMA	ダイレクト・メモリ・アクセス。典型的に、DMA はあるアドレス範囲からもう一方のアドレス範囲にメモリ・ブロックをコピーします。あるいはペリフェラルとメモリ間でデータを転送します。C64x DSP では、DMA 転送はエンハンスト DMA（EDMA）エンジンが行います。DMA 転送は、プログラム実行と並行して起こります。キャッシュ・コヒーレンスの観点から、EDMA アクセスは並列プロセッサによるアクセスと見なすことができます。
LRU	最低使用頻度。LRU 置換えポリシーの説明については、*最低使用頻度アロケーション*を参照してください。単体で使用されるとき一般に LRU は、*セット*の中で最低使用頻度の*ライン*を識別するために、キャッシュが保持しているステータス情報として参照されます。たとえば、「あるキャッシュ・ラインをアクセスすると、そのラインの LRU を更新します」というフレーズを考えてみてください。
アソシエイティビティ	各セット内の*ライン・フレーム*数。この数値は、キャッシュ内の*ウェイ*数ともいいます。
アロケーション	新しくキャッシュされるデータを格納するためのロケーションを検索するプロセス。このプロセスには、新しいデータ用の空きを作るために、現在キャッシュされているデータを*追い出す*ことを含んでいます。
インバリデート	あるキャッシュ内の*有効*なキャッシュ・*ライン*をインバリデートとマークするプロセス。このアクションは単に該当するキャッシュ・ラインの内容を破棄するだけで、いかなる更新データもライトバックしません。*ライトバック*と組み合わせると、所定のレベルのメモリからキャッシュされたデータを完全に除去する一方で、そのデータを保持している次の下位レベルのメモリを効果的に更新します。ライトバックと組み合わされるインバリデートは、*ライトバック・インバリデート*として参照され、一般にキャッシュ間の*コヒーレンス*を保持するために使用されます。
ウェイ	セット・アソシアティブ・キャッシュでは、キャッシュ内の各セットは複数のライン・フレームで構成されています。各セットのライン・フレーム数はキャッシュのウェイ数として参照されます。キャッシュ内のすべてのセットを通じて対応するライン・フレームの集合を、そのキャッシュのウェイと呼びます。たとえば、4 ウェイ・セット・アソシアティブ・キャッシュは 4 個のウェイをもっていて、キャッシュ内の各セットは関連する 4 個のライン・フレーム（4 ウェイのそれぞれと関連する）をもっています。結果的に、メモリ・マップ内のどのキャッシュ可能なアドレスも、4 ウェイ・アソシアティブ・キャッシュにマップできる 4 つの可能なロケーションがあります。
追い出し	新しくキャッシュされるデータ用に空きを作るために、キャッシュからラインを削除するプロセス。追い出しはまた、ユーザー制御により、キャッシュへアドレスまたはアドレス範囲に*ライトバック・インバリデート*をリクエストすることにより発生させることもできます。追い出されたラインは、*ビクティム*として参照されます。ビクティム・ラインが*ダーティ*な（すなわち、更新されたデータを含んでいる）場合は、*コヒーレンシ*を保つためにそのデータを次のレベルのメモリに書き出します。
下位レベルのメモリ	階層的メモリ・システムでは、CPU からより遠くにあるメモリが下位レベルのメモリです。C64x DSP システムでは、レベル 2（L2）より下のシステム・メモリ、およびすべてのメモリ・マップド・ペリフェラルが、階層における最下位レベルになります。
競合性ミス	キャッシュ・*ミス*の一種で、容量上の制約ではなく、キャッシュの*アソシエイティビティ*の制約により発生します。*フル・アソシエイティブ・キャッシュ*は、新しくキャッシュされる*ライン*を、キャッシュ内のどこにでもアロケートすることができます。ほとんどのキャッシュには、アソシエイティビティの制約があります（*セット・アソシアティブ・キャッシュ*を参照してください）。そのため、データを置く場所に制限が加わり、柔軟性のあるキャッシュでは起こりえない、キャッシュ・ミスが生じます。
クリーン	*有効*で、かつ上位レベルのメモリまたは CPU によってライトされていないキャッシュ・*ライン*。有効なキャッシュ・ラインの逆の状態は、*ダーティ*です。
コヒーレンス	簡略的には、あるデータ項目に対するいかなるリードでも、最後にライトされたデータの値を読み戻す場合は、メモリ・システムはコヒーレントです。これには、CPU および EDMA によるアクセスが含まれます。

表 B-1. キャッシュ関連の用語と定義（続き）

用語	定義
コンパルソリー・ミス	*初期参照ミス*と言われることもあります。コンパルソリー・ミスとは、キャッシュ・*ミス*の一種で、データが以前にキャッシュにアロケートされる機会がなかったために必ず起こるキャッシュ・ミスです。通常は、特定のデータに対するコンパルソリー・ミスは、そのデータへの初回のアクセスで発生します。しかし、たとえそのデータに対する初回の参照でない場合でも、コンパルソリーと考えられるいくつかのケースがあります。そのようなケースには、*ライト・アロケート*されていないキャッシュの同一ロケーションに対して繰り返されるライト・ミス、およびキャッシュ不可能なロケーションに対するキャッシュ・ミスが含まれます。*容量性ミス*および*競合性ミス*と比較してください。
最低使用頻度（LRU）アロケーション	*セットアソシアティブ・キャッシュ*および*フルアソシアティブ・キャッシュ*では、キャッシュ内で空間をアロケートするとき、最低使用頻度アロケーションがセット内の*ライン・フレーム*の選択方法として使われます。セット内のすべてのライン・フレームがアドレスにマップされ*有効*なデータを含んでいるとき、リードまたはライトを最も長い時間実行していない（前の実行から最も長い時間が経過した）セットのライン・フレームが、新しくキャッシュされるデータを保持するために選択されます。そして、新しいデータ用の空きを作るために、選択されたライン・フレームが*追い出されます*。
初期参照ミス	あるデータに対する初回の参照で発生するキャッシュ・*ミス*。初期参照ミスは、*コンパルソリー・ミス*形式の一種です。
実行パケット	単一サイクル中に並行して実行を開始する命令から構成されるブロック。各実行パケットには、1 個から 8 個の命令が含まれています。
上位レベルのメモリ	階層的メモリ・システムでは、CPU により近いメモリが上位レベルのメモリです。メモリ階層において、最上位レベルは通常 1 次キャッシュです。このレベルのメモリは、CPU の直下にあります。上位レベルのメモリは、通常、下位レベルのメモリからのデータに対するキャッシュとして作動します。
スヌープ	あるアドレスのデータを*上位レベルのメモリ*がもっているかどうかを判別するために、*下位レベルのメモリ*が上位レベルのメモリに問い合わせる手法。スヌープの第 1 の目的は、下位レベルのメモリが上位レベルのメモリに更新をリクエストすることにより*コヒーレンシ*を保つことです。スヌープ動作は、*ライトバック*、またはより一般的にライトバック・インバリデートをトリガします。*ライトバック・インバリデート*をトリガするスヌープは、スヌープ・インバリデートと呼ばれることがあります。
スラッシュ	あるアクセス・パターンがキャッシュのパフォーマンスを著しく低下させる原因であるとき、そのアルゴリズムはキャッシュをスラッシュすると言われます。スラッシュは複数の理由により発生します。その 1 つは、短時間にアルゴリズムがデータ、あるいはプログラム・コードに再利用が非常に少ないか全くなしで、過多なまでにアクセスしていることです。すなわち、そのワーキング・セットが大き過ぎて、アルゴリズムが多数の容量性ミスを引き起こしています。ほかにも、アルゴリズムでアドレスが異なるが、すべてキャッシュ内の同じセットにマップされる小グループを、繰り返しアクセスしていると、大量の競合性ミスの原因になります。
セット	キャッシュ内で、ある単一のアドレスが潜在的に存在する可能性のある*ライン・フレーム*の集合。*ダイレクト・マップド・キャッシュ*はセットあたり 1 ライン・フレームで構成され、また N ウェイ・*セット・アソシアティブ・キャッシュ*はセットあたり N 個のライン・フレームで構成されています。*フルアソシアティブ・キャッシュ*は、キャッシュ内のすべてのライン・フレームで構成される 1 つのセットのみをもっています。
セット・アソシアティブ・キャッシュ	セット・アソシアティブ・キャッシュは、各*下位レベルのメモリ*のロケーションを保持することが可能な、複数の*ライン・フレーム*で構成されています。新しいデータ・*ライン*に空きを割り当てるとき、その選択はキャッシュに対する*アロケーション*・ポリシーに基づいて行われます。C64x デバイスではセット・アソシアティブ・キャッシュに、*最低使用頻度（LRU）アロケーション*・ポリシーを使用しています。
タグ	特定ラインにおいてストアされているアドレスの最上位ビットを含む記憶エレメント。タグ・アドレスは、CPU から直接見ることのできない特別なタグ・メモリにストアされています。アクセスが*ヒット*か*ミス*かの判別をするために、アクセスごとにキャッシュはタグ・メモリに照会します。
タッチ	あるアドレスへのメモリ操作は、そのアドレスにタッチすると言われます。タッチはまた、特定のレベルのキャッシュにアロケートするためだけに、配列エレメントまたはメモリ・アドレスのリードと言われることもあります。キャッシュにアロケートする目的で、メモリのある範囲をタッチするために使用される、CPU によるループは多くの場合タッチ・ループといわれます。配列にタッチすることは、ソフトウェア制御によるデータのプリフェッチの一形式です。
ダーティ	*ライトバック・キャッシュ*では、メモリ階層内のあるレベルに到達したライトは、そのレベルを更新しますが、その下位レベルまでは更新しません。したがって、キャッシュ・*ライン*が*有効*で、次の下位レベルに送られていない更新を含んでいる場合、そのラインはダーティであると言います。有効なキャッシュ・ラインで逆の状態は、*クリーン*と言います。
ダイレクト・マップド・キャッシュ	ダイレクト・マップド・キャッシュは、*下位レベルのメモリ*の各アドレスをキャッシュの単一のロケーションにマップします。キャッシュ内の同一ロケーションに複数のロケーションがマップされることがあります。これは、キャッシュのロケーションのセットからデータを置く場所を 1 ヶ所選択する、マルチ・ウェイ・*セット・アソシアティブ・キャッシュ*と対照的です。ダイレクト・マップド・キャッシュは、1 ウェイ・セット・アソシアティブ・キャッシュと考えることができます。

表 B-1. キャッシュ関連の用語と定義（続き）

用語	定義
ヒット	キャッシュ・ヒットは、リクエストされたメモリのロケーションのデータがキャッシュ内にあるときに起こります。ヒットの反対は、*ミス*です。キャッシュ・ヒットにより、ソース・メモリからよりも格段に早くキャッシュからデータをフェッチできるので、ストールを最少に抑えることが可能です。ヒットとミスの決定は、メモリ階層の各レベルで個々に行われます。あるレベルでのミスが、それより下位のレベルではヒットの場合もあります。
ビクティム	新しいライン用のスペースがセットにアロケートされ、そのアドレスに対応したセット内のすべてのライン・フレームが有効なデータを保持しているとき、新しいデータ用の空きを作るために、キャッシュ・コントローラはラインを追い出すための有効ラインを 1 つ選択しなくてはなりません。通常は、最低使用頻度（LRU）ラインが選択されます。追い出されるラインは、ビクティム・ラインとして知られます。ビクティム・ラインがダーティの場合、その内容はビクティム・ライトバックを使用して、次の下位レベルのメモリにライトされます。
ビクティム・バッファ	ビクティムがライトバックされるまで、それらを保持する特別なバッファ。新たに入ってくるデータ用にキャッシュに空きを作るために、ビクティム・ラインをビクティム・バッファに移動します。
ビクティム・ライトバック	ダーティ・ラインが追い出されるとき（すなわち、更新データをもつラインが追い出される）、更新データは下位レベルのメモリにライトされます。このプロセスはビクティム・ライトバックとして参照されます。
フェッチ・パケット	単一サイクルでフェッチされる 8 個の命令のブロック。各フェッチ・パケットは複数の*実行パケット*から構成される場合があります。そのような場合、パケットを使いきるのに複数のサイクルを要します。
フルアソシアティブ・キャッシュ	キャッシュ内のいかなるロケーションにもすべてのメモリ・アドレスがストア可能なキャッシュ。そのようなキャッシュは柔軟性に富んでいますが、ハードウェアに作りこむのは通常、現実的ではありません。フルアソシアティブ・キャッシュは、より多くの制約のある*アロケーション・ポリシー*をもつ、*ダイレクト・マップド・キャッシュ*および*セット・アソシアティブ・キャッシュ*とは明確な対照をなしています。ダイレクト・マップまたはセット・アソシアティブ・キャッシュのパフォーマンスを分析する場合に、フルアソシアティブ・キャッシュは*競合性ミス*と*容量性ミス*を概念的に区別するのに役に立ちます。セット・アソシアティブ・キャッシュの用語において、フルアソシアティブ・キャッシュは*ライン・フレーム*と同数の*ウェイ*で、1 つの*セット*だけをもつセット・アソシエイティブ・キャッシュと等価です。
ミス	リクエストされたメモリ・ロケーションのデータがキャッシュ内にないとき、キャッシュ・ミスが起こります。ミスにより*ライン・フレーム*がアロケートされ、データが次の下位レベルのメモリからフェッチされる間、リクエスタがストールされます。L1D からの CPU ライト・ミスのような場合では、厳密に CPU をストールさせる必要がないこともあります。キャッシュ・ミスは、3 つのカテゴリー（*コンパルソリー・ミス*、*競合性ミス*、および*容量性ミス*）に分類されます。
ミス・パイプライン化	1 つのキャッシュ・*ミス*をサービスするプロセスは、数サイクルにわたりパイプライン化されます。ミスが連続して起こる場合、ミスをパイプライン化することで、いくつかのミスの処理をオーバーラップすることが可能になります。結果として、続いて起こるミスのオーバーヘッドは見かけ上なくなり、付加的なミスによるストール・ペナルティの増分は、1 つのミスを個別に処理する場合の増分よりもはるかに減少します。
メモリ・オーダリング	どのような順序でメモリ操作の効果がメモリ内に現われるか定義します（コンシステンシーと呼ばれることもあります）。メモリ階層内の特定レベルにおけるストロング・メモリ・オーダリングは、プログラムの順序と異なる順序でのそのレベルのメモリへのメモリ・アクセスの効果を観測することができないことを示しています。リラックス・メモリ・オーダリングでは、メモリ階層で異なる順序でのメモリ操作の効果が観測できるようにします。ストロング・メモリ・オーダリングは、メモリ・システムがプログラムの順序通りにメモリ操作を実行することをリクエストしているのではなく、プログラムの順序と一致する順序で、それらの効果が他のリクエスタから観測可能であることだけに注意してください。C64x DSP のメモリ階層が提供するメモリ・オーダリングの保証については、8.3 節で説明します。
有効	キャッシュ・ラインが次のレベルのメモリからフェッチされたデータを保持しているとき、そのライン・フレームは有効です。ライン・フレームがデータを保持していないとき、インバリデート状態が発生します。これは、まだ何もキャッシュされていないか、あるいはコヒーレンス・プロトコル、プログラム・リクエストなど何らかの理由により、以前キャッシュされたデータがインバリデートになった場合のどちらかが原因です。有効の状態は、下位レベルのメモリからフェッチされた後で、データが変更されているかどうかについては示しません。これは、ラインがダーティ、またはクリーンという状態によって表されます。
容量性ミス	キャッシュに、あるプログラムのワーキング・セット全体を保持するのに十分な空き領域がないために起こるキャッシュ・*ミス*。*コンパルソリー・ミス*および*競合性ミス*と比較してください。
ライト・アロケート	ライト・アロケート・キャッシュは、ライト・ミスが発生するとキャッシュ内にスペースをアロケートします。スペースはキャッシュのアロケーション・ポリシー（たとえば LRU）に従ってアロケートされ、ラインのデータは次の下位レベルのメモリからリードされます。キャッシュにデータがあると、ライトが処理されます。ライトバック・キャッシュでは、現在のレベルのメモリだけが更新されます。ライト・データは、すぐには次のレベルのメモリに渡されません。

表 B-1. キャッシュ関連の用語と定義（続き）

用語	定義
ライト・マージング	ライト・マージングは、複数の独立したライトを、単一のより大きなライトに結合します。これは、処理の必要な個別のメモリ・アクセス数を削減することで、メモリ・システムのパフォーマンスを改善します。たとえば、C64x デバイスでは、同じダブルワード・アドレスへのライトの場合、L1D ライト・バッファはいくつかの条件のもとで複数のライトをマージすることが可能です。この例ではその結果として、より大きく効果的なライト・バッファのキャパシティが得られ、また L2 に対するバンド幅のインパクトを軽減することができます。
ライトスルー・キャッシュ	ライトスルー・キャッシュは、すべてのライトを下位レベルのメモリに渡します。下位レベルのメモリに渡されなかった更新データは、ありません。結果的に、ライトスルー・キャッシュ内のキャッシュ・ラインは、決してダーティになることはありません。C64x デバイスは、ライトスルー・キャッシュを利用していません。
ライトバック	有効だがダーティなキャッシュ・ラインから、下位レベルのメモリに更新データをライトする処理。ライトバックが起こった後、そのキャッシュ・ラインはクリーンと見なされます。インバリデートと組み合わされない限り（ライトバック・インバリデート）、ライトバック後そのラインは有効のままです。
ライトバック・インバリデート	ライトバック操作とそれに続くインバリデート。ライトバックおよびインバリデートを参照してください。C64x デバイスでは、キャッシュ・ラインのグループに対するライトバック・インバリデートは、ダーティなキャッシュ・ラインのデータのみ書き出し、対象キャッシュ・ラインのすべての内容がインバリデートされます。
ライトバック・キャッシュ	ライトバック・キャッシュは、ライト・ヒット時に自身のデータを変更するだけです。更新を直に次の下位レベルのメモリには送りません。そのデータは将来のある時点でライトバックされます。ライトバックされるのは、キャッシュ・ラインが追い出されるとき、または下位レベルのメモリが上位レベルのメモリからアドレスをスヌープするときのような場合です。また、キャッシュ・コントロール・レジスタを使用して、ある範囲のアドレスに対してライトバックを直接起動することも可能です。ライトバック・キャッシュに対するライト・ヒットは、対応するラインがダーティとしてマークされる原因になります。すなわち、そのラインには下位レベルのメモリにまだ送っていない更新が含まれています。
ライン	キャッシュ・ラインはキャッシュが扱う最も小さなデータ・ブロックです。キャッシュ・ラインは、CPU またはすぐ上位レベルのメモリからのデータ・アクセスのサイズよりも大きいのが一般的です。たとえば、CPU がメモリから 1 バイトをリクエストする場合であっても、リード・ミス時のキャッシュは、そのリクエストを満たすために、1 ライン分のデータをリードします。
ライン・サイズ	1 キャッシュ・*ライン*のバイト単位でのサイズ。
ライン・フレーム	キャッシュ・データ（1 *ライン*）、関連する *タグ*・アドレス、およびそのラインのステータス情報を保持するキャッシュ内のロケーション。ステータス情報には、そのラインが *有効*なのか、*ダーティ*なのか、また *最低使用頻度*（LRU）の現在状況を含みます。
リード・アロケート	リード・アロケート・キャッシュは、リード・ミス時にキャッシュ内にスペースをアロケートします。キャッシュが *ライト・アロケート*・キャシュでない限り、ライト・ミスは *アロケート*を起こす要因にはなりません。ライト・アロケートを行わないキャッシュでは、ライト・データは次の下位レベルのキャッシュに渡されます。
ロード・スルー	CPU からのリクエストが 1 次、および 2 次キャッシュの両方でミスすると、そのデータは外部メモリからフェッチされ、そして 1 次、および 2 次キャッシュの両方に同時にストアされます。データをストアし、それと同時に上位レベルのキャッシュにそのデータを送るキャッシュがロード・スルー・キャッシュです。最初に下位レベルにデータをストアし、次のステップで上位レベルのキャッシュにデータを送るキャッシュと比較すると、ロード・スルー・キャッシュはストール時間を軽減します。
ロング・ディスタンス・アクセス	キャッシュ不可能なメモリに対して、CPU が行うアクセス。ロング・ディスタンス・アクセスは、キャッシュ可能とマークされていない外部メモリをアクセスするときに使用されます。
ワーキング・セット	プログラムまたはアルゴリズムのワーキング・セットとは、特定の時間周期内に参照されるデータとプログラム・コードの全体のことです。ワーキング・セットについて、上位レベルのメモリを分析するときは各アルゴリズムを基にし、また下位レベルのメモリを分析するときは全体のプログラムを基にして考えることは多くの場合役に立ちます。

改訂履歴

表 C-1 に、本書の旧版からの変更点を示します。

表 C-1. 資料改訂履歴

参照先	追加 / 変更 / 削除
2.3.1.2 項	サブセクションの追加。
7.1.2 項	注の追加
表 7-3	表の内容を変更。
7.5.4 項	サブセクションの変更。

日本テキサス・インスツルメンツ株式会社

本　　社　〒160-8366　東京都新宿区西新宿6丁目24番1号　西新宿三井ビルディング3階　☎03(4331)2000(番号案内)

西日本ビジネスセンター　〒530-6026　大阪市北区天満橋1丁目8番30号　OAPオフィスタワー26階　☎06(6356)4500(代　表)
名古屋オフィス　〒460-0003　名古屋市中区錦2丁目4番3号　錦パークビル7階　☎052(232)5601(代　表)
工　　場　大分県・日出町／茨城県・美浦村
研究開発センター　茨城県・つくば市（筑波テクノロジー・センター）／神奈川県・厚木市（厚木テクノロジー・センター）

■お問い合わせ先
日本TIプロダクト・インフォメーション・センター（PIC）――――― FAX 0120-81-0036
URL: http: //www.tij.co.jp/pic/

TMS320C64x+ DSP
メガモジュール
リファレンス・ガイド

第 1 版 2008 年 5 月

発行所 **日本テキサス・インスツルメンツ株式会社**
〒160-8366
東京都新宿区西新宿 6-24-1 (西新宿三井ビルディング)

Made in Japan
’08.05

日本テキサス・インスツルメンツ株式会社

JAJU058