Cortex™-A9

リビジョン：r2p2

テクニカルリファレンス マニュアル

ARM®

ARM DDI 0388FJ (ID013111)

Cortex-A9
テクニカルリファレンス マニュアル

リリース情報

本書には次の変更が加えられています。

改訂履歴

日付	変更箇所	公開の有無	変更内容
2008 年 3 月 31 日	A	公開	r0p0 用の最初のリリース
2008 年 7 月 8 日	B	公開版、限定アクセス	r0p1 用の最初のリリース
2008 年 12 月 17 日	C	公開版、限定アクセス	r1p0 用の最初のリリース
2009 年 9 月 30 日	D	公開版、限定アクセス	r2p0 用の最初のリリース
2009 年 11 月 27 日	E	公開	r2p0 用の 2 番目のリリース
2010 年 4 月 30 日	F	公開	r2p2 用の最初のリリース

著作権

機密保持ステータス

本書は非機密扱いであり、本書を使用、複製、および開示する権利は、ARM および ARM が本書を提供した当事者との間で締結した契約の条項に基づいたライセンスの制限により異なります。

製品ステータス

本書の情報は最終版であり、開発済み製品に対応しています。

Web アドレス

http://www.arm.com

目次
Cortex-A9 テクニカルリファレンス マニュアル

表一覧
Cortex-A9 テクニカルリファレンス マニュアル

図一覧
Cortex-A9 テクニカルリファレンス マニュアル

序章

本章では、*Cortex-A9 テクニカルリファレンス マニュアル (TRM)* の概要について説明します。本章は次のセクションから構成されています。

- *「本書について」*（ページ xii）
- *「ご意見・ご質問」*（ページ xvii）

本書について

本書は、Cortex-A9 プロセッサ用のテクニカルリファレンス マニュアルです。

製品リビジョンステータス

r*n*p*n* 識別子は、本書に記載されている製品のリビジョンステータスを示しています。各識別子の意味は次のとおりです。

r*n* 製品が大幅に修正されたことを示しています。

p*n* 製品に小さな修正または変更が加えられたことを示しています。

対象読者

本書は、Cortex-A9 システムの実装に携わるハードウェア / ソフトウェアエンジニアを対象に書かれています。プロセッサをターゲットシステムに統合するために必要な情報を設計者に提供します。

注

- Cortex-A9 プロセッサはシングルコア プロセッサです。
- マルチプロセッサバリアントの Cortex-A9 MPCore™ プロセッサは、1 つ～ 4 つの Cortex-A9 プロセッサと *スヌープ制御ユニット* (SCU) で構成されています。詳細については、*『Cortex-A9 MPCore テクニカルリファレンス マニュアル』* を参照して下さい。

本書の構成

本書は以下の章に分かれています。

第 1 章 *はじめに*

Cortex-A9 プロセッサを紹介し、主な機能ブロックについて説明します。

第 2 章 *機能の説明*

Cortex-A9 の機能について説明します。

第 3 章 *プログラマモデル*

Cortex-A9 のレジスタとプログラミングの詳細について説明します。

第 4 章 *システム制御*

Cortex-A9 のシステムレジスタとプログラミングの詳細について説明します。

第 5 章 *Jazelle DBX レジスタ*

CP14 コプロセッサと、その Jazelle DBX のデバッグ以外の使用方法について説明します。

第 6 章 *メモリ管理ユニット*

Cortex-A9 の *メモリ管理ユニット* (MMU) と、アドレス変換処理について説明します。

第 7 章 *レベル 1 メモリシステム*

キャッシュ、*変換ルックアサイドバッファ* (TLB)、ストアバッファを含む、Cortex-A9 のレベル 1 メモリシステムについて説明します。

第 8 章 *レベル 2 メモリインタフェース*

Cortex-A9 のレベル 2 メモリインタフェース、AXI インタフェース属性、および STRT 命令に関する情報について説明します。

第 9 章 *プリロードエンジン*

プリロードエンジン (PLE) と PLE 操作について説明します。

第 10 章 *デバッグ*

Cortex-A9 のデバッグサポートについて説明します。

第 11 章 *パフォーマンス監視ユニット*

パフォーマンス監視ユニット (PMU) と関連イベントについて説明します。

付録 A *信号の説明*

Cortex-A9 の信号の概要について説明します。

付録 B *命令サイクルタイミング*

Cortex-A9 の命令サイクルタイミングについて説明します。

付録 C *リビジョン*

本書の各版における技術的な変更点について説明します。

用語集 本書で使用されている用語の定義について説明します。

表記規則

本書では次の表記規則が採用されています。
- *「書体の一般的な規則」*
- *「タイミング図」*
- *「信号」* (ページ xv)

書体の一般的な規則

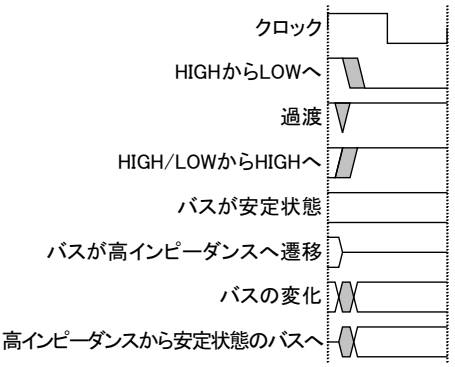
本書で使用されている書体の一般的な規則は次のとおりです。

斜体	重要な注釈の強調、特別な用語の初出時、本書内での相互参照と引用に使用されます。
太字	メニュー名などのインタフェース要素を強調するために太字が使用されます。信号名を示すためにも使用されています。また、必要に応じて説明表の用語にも太字が使用されています。
`monospace`	コマンド、ファイル名、プログラム名、ソースコードなどの、キーボードから入力可能なテキストを示しています。
`mono`space	コマンドまたはオプションに使用可能な略語を示しています。コマンドやオプションの名前を全部入力する代わりに、下線部分のテキストだけを入力してこれらを指定できます。
`monospace italic`	具体的な値に置き換えられる引数を示しています。
`monospace bold`	サンプルコード以外で使用されている場合、言語のキーワードを示しています。
< および >	コードまたはコード片の中で不等号の括弧で囲まれている部分は、アセンブラ構文内で置き換え可能なことを示しています。次に例を示します。 • `MRC p15, 0 <Rd>, <CRn>, <CRm>, <opc2>`

タイミング図

*「タイミング図の表記に使用される記号」*は、タイミング図で使用される構成要素を示しています。この図と異なる意味で使用されている場合は、その都度明記しています。タイミング図に明示されていないタイミング情報については、推測で判断しないで下さい。

バスと信号で影が付いている部分は定義されていないため、その時点のバスと信号は、影付きの領域内の任意の値を取り得ます。実際のレベルは重要ではなく、通常の動作には影響しません。



タイミング図の表記に使用される記号

信号

信号の表記規則は次のとおりです。

信号レベル アサートされる信号レベルは、その信号がアクティブ HIGH かアクティブ LOW かに依存します。「アサートされた」とは、次の状態を意味します。

- アクティブ HIGH の信号が HIGH の状態
- アクティブ LOW の信号が LOW の状態

小文字の n アクティブ LOW 信号の信号名の最初または最後に付加されます。

参照資料

このセクションでは、ARM Limited やサードパーティが発行している出版物を紹介します。

ARM の出版物は Infocenter , http://infocenter.arm.com で参照できます。

ARM の刊行物

本書には、この製品に固有の情報が記載されています。他の関連情報については、以下の出版物を参照して下さい。

- *ARM アーキテクチャ リファレンスマニュアル、ARMv7-A および ARMv7-R エディション* (ARM DDI 0406)
- *Cortex™-A9 MPCore テクニカルリファレンス マニュアル* (ARM DDI 0407)
- *Cortex-A9 浮動小数点 (FPU) テクニカルリファレンス マニュアル* (ARM DDI 0408)
- *Cortex-A9 NEON® メディア処理エンジン テクニカルリファレンス マニュアル* (ARM DDI 0409)
- *Cortex-A9 構成およびサインオフ ガイド* (ARM DII 00146)
- *Cortex-A9 MBIST コントローラ テクニカルリファレンス マニュアル* (ARM DDI 0414)
- *CoreSight™ PTM™-A9 テクニカルリファレンス マニュアル* (ARM DDI 0401)
- *CoreSight PTM-A9 統合マニュアル* (ARM DII 0162)
- *CoreSight プログラムフロー トレース™ アーキテクチャ仕様、v1.0*、v1.0 (ARM IHI 0035)
- *AMBA® レベル 2 キャッシュコントローラ (L2C-310) テクニカルリファレンス マニュアル* (ARM DDI 0246)
- *AMBA AXI プロトコル v10 仕様* (ARM IHI 0022)
- *ARM 汎用割り込みコントローラアーキテクチャ仕様* (ARM IHI 0048)
- *PrimeCell® 汎用割り込みコントローラ (PL390) テクニカルリファレンス マニュアル* (ARM DDI 0416)
- *RealView ICE ユーザガイド* (ARM DUI 0155)
- *CoreSight アーキテクチャ仕様* (ARM IHI 0029)
- *CoreSight テクノロジシステム設計ガイド* (ARM DGI 0012)
- *ARM デバッグインタフェース v5 アーキテクチャ仕様* (ARM IHI 0031)
- *The ARM Cortex-A9 Processors* ホワイトペーパー

その他の刊行物

このセクションでは、サードパーティが発行している関連出版物を紹介します。

- *ANSI/IEEE Std 754-1985、IEEE Standard for Binary Floating-Point Arithmetic*
- *IEEE Std 1500-2005、IEEE Standard Testability Method for Embedded Core-based Integrated Circuits*

ご意見・ご質問

ARM では、本製品と本書に関するご意見をお待ちしております。

製品に関するご意見

本製品に関するご意見・ご質問がございましたら、次の情報とともに製品購入元までご連絡下さい。

- 製品名
- 製品のリビジョンまたはバージョン
- できるだけ詳細な説明。該当する場合には、現象もご記載下さい。

本書に関するご意見

本書に関するご意見がございましたら、電子メールに次の情報をご記入の上、errata@arm.com までお寄せ下さい。

- 題名
- 資料番号、ARM DDI 0388FJ
- ご意見のあるページ番号
- ご意見についての簡潔な説明

補足または改善すべき点についての一般的なご意見もお待ちしております。

第 1 章
はじめに

本章では、Cortex-A9 プロセッサとその機能について紹介します。本章は次のセクションから構成されています。

1.1 Cortex-A9 プロセッサについて

Cortex-A9 プロセッサは、完全な仮想メモリ機能を提供するレベル 1 キャッシュサブシステムを搭載した、高性能で低消費電力の ARM マクロセルです。Cortex-A9 プロセッサは、ARMv7-A アーキテクチャを実装し、32 ビット ARM 命令、16 ビットおよび 32 ビット Thumb 命令、および Jazelle 状態での 8 ビット Java™ バイトコードを実行します。

Cortex-A9 ユニプロセッサと、PL390 割り込みコントローラおよび L2C-310 レベル 2 キャッシュコントローラを使用した設計の例を、図 1-1 に示します。

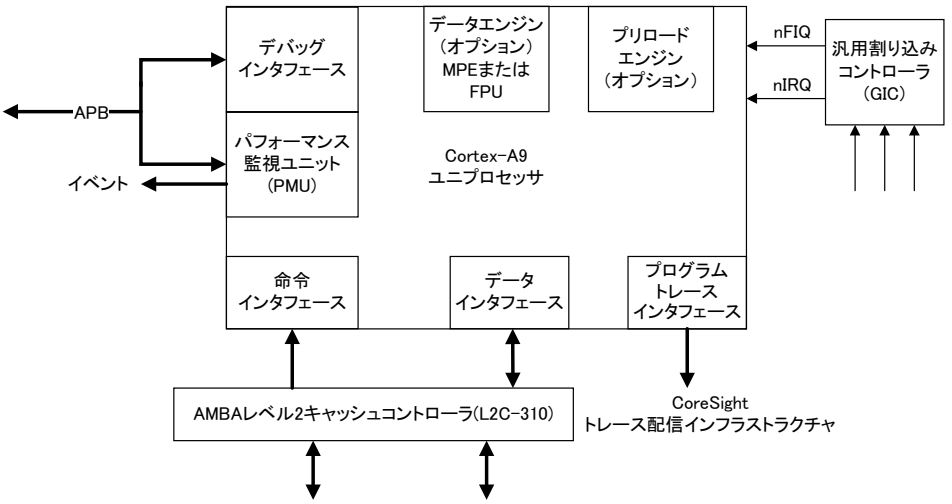


図 1-1 Cortex-A9 ユニプロセッサシステム

1.1.1 データエンジン

設計にはデータエンジンを含めることができます。次に示すセクションでは、データエンジン オプションについて説明します。

- *「メディア処理エンジン」*
- *「浮動小数点ユニット」*

メディア処理エンジン

オプションの *NEON メディア処理エンジン* (MPE) は、ARMv7-A アーキテクチャ用の ARM *アドバンスト単一命令複数データ* (SIMD) メディア処理エンジン拡張機能です。NEON MPE は、整数と浮動小数点のベクタ演算をサポートし、3 次元グラフィックや画像処理などのマルチメディアアプリケーションの性能向上に役立ちます。

NEON MPE オプションを実装すると、プロセッサの機能が拡張され、ARMv7 アドバンスト SIMD および VFPv3 D-32 命令セットがサポートされます。

『Cortex-A9 NEON メディア処理エンジン テクニカルリファレンス マニュアル』 を参照して下さい。

浮動小数点ユニット

設計にオプションの MPE が含まれていない場合、アドバンスト SIMD 拡張機能なしの ARMv7 VFPv3-D16 FPU を含めることもできます。この FPU は、トラップレス実行を提供し、スカラ演算に最適化されています。Cortex-A9 FPU ハードウェアは、非

推奨の VFP ショートベクタ機能をサポートしていません。FPSCR.LEN フィールドが 0 以外のときに VFP データ処理命令の実行を試みると、FPSCR.DEX ビットがセットされ、同期未定義命令例外が取得されます。必要であれば、ソフトウェアでショートベクタ機能をエミュレートできます。

*『Cortex-A9 浮動小数点ユニット テクニカルリファレンス マニュアル』*を参照して下さい。

1.1.2 システム設計コンポーネント

ここでは、図 1-1 （ページ 1-2）に示されている PrimeCell コンポーネントについて説明します。

- *「PrimeCell 汎用割り込みコントローラ」*
- *「AMBA レベル 2 キャッシュコントローラ (L2C-310)」*

PrimeCell 汎用割り込みコントローラ

PrimeCell 汎用割り込みコントローラ (PL390) などの汎用割り込みコントローラを、Cortex-A9 ユニプロセッサに接続することができます。Cortex-A9 MPCore には、PL390 と同じプログラマモデルを共有する割り込みコントローラが内蔵されていますが、細部は実装によって相違点があります。

Cortex-A9 MPCore 割り込みコントローラについては、*『Cortex-A9 MPCore テクニカルリファレンス マニュアル』*を参照して下さい。

AMBA レベル 2 キャッシュコントローラ (L2C-310)

ARM ベースのシステムで、プロセッサにより大量のメモリトラフィックが生成される場合は、パフォーマンス改善のためにオンチップの 2 次キャッシュ（レベル 2 キャッシュ、または L2 キャッシュとも呼ばれます）を追加するのが一般的な手法です。AMBA レベル 2 キャッシュコントローラは、外部メモリへのアクセス数の削減に役立ち、Cortex-A9 プロセッサや Cortex-A9 MPCore プロセッサで使用するために最適化されています。

1.2 Cortex-A9 のバリアント

Cortex-A9 プロセッサは、ユニプロセッサ構成とマルチプロセッサ構成の両方で使用できます。

マルチプロセッサ構成では、レベル 1 データキャッシュ コヒーレンシを維持する*スヌープ制御ユニット* (SCU) の制御下で、最大 4 つの Cortex-A9 プロセッサをキャッシュコヒーレントなクラスタとして使用できます。

Cortex-A9 MPCore マルチプロセッサには、次のコンポーネントが含まれます。

- 最大 4 つの Cortex-A9 プロセッサ
- SCU。次の機能を果たします。
  - レベル 1 データキャッシュ間でのコヒーレンシ維持
  - *アクセラレータコヒーレンシ ポート* (ACP) のコヒーレンシ操作
  - Cortex-A9 MPCore AXI マスタインタフェース上でのトランザクションのルーティング
  - Cortex-A9 ユニプロセッサによるプライベート メモリ領域へのアクセス
- 従来の ARM 割り込みをサポートする*割り込みコントローラ* (IC)
- プロセッサごとに 1 つのプライベートタイマとプライベートウォッチドッグ
- グローバルタイマ
- AXI 高速*アドバンスト マイクロプロセッサ バスアーキテクチャ* (AMBA3) L2 インタフェース
- *アクセラレータコヒーレンシ ポート* (ACP)。これは、DMA エンジンまたは非キャッシュ対象のペリフェラルに接続可能な、オプションの AXI 64 ビット スレーブポートです。

詳細については、*『Cortex-A9 MPCore テクニカルリファレンス マニュアル』*を参照して下さい。

次に示すシステムレジスタには、Cortex-A9 MPCore 専用の使用法があります。

- *「マルチプロセッサ類似性レジスタ」*（ページ 4-10）
- *「補助制御レジスタ」*（ページ 4-18）
- *「構成ベースアドレス レジスタ」*（ページ 4-38）

一部の PMU イベント信号には Cortex-A9 MPCore 専用の使用法があります。*「パフォーマンス監視信号」*（ページ A-14）を参照して下さい。

1.3 準拠性

Cortex-A9 プロセッサは、次のような機能を持つ ARMv7-A アーキテクチャを実装しています。

- 全体的なコード密度が Thumb と同程度で、性能が ARM 命令と同程度の ARM Thumb®-2 32 ビット命令セットアーキテクチャ。ARM 命令セットと Thumb 命令セットについては、*『ARM アーキテクチャ リファレンスマニュアル』*を参照して下さい。
- 実行環境を高速化するための *Thumb 実行環境* (ThumbEE) アーキテクチャ。ThumbEE 命令セットについては、*『ARM アーキテクチャ リファレンスマニュアル』*を参照して下さい。
- セキュリティを強化するためのセキュリティ拡張機能。セキュリティ拡張機能については、*『ARM アーキテクチャ リファレンスマニュアル』*を参照して下さい。
- 3 次元グラフィックや画像処理などのマルチメディアアプリケーションの性能を向上させる、アドバンスト SIMD アーキテクチャ拡張機能。アドバンスト SIMD アーキテクチャ拡張機能については、*『ARM アーキテクチャ リファレンスマニュアル』*を参照して下さい。

  実装固有の情報については、*『Cortex-A9 NEON メディア処理エンジン テクニカルリファレンス マニュアル』*を参照して下さい。
- IEEE 754 標準に完全準拠した浮動小数点演算機能を持つ、*ベクタ浮動小数点バージョン 3*(VFPv3) アーキテクチャ拡張機能。VFPv3 拡張機能については、*『ARM アーキテクチャ リファレンスマニュアル』*を参照して下さい。

  実装固有の情報については、*『Cortex-A9 浮動小数点ユニット テクニカルリファレンス マニュアル』*を参照して下さい。
- セキュリティ拡張機能と CoreSight のサポートを含む、ARMv7 デバッグアーキテクチャ。ARMv7 デバッグアーキテクチャについては、*『ARM アーキテクチャ リファレンスマニュアル』*を参照して下さい。

1.4 機能

Cortex-A9 プロセッサには次のような機能があります。

- 動的分岐予測機能を備えたスーパースカラ、可変長、アウトオブオーダ パイプライン
- ARM アーキテクチャ v7-A 命令セットの完全な実装
- セキュリティ拡張機能
- *メモリ管理ユニット* (MMU) を備えたハーバードレベル 1 メモリシステム
- データ側バス用のマスタ 0 と、命令側バス用のマスタ 1 とで構成される、2 つの 64 ビット AXI マスタインタフェース
- ARMv7 デバッグアーキテクチャ
- *プログラム トレースマクロセル* (PTM) インタフェースによるトレースのサポート
- 最大 3 つの電力ドメインによる高度な電力管理のサポート
- （オプション）プリロードエンジン
- （オプション）Jazelle ハードウェアアクセラレーション
- （オプション）MPE と VFPv3 を搭載したデータエンジン

1.5 インタフェース

プロセッサには次の外部インタフェースがあります。

- AMBA AXI インタフェース
- デバッグ APBv3 外部デバッグインタフェースを含む、v7 準拠のデバッグインタフェース
- DFT

これらのインタフェースの詳細については、*『AMBA AXI プロトコル仕様』*、*『CoreSight アーキテクチャ仕様』*、*『Cortex-A9 MBIST コントローラ テクニカルリファレンス マニュアル』*を参照して下さい。

1.6 構成可能なオプション

Cortex-A9 プロセッサの構成可能なオプションを、表 1-1 に示します。

表 1-1 Cortex-A9 プロセッサの構成可能なオプション

機能	選択可能な範囲	デフォルト値
命令キャッシュサイズ	16KB、32KB、64KB	32KB
データキャッシュ サイズ	16KB、32KB、64KB	32KB
TLB のエントリ数	64 エントリまたは 128 エントリ	128 エントリ
Jazelle アーキテクチャ拡張機能	フルまたはトリビアル	フル
NEON テクノロジを搭載したメディア処理エンジン	ありまたはなし[a]	なし
FPU	ありまたはなし [a]	
PTM インタフェース	ありまたはなし	
電源オフ / 休眠モード用のラッパ	ありまたはなし	
パリティエラー検出のサポート	-	この機能を含めるかどうかは、構成と設計により決定されます。
プリロードエンジン	ありまたはなし	
プリロードエンジンの FIFO サイズ[b]	16、8、または 4 エントリ	16 エントリ
ARM_BIST	ありまたはなし	あり
デザインウェアの使用	使用または未使用	使用

a. MPE オプションと FPU RTL オプションは相互排他的です。MPE オプションを選択した場合は、MPE が VFPv3-D32 FPU とともに実装されます。この場合、FPU RTL オプションは使用できません。MPE RTL オプションが実装されていない場合は、FPU RTL オプションを選択し、VFPv3-D16 FPU を実装することができます。
b. 設計にプリロードエンジンが含まれている場合のみ

MBIST ソリューションは、選択した Cortex-A9 プロセッサのキャッシュサイズに適合するように構成する必要があります。また、Cortex-A9 設計内の RAM ブロックに対する MBIST ソリューションの形式は、プロセッサの実装時に決定する必要があります。

詳細については、*『Cortex-A9 MBIST コントローラ テクニカルリファレンス マニュアル』*を参照して下さい。

1.7 テスト機能

Cortex-A9 プロセッサには、テスト機能はありません。

1.8 製品説明書、設計フロー、アーキテクチャ

ここでは、Cortex-A9 ファミリの説明書、設計フローとの関係、および関連するアーキテクチャの標準とプロトコルについて説明します。

このセクションに記載されている説明書の詳細については、*「参照資料」*（ページ xv）を参照して下さい。

1.8.1 説明書

Cortex-A9 ファミリの説明書は次のとおりです。

テクニカルリファレンス マニュアル

『テクニカル リファレンスマニュアル』 (TRM) には、Cortex-A9 ファミリの機能と、動作に対する機能オプションの影響が記載されています。設計フローのすべての段階で必要となります。Cortex-A9 プロセッサの実装と統合の方法によっては、TRM に記載されている動作の一部が適用されない場合があります。

- *『Cortex-A9 TRM』* には、ユニプロセッサバリアントに関する説明が記載されています。
- *『Cortex-A9 MPCore TRM』* には、Cortex-A9 プロセッサのマルチプロセッサバリアントに関する説明が記載されています。
- *『Cortex-A9 浮動小数点 (FPU) TRM』* には、データエンジンの実装固有の FPU 部分に関する説明が記載されています。
- *『Cortex-A9 NEON メディア処理エンジン テクニカルリファレンス マニュアル』* には、データエンジンのアドバンスト SIMD Cortex-A9 実装固有の部分に関する説明が記載されています。

Cortex-A9 プロセッサのプログラミングに関してのお問い合わせ先は次のとおりです。

- 実装のビルド構成を決定する場合は、実装者にお問い合わせ下さい。
- 使用する SoC のピン構成を決定する場合は、インテグレータにお問い合わせ下さい。

構成およびサインオフ ガイド

『構成およびサインオフ ガイド』 (CSG) には、次の内容が記載されています。

- 利用可能なビルド構成オプションと、それらのオプションの選択に関連する考慮事項
- ビルド構成オプションで *レジスタ転送レベル* (RTL) を構成する方法
- 構成された設計をサインオフするための手順

ARM 製品の配布物には、リファレンススクリプトと、それらを使用して設計を実装する方法についての説明が含まれています。EDA ツールのベンダから入手できるリファレンス手法の説明書は、この CSG を補完するものです。

CSG は非公開書籍で、ライセンスによってのみ入手できます。

1.8.2 設計フロー

プロセッサは、合成可能な RTL として配布されます。プロセッサを製品で使用する前に、次の手順を実行する必要があります。

1. 実装。実装者は、RTL を構成して合成し、ハードマクロセルを製造します。必要に応じて、これに RAM の設計への統合が含まれます。

2. 統合。インテグレータは、実装された設計を SoC に接続します。これには、SoC とメモリシステムおよびペリフェラルとの接続が含まれます。

3. プログラミング。システムプログラマは、プロセッサの構成と初期化に必要なソフトウェアを開発し、必要なアプリケーションソフトウェアをテストします。

手順の各段階は、
- 別の団体が行ってもかまいません。
- 次の段階の動作や機能に影響を与えるオプションを含むことができます。

ビルドの構成

実装者は、RTL ソースファイルが事前処理される方法に影響するオプションを選択します。このオプションにより、特定のロジックが含まれるか、または除外され、結果として生成されるマクロセルの実装面積や最大周波数に影響します。

構成入力

インテグレータは、入力を特定の値に固定することで、プロセッサの機能の一部を構成します。これらの構成は、ソフトウェア構成が行われる前の、起動時の動作に影響します。構成により、ソフトウェアで使用可能なオプションを制限することもできます。

ソフトウェア構成

プログラマは、ソフトウェアから可視のレジスタに特定の値をプログラムして、プロセッサを構成します。この操作は、プロセッサの動作に影響を与えます。

注

本書には、ビルド構成オプションに適用可能な実装定義の機能が記載されています。機能が含まれているという表現は、適切なビルドとピン構成オプションが選択されていることを意味します。機能が稼働しているという表現は、その機能がソフトウェアでも構成されていることを意味します。

1.8.3 アーキテクチャとプロトコルの情報

Cortex-A9 プロセッサは、次に示すセクションで説明されている仕様に準拠しているか、その仕様を実装しています。
- *「ARM アーキテクチャ」*
- *「アドバンスト マイクロコントローラバス アーキテクチャ」*
- *「トレースマクロセル」*（ページ 1-12）
- *「デバッグアーキテクチャ」*（ページ 1-12）

本テクニカルリファレンス マニュアルは、アーキテクチャリファレンス マニュアル、アーキテクチャ仕様、プロトコル仕様、および関連する外部標準を補完するものです。これらのソースに記載されている情報は、本書で繰り返し言及されていません。

ARM アーキテクチャ

Cortex-A9 プロセッサは、ARMv7-A アーキテクチャプロファイルを実装しています。詳細については、*『ARM アーキテクチャ リファレンスマニュアル』*を参照して下さい。

アドバンスト マイクロコントローラバス アーキテクチャ

Cortex-A9 プロセッサは AMBA 3 プロトコルに準拠しています。詳細については、*『AMBA AXI プロトコル仕様』*と*『AMBA 3 APB プロトコル仕様』*を参照して下さい。

トレースマクロセル

v1.0 PFT アーキテクチャです。詳細については、*『CoreSight プログラムフロー トレースアーキテクチャ仕様』*を参照して下さい。

デバッグアーキテクチャ

プロセッサは、ARMv7 デバッグアーキテクチャを実装しており、セキュリティ拡張機能と CoreSight をサポートしています。*『CoreSight アーキテクチャ仕様』*を参照して下さい。

1.9 製品リビジョン

ここでは、プロセッサのリリースごとの機能の違いについて簡単に説明します。

- *「r0p0 と r0p1 の機能的な違い」*
- *「r0p1 と r1p0 の機能的な違い」*
- *「r1p0 と r2p0 の機能的な違い」*（ページ 1-14）

1.9.1 r0p0 と r0p1 の機能的な違い

r0p0 と r0p1 とで、説明されている機能に変更はありません。

この 2 つのリビジョンの唯一の違いは次のとおりです。

- r0p1 では、r0p0 に関する既知のエンジニアリングに関する誤植がすべて修正されています。
- r0p1 では、命令側とデータ側の両方で、マイクロ TLB エントリ数が 8 から 32 にアップグレードされています。

これらの変更はいずれも、本書に記載されている機能に影響しません。

1.9.2 r0p1 と r1p0 の機能的な違い

この 2 つのリビジョンの違いは次のとおりです。

- r1p0 では、r0p1 に関する既知のエンジニアリングに関する誤植がすべて修正されています。
- r1p0 では、**CPUCLKOFF** と **DECLKOFF** を使用して、リセットシーケンス中に Cortex-A9 プロセッサを制御することができます。*「構成信号」*（ページ A-5）を参照して下さい。
  - 設計のマルチプロセッサ実装には、Cortex-A9 プロセッサと同数の **CPUCLKOFF** ピンが存在します。
  - **DECLKOFF** は、リセットシーケンス中のデータエンジン クロックを制御します。
- r1p0 には、Cortex-A9 プロセッサの動的高水準クロックゲートが搭載されています。「動的高水準クロックゲート」（ページ 2-8）を参照して下さい。
  - **MAXCLKLATENCY[2:0]** バスが追加されました。*「構成信号」*（ページ A-5）を参照して下さい。
  - CP15 電力制御レジスタの追加。*「電力制御レジスタ」*（ページ 4-36）を参照して下さい。
- パフォーマンス監視イベントバスの拡張。r1p0 では、**PMUEVENT** が 52 ビット幅になりました。
  - Cortex-A9 固有のイベントの追加。表 2-2（ページ 2-5）を参照して下さい。
  - イベントの説明が拡張されました。表 2-2（ページ 2-5）を参照して下さい。
- **PMUSECURE** と **PMUPRIV** の追加。*「パフォーマンス監視信号」*（ページ A-14）を参照して下さい。
- 128 エントリまたは 64 エントリ用のメイン TLB オプション。*「TLB タイプレジスタ」*（ページ 4-9）を参照して下さい。
- **DEFLAGS[6:0]** が追加されました。「DEFLAGS[6:0]」（ページ 4-37）を参照して下さい。
- 電力管理信号の **BISTSCLAMP** が削除されました。

- スキャンテスト信号の **SCANTEST** が削除されました。
- 第 2 の置換方式の追加。SCTLR.RR ビットで選択されます。*「システム制御レジスタ」*（ページ 4-15）を参照して下さい。
- PL310 キャッシュコントローラの最適化に関する説明の追加。*「レベル 2 メモリインタフェースへのアクセスの最適化」*（ページ 8-7）を参照して下さい。
- DMB の直列化動作の変更。*「直列化命令」*（ページ B-9）を参照して下さい。
- ID レジスタの値が、正しいリビジョンを反映するように変更されました。

1.9.3 r1p0 と r2p0 の機能的な違い

リビジョンの違いは次のとおりです。

- オプションのプリロードエンジン ハードウェアの機能とサポートの追加。
  - NSACR に PLE ビットが追加されました。*「非セキュアアクセス制御レジスタ」*（ページ 4-23）を参照して下さい。
  - プリロードエンジン レジスタが追加されました。*「CP15 c11 レジスタの概要」*（ページ 4-30）を参照して下さい。
  - プリロード操作と MCRR 命令が追加されました。第 9 章 *プリロードエンジン*を参照して下さい。
  - プリロードエンジン イベントの追加。
    「パフォーマンス監視」（ページ 2-3）、表 11-7 （ページ 11-7）、および表 A-18 （ページ A-14）を参照して下さい。
- 電圧ドメインの変更。図 2-4（ページ 2-14）を参照して下さい。
- NEON ビジーレジスタ。*「NEON ビジーレジスタ」*（ページ 4-37）を参照して下さい。
- ID レジスタの値が、正しいリビジョンを反映するように変更されました。

1.9.4 r2p0 と r2p1 の機能的な違い

- なし。

1.9.5 r2p1 と r2p2 の機能的な違い

- なし。説明書の更新と修正のみ。*「D 版と F 版の相違点」*（ページ C-6）を参照して下さい。

第 2 章
機能の説明

本章では、製品の機能について説明します。本章は次のセクションから構成されています。

2.1 機能について

Cortex-A9 プロセッサは、完全な仮想メモリ機能を提供するレベル 1 キャッシュサブシステムを搭載した、高性能で低消費電力の ARM マクロセルです。

Cortex-A9 プロセッサのトップレベル図を、図 2-1 に示します。

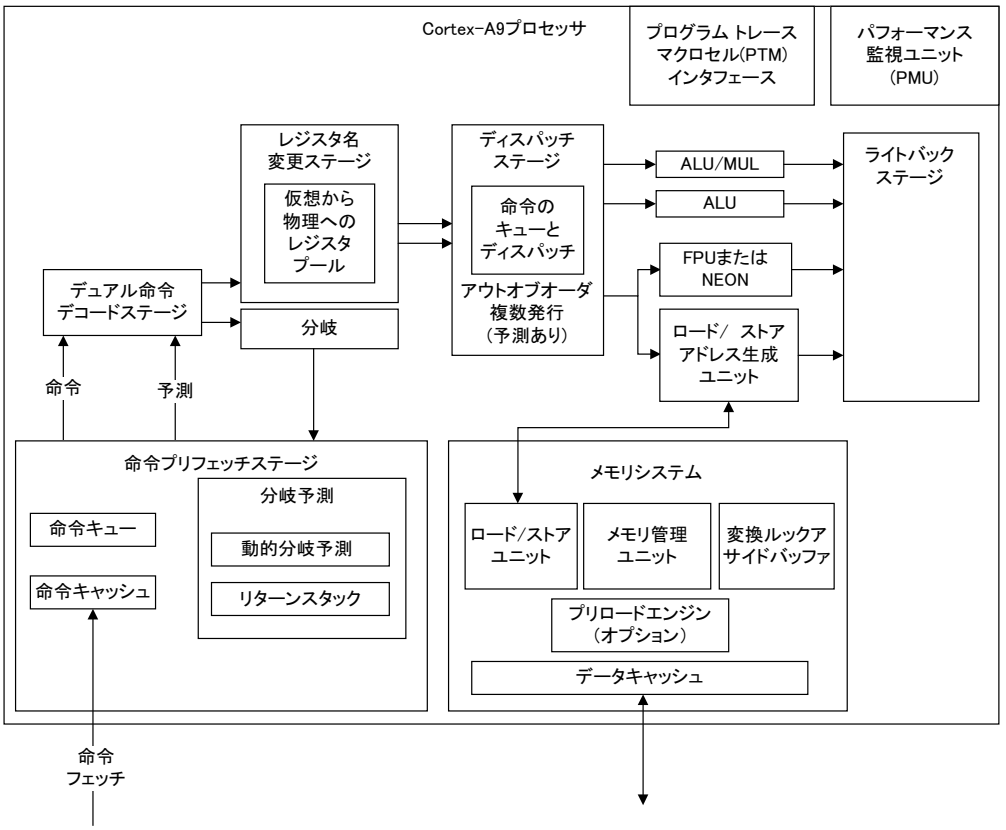


図 2-1 Cortex-A9 プロセッサのトップレベル図

2.1.1 レジスタ名の変更

レジスタ名の変更方式によって、汎用レジスタと、カレントプログラム ステータス レジスタ (CPSR) のフラグビットの*書き込み後書き込み* (WAW) 状態および読み出し後書き込み (WAR) 状態におけるアウトオブ オーダ実行が促進されます。

この方式では、32 個の ARM アーキテクチャレジスタが 56 個の 32 ビット物理レジスタのプールにマッピングされ、8 個の 9 ビット物理レジスタの専用プールを使用して CPSR のフラグ（N、Z、C、V、Q、GE）の名前が変更されます。

2.1.2 命令キュー

命令キューでは、小ループモードによって、小さな命令ループの実行中の消費電力が低減されます。*「エネルギー効率機能」*（ページ 2-10）を参照して下さい。

2.1.3 動的分岐予測

プリフェッチユニットは、*グローバル履歴バッファ* (GHB)、*分岐ターゲット アドレスキャッシュ* (BTAC)、リターンスタックを使用する、2 レベルの動的分岐予測を実装しています。*「レベル 1 命令側メモリシステムについて」*（ページ 7-5）を参照して下さい。

2.1.4 PTM インタフェース

Cortex-A9 プロセッサは、オプションとして、*プログラムフロー トレース* (PFT) 命令専用アーキテクチャプロトコルに準拠した*プログラム トレースマクロセル* (PTM) インタフェースを実装します。ウェイポイントやプログラムフローの変化とともに、コンテキスト ID の変化などのイベントが出力され、トレースとコードイメージの相互関連付けが可能になります。*「プログラムフロー トレースとプログラムトレース マクロセルインタフェース」*（ページ 2-4）を参照して下さい。

2.1.5 パフォーマンス監視

Cortex-A9 プロセッサが提供するプログラムカウンタとイベントモニタは、プロセッサとメモリシステムの動作に関する統計データを収集するように構成可能です。

パフォーマンス監視カウンタと関連する制御レジスタには、CP15 コプロセッサインタフェースと APB デバッグインタフェースからアクセスできます。第 11 章 *パフォーマンス監視ユニット*を参照して下さい。

2.1.6 割り込みの仮想化

割り込みの仮想化により、ゲスト*オペレーティングシステム* (OS) で修正版の例外動作モデルを使用して、割り込み処理を高速化できます。

「仮想化制御レジスタ」（ページ 4-24）を参照して下さい。

仮想化制御レジスタの動作は、プロセッサがセキュア状態にあるか、非セキュア状態にあるかによって異なります。

プロセッサがセキュア状態にあるときに例外が発生した場合、仮想化制御レジスタの AMO、IMO、IFO ビットは無視されます。例外が取得されるかどうかは、CPSR の A、I、F ビットの設定のみによって決定されします。

プロセッサが非セキュア状態にあるときに例外が発生した場合、SCR の EA、FIQ、または IRQ ビットがセットされていなければ、対応する例外が取得されるかどうかは、CPSR の A、I、F ビットの設定のみによって決定されます。

「非セキュアアクセス制御レジスタ」（ページ 4-23）を参照して下さい。

SCR の EA、FIQ、IRQ のいずれかのビットがセットされている場合は、対応する例外がモニタモードにトラップされます。この場合、対応する例外が取得されるかどうかは、仮想化制御レジスタの AMO、IMO、または IFO ビットによりマスクされた、CPSR の A、I、F ビットによって決定されます。

2.2 インタフェース

プロセッサには次の外部インタフェースが存在します。

- AMBA AXI インタフェース
- APB CoreSight インタフェース
- DFT インタフェース

これらのインタフェースの詳細については、*『AMBA AXI プロトコル仕様』*、*『CoreSight アーキテクチャ仕様』*、*『CoreSight PFT アーキテクチャ仕様』*、*『Cortex-A9 MBIST コントローラ テクニカルリファレンス マニュアル』*を参照して下さい。

2.2.1 プログラムフロー トレースとプログラムトレース マクロセルインタフェース

Cortex-A9 プロセッサには、*プログラムフロー トレース* (PFT) アーキテクチャプロトコルも実装されています。次に示すセクションでは、Cortex-A9 の*プログラム トレースマクロセル* (PTM) インタフェースについて説明します。

- *「プログラムフロー トレース」*
- *「プログラム トレースマクロセル信号」*

プログラムフロー トレース

PFT は、ウェイポイントを使用してトレースとコードイメージを関連付ける、命令専用のトレースプロトコルです。ウェイポイントとは、分岐やコンテキスト ID の変更などの、プログラムフローの変更やイベントで、トレースを行うためにために出力する必要のあるものです。。

ウェイポイントを使用したトレース方法については、*『CoreSight プログラムフロー トレースアーキテクチャ仕様』*と*『CoreSight PTM-A9 テクニカルリファレンス マニュアル』*を参照して下さい。

プログラム トレースマクロセル信号

PTM インタフェース信号を、図 2-2 に示します。

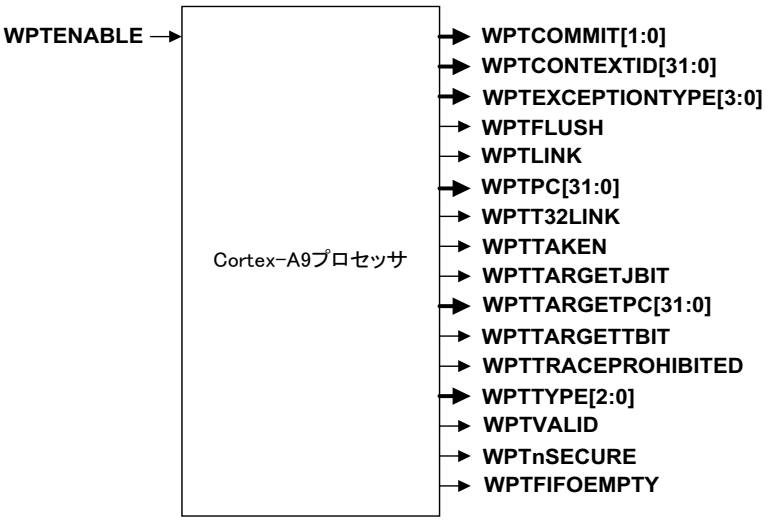


図 2-2 PTM インタフェース信号

詳細については、付録 A *信号の説明*および*『CoreSight PTM-A9 テクニカルリファレンス マニュアル』*を参照して下さい。

禁止領域

トレースは一部の領域で禁止する必要があります。禁止領域の詳細については、*『ARM アーキテクチャ リファレンスマニュアル』*を参照して下さい。Cortex-A9 プロセッサは、トレース、パフォーマンス監視、PC サンプリングを含む、領域内の非侵襲性デバッグ用の禁止領域を判断する必要があります。禁止領域内の命令については、ウェイポイントが生成されません。

Jazelle 状態の開始と終了のみがトレースされます。Jazelle 状態開始のウェイポイントの後には、Jazelle 状態終了のウェイポイントが続きます。

2.3 クロックとリセット

ここでは、プロセッサのクロックとリセットについて説明します。

- *「同期クロック」*
- *「リセット」*
- *「動的高水準クロックゲート」*（ページ 2-8）

2.3.1 同期クロック

Cortex-A9 ユニプロセッサには、**CLK** という機能クロック入力が 1 つ存在します。

Cortex-A9 ユニプロセッサには非同期インタフェースは存在しません。すべてのバスインタフェースと割り込み信号が、**CLK** を基準にして同期している必要があります。

AXI バスクロックドメインは、**ACLKEN** 信号を使用して、n:1（**CLK** を基準とした AXI 対プロセッサの比率）で実行することができます。

Cortex-A9 ユニプロセッサで、**CLK** と **ACLK** のクロック比が 3:1 で使用される **ACKLENM0** のタイミング例を、図 2-3 に示します。

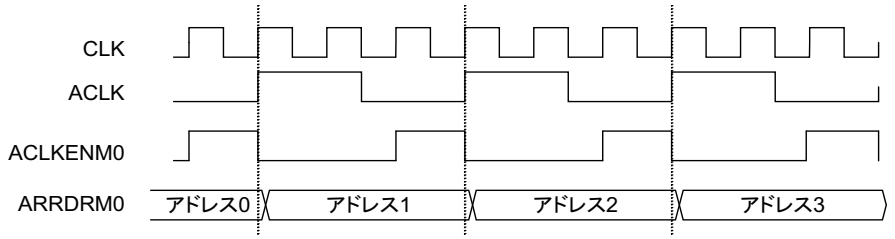


図 2-3 3:1 のクロック比で使用される ACLKENM0

マスタポートの Master0 では、**ACLKENM0** が HIGH の場合の **CLK** 立ち上がりエッジでのみ、AXI 出力が変化します。

2.3.2 リセット

Cortex-A9 プロセッサには次のリセット入力が存在します。

nCPURESET **nCPURESET** 信号は、メイン Cortex-A9 プロセッサリセットです。このリセットによって、MPE または FPU オプションが存在する場合に、Cortex-A9 プロセッサロジックと、FPU レジスタファイルを含む FPU ロジックが初期化されます。

nNEONRESET **nNEONRESET** 信号は、メイン Cortex-A9 プロセッサリセットとは無関係に、NEON SIMD を制御するリセットです。

nDBGRESET **nDBGRESET** 信号は、デバッグロジックを初期化するリセットです。第 10 章 *デバッグ*を参照して下さい。

これらの信号はすべて、アクティブ LOW です。

リセットモード

Cortex-A9 の設計に存在するリセット信号によって、プロセッサの各部を個別にリセットできます。リセット信号と、それらの使用可能な組み合わせと用途を、表 2-1 に示します。

表 2-1 リセットモード

モード	nCPURESET	nNEONRESET	nDBGRESET
パワーオンリセット、コールドリセット	0	0	0
プロセッサリセット、ソフトまたはウォームリセット	0	0	1
SIMD MPE パワーオン リセット	1	0	1
デバッグロジックのリセット	1	1	0
リセットなし、通常の実行モード	1	1	1

パワーオン リセット

システムの電力を最初にオンにするときに、パワーオン リセットまたは*コールド*リセットを、Cortex-A9 ユニプロセッサに適用する必要があります。パワーオン リセットの場合、リセット信号の先行エッジ、つまり立ち下がりエッジは、**CLK** と同期している必要はありませんが、立ち上がりエッジは同期している必要があります。

正しいリセット動作を保証するために、リセット信号を 9 **CLK** サイクル以上アサートする必要があります。

次のリセットシーケンスをお勧めします。

1. **nCPURESET**、**nDBGRESET**、**nNEONRESET**（SIMD MPE が存在する場合）を適用します。
2. 9 **CLK** サイクル以上待機してから、他のクロックドメインごとに 1 クロック以上待機します。他のコンポーネントの説明書で要求されている場合は、さらに多くのクロック分を待機します。これを上回るクロックサイクル分を適用しても害はありません。クロックドメインごとに 15 サイクルを適用することによって最大限の冗長性が実現されます。
3. Cortex-A9 ユニプロセッサへの **CLK** クロック入力を停止します。データエンジンが存在する場合は、**NEONCLKOFF** を使用します。*「構成信号」*（ページ A-5）を参照して下さい。
4. 実装に応じて、約 10 サイクルに相当する時間だけ待機します。これによって、クロックとリセットのツリーレイテンシが補償されます。
5. すべてのリセットを解放します。
6. 再度、クロックとリセットのツリーレイテンシを補償するために、さらに 10 サイクルに相当する時間だけ待機します。
7. クロックを再起動します。

ソフトウェアリセット

プロセッサまたはウォームリセットでは、デバッグロジックを除く Cortex-A9 プロセッサの大部分が初期化されます。プロセッサリセットでは、ブレークポイントとウォッチポイントが保持されます。通常、プロセッサリセットは、一定期間稼動していたシステムのリセットに使用されます。*「パワーオン リセット」*で説明したリセットシーケンスと同じものを使用します。唯一の違いは、シーケンス中に **nDBGRESET** を HIGH に保持して、デバッグレジスタ内のすべての値が保持されることを保証する必要がある点です。

プロセッサリセット

プロセッサまたはウォームリセットでは、デバッグロジックを除く Cortex-A9 プロセッサの大部分が初期化されます。プロセッサリセットでは、ブレークポイントとウォッチポイントが保持されます。通常、プロセッサリセットは、一定期間稼動していたシステムのリセットに使用されます。ウォームリセットでは、**nCPURESET** と **nNEONRESET** を使用します。

MPE SIMD ロジックリセット

このリセットでは、MPE のすべての SIMD ロジックが初期化されます。このリセットは、MPE の SIMD 部分が電力オフ状態を終了したときに適用されることを想定したものです。

このリセットは、SIMD MPE ロジックが、他のプロセッサロジックから分離された専用の電力ドメインに実装されている構成にのみ適用されます。

MPE SIMD リセットについては、次のリセットシーケンスをお勧めします。

1. **nNEONRESET** を適用します。
2. 9 CLK サイクル以上待機します。. これを上回るクロックサイクルを適用しても害はありません。例えば、クロックドメインごとに 15 サイクルを適用することによって最大限の冗長性が実現されます。
3. 1'b1 の値で **NEONCLKOFF** をアサートします。
4. 実装に応じて、約 10 サイクルに相当する時間だけ待機します。これによって、クロックとリセットのツリーレイテンシが補償されます。
5. **nNEONRESET** を解放します。
6. 再度、クロックとリセットのツリーレイテンシを補償するために、さらに 10 サイクルに相当する時間だけ待機します。
7. **NEONCLKOFF** をアサート解除します。これによって、プロセッサの SIMD MPE 部分に存在するすべてのレジスタで、リセットシーケンスを終了するときに同じ CLK エッジが検出されることが保証されます。

nNEONRESET を使用して、Cortex-A9 プロセッサリセットと別に、MPE ロジックの SIMD 部分を制御します。このリセットは、MPE の SIMD 部分をリセット状態に保持し、MPE の SIMD 部分への電力供給のオン / オフが安全に行えるようにするために使用します。表 2-2 （ページ 2-10）を参照して下さい。

デバッグリセット

このリセットでは、ブレークポイントやウォッチポイントの値を含む、Cortex-A9 ユニプロセッサ内のデバッグロジックが初期化されます。

デバッグリセットを実行するには、**nDBGRESET** 信号を数 CLK サイクルの間だけ LOW にアサートする必要があります。

2.3.3 動的高水準クロックゲート

次に示すセクションでは、動的高水準クロックゲートについて説明します。

- *「ゲートされるブロック」*（ページ 2-9）
- *「電力制御レジスタ」*（ページ 2-9）
- *「max_clk レイテンシビットの影響」*（ページ 2-9）
- *「動的高水準クロックゲート動作」*（ページ 2-9）

ゲートされるブロック

Cortex-A9 プロセッサ、または Cortex-A9 MPCore 設計の各プロセッサは、次の動的高水準クロックゲートをサポートします。

- 整数コア
- システム制御ブロック
- データエンジン（実装されている場合）

電力制御レジスタ

電力制御レジスタは、動的高水準クロックゲートを制御します。このレジスタには、これらのブロックに共通のフィールドが含まれています。

- クロックゲートのイネーブルビット
- max_clk レイテンシビット

「電力制御レジスタ」（ページ 4-36）を参照して下さい。

max_clk レイテンシビットの影響

max_clk レイテンシビットによって、これらのブロックのいずれかでクロックが停止されてから、新しいアクティブ信号が受信できるまでの遅延時間が決定されます。

max_clk レイテンシで決定された値が実際の遅延より短い場合、クロックが停止されたブロックは、クロックが供給されなくてもアクティブ信号を受信できます。これによって、デバイスが誤動作する可能性があります。

max_clk レイテンシで決定された値が実際の遅延より長い場合、マスタブロックは、クロックが停止されたブロックへの信号を送信する前に、数サイクルだけ余分に待機を行います。これによって、パフォーマンスにいくらかの影響が発生する可能性があります。

値が正しく設定されていれば、クロックが停止されたブロックで、ウェークアップの最初のクロックエッジでアクティブ信号が受信されます。これによって、最適のパフォーマンスが得られます。

動的高水準クロックゲート動作

動的高水準クロックゲートが可能であれば、次の場合に整数コアのクロックが停止されます。

- 整数コアが空で、ラインフィルを引き起こす命令ミスが発生した場合
- 整数コアが空で、命令 TLB ミスが発生した場合
- 整数コアがいっぱいで、ラインフィルを引き起こすデータミスが発生した場合
- 整数コアがいっぱいで、ラインフィルバッファがビジーのためにデータストアが停止された場合

動的クロックゲートが可能であれば、次の場合にシステム制御ブロックのクロックが停止されます。

- 実行中のシステム制御コプロセッサ命令が存在しない場合
- システム制御コプロセッサ命令がパイプラインに存在しない場合
- パフォーマンスイベントが不可能な場合
- デバッグが不可能な場合

動的クロックゲートが可能なとき、データエンジン命令がデータエンジンにもパイプラインにも存在しない場合は、データエンジンのクロックが停止されます。

2.4 電力管理

プロセッサは、動的と静的の両方の電力消費を制御するための機構を提供します。静的電力制御は実装固有です。説明については、次に示すセクションを参照して下さい。

- エネルギー効率機能
- Cortex-A9 プロセッサの電力制御

2.4.1 エネルギー効率機能

エネルギー効率を向上させる Cortex-A9 プロセッサの機能には、次のものが含まれます。

- 正確な分岐予測と復帰予測により、不正確な命令フェッチとデコード処理の回数を減らします。
- 物理的にアドレス指定されたキャッシュの使用により、キャッシュのフラッシュおよびリフィルの回数を減らし、システムの消費電力を抑えます。
- マイクロ TLB の使用により、サイクルごとに変換および保護ルックアップで消費される電力を減らします。
- キャッシュがシーケンシャルアクセス情報を使用することで、タグ RAM へのアクセス回数と、不要なデータ RAM へのアクセス回数を減らします。
- 64 バイト未満の命令ループの多くが、追加の命令キャッシュアクセスなしで完了することで、電力消費が抑えられます。

2.4.2 Cortex-A9 プロセッサの電力制御

各種の電力ドメインの実装を容易にするため、Cortex-A9 プロセッサの周辺にはレベルシフタとクランプのプレースホルダが挿入されています。

Cortex-A9 プロセッサには、次の電力ドメインを含めることができます。

- Cortex-A9 プロセッサロジック用の電力ドメイン
- Cortex-A9 プロセッサ MPE 用の電力ドメイン
- Cortex-A9 プロセッサ RAM 用の電力ドメイン

電力モードを、表 2-2 に示します。

表 2-2 Cortex-A9 プロセッサの電力モード

モード	Cortex-A9 プロセッサの RAM アレイ	Cortex-A9 プロセッサロジック	Cortex-A9 データエンジン	コメント
フル実行モード	電力オン	電力オン	電力オン	-
		クロックが供給されている	クロックが供給されている	
MPE が非稼働の実行モード	電力オン	電力オン	電力オン	MPE を非稼働にする方法については、*「コプロセッサアクセス制御レジスタ」*（ページ 4-20）を参照して下さい。
		クロックが供給されている	クロックなし	
MPE が電力オフの実行モード	電力オン	電力オン	電力オフ	MPE は、別の電力ドメインに実装でき、個別に電力オフにすることもできます。
		クロックが供給されている		

表 2-2 Cortex-A9 プロセッサの電力モード （続き）

モード	Cortex-A9 プロセッサの RAM アレイ	Cortex-A9 プロセッサロジック	Cortex-A9 データエンジン	コメント
スタンバイ	電力オン	電力オン	電力オン	スタンバイモード、*「スタンバイモード」* を参照
		ウェークアップ ロジックのみにクロックが供給されている	クロックが非稼働か、電力オフ	
休眠	状態 / 電圧を保持	電力オフ	電力オフ	ウェークアップには外部ウェークアップ イベントが必要
シャットダウン	電力オフ	電力オフ	電力オフ	ウェークアップには外部ウェークアップ イベントが必要

休眠モードまたはシャットダウンモードの開始は、外部電力コントローラで制御する必要があります。

実行モード

実行モードは、Cortex-A9 プロセッサのすべての機能が使用できる通常の動作モードです。

スタンバイモード

WFI および WFE スタンバイモードでは、プロセッサのクロックのほとんどが非稼働になりますが、ロジックへ引き続き電力が供給されます。これにより、静的な漏洩電流と、デバイスをウェークアップするためのわずかなクロック電力オーバヘッドのみに消費電力が削減されます。

WFI スタンバイモードは、`WFI` 命令の実行によって開始されます。

WFI スタンバイモードから実行モードへの移行は、次のいずれかの場合に発生します。

- マスクされているかどうかにかかわらず、割り込みが発生した場合
- CPSR.A ビットの値にかかわらず、非同期データアボートが発生した場合。ウェークアップ イベントの保留中は、プロセッサは低消費電力モードに移行できません。
- デバッグが可能かどうかにかかわらず、デバッグ要求が発生した場合
- リセット時

WFE スタンバイモードは、WFE 命令の実行によって開始されます。

WFE スタンバイモードから実行モードへの移行は、次のいずれかの場合に発生します。

- マスクされていない割り込みが発生した場合
- デバッグが可能かどうかにかかわらず、デバッグ要求が発生した場合
- 同じプロセッサ上で以前に発生した例外復帰
- リセット時
- **EVENTI** 入力信号がアサートされた場合

デバッグ要求は、Cortex-A9 プロセッサの **EDBGRQ** ピンを使用して、または、APB デバッグポート経由で Cortex-A9 プロセッサに発行されたデバッグホールト命令から、外部で生成されたデバッグ要求によって生成されます。

デバッグチャネルは、`WFI` を通してアクティブのままです。

休眠モード

休眠モードでは、キャッシュへの電力供給は変わらず、キャッシュの状態を保持しながら、Cortex-A9 プロセッサへの電力供給を停止できます。

休眠モード中に電力オンに保持する必要のある RAM ブロックは次のとおりです。

- キャッシュに関連付けられているすべてのデータ RAM
- キャッシュに関連付けられているすべてのタグ RAM
- 外部 RAM

電力オン状態に保持する RAM ブロックは、別の電力ドメインに実装する必要があります。

休眠モードに移行する前に、Cortex-A9 プロセッサの状態（休眠モード中も電力オンに保持される RAM の内容は除く）を外部メモリに保存する必要があります。これらの状態保存操作では、次の処理を必ず行う必要があります。

- すべての ARM レジスタ（CPSR レジスタおよび SPSR レジスタを含む）を保存する。
- すべてのシステムレジスタを保存する。
- すべてのデバッグ関連の状態を保存する。
- すべての状態保存が完了したことを保証するため、データ同期バリア命令を実行する。
- その後で、Cortex-A9 プロセッサが **STANDBYWFI** を使用して電力コントローラと通信し、`WFI` 命令の実行により、休眠モードに移行する準備ができたことを通知します。詳細については、*「電力管理コントローラとの通信」*（ページ 2-13）を参照して下さい。
- 電力をオフにする前に、外部の電力制御機構を使用して、Cortex-A9 プロセッサへのリセット信号をアサートする必要があります。

外部の電力コントローラが、休眠状態から実行状態への移行をトリガします。外部の電力コントローラは、電力が復元される前に Cortex-A9 プロセッサへのリセットをアサートする必要があります。電力が復元された後で、Cortex-A9 プロセッサは、リセット状態を終了して、保存された状態を復元する必要があるかどうかを判断できます。

シャットダウンモード

シャットダウンモードでは、デバイス全体への電力がオフになります。キャッシュを含めて、すべての状態をソフトウェアによって外部に保存する必要があります。この状態保存は割り込みが不可能な状態で実行され、データ同期バリア操作で終了します。その後で、Cortex-A9 プロセッサは、休眠モードへの移行時と同様に、デバイスの電力をオフにする準備ができたことを電力コントローラに通知します。プロセッサは、リセットをアサートすることによって実行状態に戻ります。

注

リセットを実行する前にプロセッサに電力をオンにする必要があります。

電力管理コントローラとの通信

Cortex-A9 プロセッサと外部の電力管理コントローラとの通信は、スタンバイ信号、Cortex-A9 の入力クランプ信号、および **DBGNOPWRDWN** を使用して実行できます。

スタンバイ信号

これらの信号は、外部の電力管理コントローラを制御します。

STANDBYWFI 信号は、Cortex-A9 プロセッサが電力オフモードに入る準備ができたことを示します。*「スタンバイ信号とイベント待ち信号」*（ページ A-6）を参照して下さい。

Cortex-A9 の入力信号

外部の電力管理コントローラは、**NEONCLAMP** と **CPURAMCLAMP** を使用して、電力がオフになる前に、Cortex-A9 の電力ドメインを相互に分離します。これらの信号は、Cortex-A9 プロセッサに電力ドメインクランプが実装されている場合にのみ意味を持ちます。*「電力管理信号」*（ページ A-7）を参照して下さい。

DBGNOPWRDWN

DBGNOPWRDWN は、システム電力コントローラに接続され、エミュレートモードで動作する要求と解釈されます。このモードでは、ソフトウェアまたはハードウェアのハンドシェークによって要求されても、Cortex-A9 プロセッサと PTM への電力供給は実際には停止されません。*「その他のデバッグインタフェース信号」*（ページ A-23）を参照して下さい。

2.4.3 電力ドメイン

Cortex-A9 ユニプロセッサには、Cortex-A9 ロジックと RAM アレイの間、または Cortex-A9 ロジックと NEON SIMD ロジックの間（NEON が存在する場合）にオプションのプレースホルダが含まれているため、これらの部品を別々の電圧ドメインに実装できます。

2.4.4 Cortex-A9 の電圧ドメイン

Cortex-A9 プロセッサには、次の電力ドメインを含めることができます。

- Cortex-A9 プロセッサロジック セル用の電力ドメイン
- Cortex-A9 プロセッサ データエンジン用の電力ドメイン
- Cortex-A9 プロセッサ RAM 用の電力ドメイン

電力ドメインを、図 2-4 （ページ 2-14）に示します。

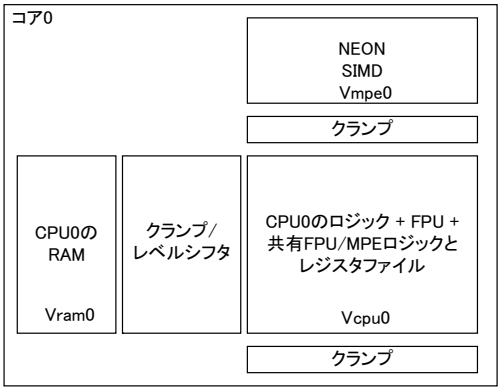


図 2-4 Cortex-A9 r2 設計の電力ドメイン

FPU は CPU 電力ドメインの一部です。FPU クロックは CPU クロックに基づきます。静的および動的の高水準クロックゲートが存在します。

NEON SIMD データパスとロジックは、専用のクロックとリセット信号を持つ、別の電力ドメインに存在します。静的および動的の高水準クロックゲートが存在します。

NEON が存在する場合は、SIMD 部分に電力やクロックを供給せずに、FPU（非 SIMD）コードを実行できます。

2.5 使用制限

ここでは、メモリの整合性について説明します。

Cortex-A9 プロセッサ内のメモリコヒーレンシは、ウィークリーオーダ メモリ整合性モデルに従って維持されます。

注

共有可能属性がライトバック、ノーマルメモリ以外のメモリ領域に適用されている場合、その領域に保存されているデータはキャッシュ不可として扱われます。

第 3 章
プログラマモデル

本章では、プロセッサのレジスタとプログラミング方法について説明します。本章は次のセクションから構成されています。

3.1 プログラマモデルについて

Cortex-A9 プロセッサは ARMv7-A アーキテクチャを実装しています。

ARMv7-A アーキテクチャの詳細については、*『ARM アーキテクチャ リファレンスマニュアル』*を参照して下さい。

3.2 ThumbEE アーキテクチャ

Thumb 実行環境 (ThumbEE) 拡張機能は、動的に生成されるコードのターゲットとして設計された、Thumb 命令セットのバリアントです。

ThumbEE 命令セットについては、*『ARM アーキテクチャ リファレンスマニュアル』*を参照して下さい。

3.3 アドバンスト SIMD アーキテクチャ

アドバンスト SIMD アーキテクチャ拡張機能は、主にオーディオ、ビデオ、3 次元グラフィック、画像、音声の処理を対象とした命令が追加された、メディアおよび信号処理アーキテクチャです。

注

アドバンスト SIMD アーキテクチャ拡張機能、それに関連した実装、および対応ソフトウェアをまとめて、NEON MPE と呼びます。

NEON MPE には、*アドバンスト* SIMD 命令と ARM VFPv3 命令の両方が含まれます。アドバンスト SIMD 命令と VFP 命令はすべて、ARM 状態と Thumb 状態の両方で使用できます。

詳細については、*『ARM アーキテクチャ リファレンスマニュアル』*を参照して下さい。

実装固有の情報については、*『Cortex-A9 NEON メディア処理エンジン テクニカルリファレンス マニュアル』*を参照して下さい。

3.4 セキュリティ拡張機能アーキテクチャ

セキュリティ拡張機能の目的は、セキュアなソフトウェア環境の構築を可能にすることです。ここでは、次の操作について説明します。

- *「システムブート シーケンス」*

詳細については、*『ARM アーキテクチャ リファレンスマニュアル』*を参照して下さい。

3.4.1 システムブート シーケンス

注意

セキュリティ拡張機能を使用すれば、プロセッサ周辺の適切なシステム設計によって、よりセキュアな実行のための隔離されたソフトウェア環境の構築が可能になります。このテクノロジではプロセッサがハードウェア攻撃から保護されないため、リセット処理コードを含むハードウェアが適切に保護されていることを確認する必要があります。

プロセッサは常に、NS ビットが 0 に設定された、セキュア状態の特権スーパバイザモードでブートします。このため、セキュリティ拡張機能の使用を試みないコードは、常にセキュア状態で実行されます。ソフトウェアでセキュア状態と非セキュア状態の両方が使用される場合は、複雑なオペレーティングシステムなどの信頼性の低いソフトウェアは非セキュア状態で、より信頼性の高いソフトウェアはセキュア状態で実行されます。

次のシーケンスは、セキュリティ拡張機能の標準的な使用を想定したものです。

1. セキュア状態でリセットを終了します。
2. メモリとペリフェラルのセキュリティ状態を構成します。メモリとペリフェラルの一部は、セキュア状態で実行されているソフトウェアからのみアクセス可能です。
3. セキュア オペレーティングシステムを初期化します。必要な操作はオペレーティングシステムによって異なりますが、一般的に、キャッシュ、MMU、例外ベクタ、スタックの初期化が含まれます。
4. セキュア オペレーティングシステムと非セキュア オペレーティングシステムとの実行を切り替える例外を処理する、セキュアモニタ ソフトウェアを初期化します。
5. （オプション）セキュア状態環境の一部の要素を、以後の構成で変更されないように固定します。
6. セキュアモニタ ソフトウェアで SMC 命令を使用して、制御を非セキュア オペレーティングシステムに渡し、非セキュア オペレーティングシステムの初期化を可能にします。必要な操作はオペレーティングシステムによって異なりますが、一般的にキャッシュ、MMU、例外ベクタ、スタックの初期化が含まれます。

セキュアソフトウェア全体のセキュリティは、システム設計とセキュアソフトウェア自体に依存します。

3.5 マルチプロセッシング拡張機能

マルチプロセッシング拡張機能は、マルチプロセッシング機能を拡張する機能セットです。詳細については、*『ARM アーキテクチャ リファレンスマニュアル』*を参照して下さい。

3.6 Jazelle 拡張機能

Cortex-A9 プロセッサは、Jazelle 拡張機能をハードウェアでサポートしています。このプロセッサは、ほとんどのバイトコードの実行を高速化します。一部のバイトコードは、ソフトウェアルーチンにより実行されます。

詳細については、*『ARM アーキテクチャ リファレンスマニュアル』*を参照して下さい。

第 5 章 *Jazelle DBX レジスタ*も参照して下さい。

3.7 メモリモデル

Cortex-A9 プロセッサは、メモリを、0 から昇順に番号が付けられたバイトの配列として参照します。例えば、バイト 0 ～ 3 は最初にストアされるワードを、バイト 4 ～ 7 は 2 番目にストアされるワードを保持します。プロセッサは、リトルエンディアン形式とビッグエンディアン形式のいずれかでワードをメモリにストアできます。

命令は常にリトルエンディアンとして扱われます。

注

ARMv7 は、BE-32 メモリモデルをサポートしていません。

3.8 Cortex-A9 プロセッサのアドレス

Cortex-A9 では、VA と MVA は同一です。

Cortex-A9 プロセッサが非セキュア状態で動作している場合は、非セキュアバージョンの変換テーブル ベースレジスタを使用して、変換テーブルルックアップが実行されます。この状況では、VA は非セキュア PA にしか変換できません。プロセッサがセキュア状態の場合は、セキュアバージョンの変換テーブル ベースレジスタを使用して変換テーブルルックアップが実行されます。この状況では、VA のセキュリティ状態は、そのアドレスの変換テーブル記述子の NS ビットによって決定されます。

プロセッサシステム内のアドレスタイプを、表 3-1 に示します。

表 3-1 プロセッサシステム内のアドレスタイプ

プロセッサ	キャッシュ	変換ルックアサイドバッファ	AXI バス
データ VA	データキャッシュは、*物理インデクス物理タグ付き* (PIPT) です。	仮想アドレスを 物理アドレスに変換する	物理 アドレス
命令 VA	命令キャッシュは、*仮想インデクス物理タグ付き* (VIPT) です。		

次に示すのは、Cortex-A9 プロセッサから命令が要求されたときに発生するアドレス操作の例です。

1. Cortex-A9 プロセッサは、そのときの状態に応じて、セキュア VA または非セキュア VA として、命令の VA を発行します。
2. 命令キャッシュには、VA の低位ビットによってインデクスが付けられます。TLB は、キャッシュルックアップと並行して変換を実行します。プロセッサがセキュア状態の場合は、変換にセキュア記述子が使用されます。そうでない場合は、非セキュア記述子が使用されます。
3. TLB が VA に対して実行した保護チェックでアボートが発生せず、PA タグが命令キャッシュ内に存在する場合は、命令データがプロセッサに返されます。
4. キャッシュミスが発生した場合は、PA が AXI バスインタフェースに渡され、外部アクセスが実行されます。コアが非セキュア状態にあるとき、外部アクセスは常に非セキュアです。セキュア状態では、外部アクセスは、選択された記述子の NS 属性値に応じて、セキュアまたは非セキュアになります。また、セキュア状態では、レベル 1 記述子が NS としてマークされていても、レベル 1 とレベル 2 のテーブルウォーク アクセスはセキュアとしてマークされます。

   注

   レベル 1 エントリが非セキュアとしてマークされていても、セキュアレベル 2 ルックアップはセキュアになります。

第 4 章
システム制御

本章では、システム制御レジスタの構造、動作、および使用方法について説明します。
本章は次のセクションから構成されています。

- *「システム制御について」*（ページ 4-2）
- *「レジスタの概要」*（ページ 4-3）
- *「レジスタの説明」*（ページ 4-8）

4.1 システム制御について

システム制御コプロセッサ CP15 は、プロセッサに実装されている機能に関するステータス情報を制御および提供します。システム制御コプロセッサの主な機能は次のとおりです。

- システム全体の制御と構成
- MMU の構成と管理
- キャッシュの構成と管理
- システムパフォーマンスの監視

4.1.1 非推奨レジスタ

ARMv7-A では、次に示す機能については、等価な機能が命令セットに存在します。

- 命令同期バリア
- データ同期バリア
- データ メモリバリア
- 割り込み待ち

これらのレジスタの使用はオプションで、非推奨です。

また、ARM アーキテクチャ v7 では、高速コンテキストスイッチ拡張機能が非推奨で、Cortex-A9 プロセッサには実装されていません。

4.2 レジスタの概要

すべての CP15 システム制御レジスタを、レジスタへのアクセスに使用されるパラメータ順に、表 4-1 に示します。

- 1 次 CP15 コプロセッサレジスタ、CRn
- opcode_1 の値
- 2 次 CP15 コプロセッサレジスタ、CRm
- opcode_2 の値

表には、レジスタの CRn グループごとの属性概要への参照が含まれています。

『ARM アーキテクチャ リファレンスマニュアル』 に記載されているレジスタについては、*『ARM アーキテクチャ リファレンスマニュアル、ARMv7-A および ARMv7-M エディション』*, http://silver.arm.com/browse/AR570 を参照して下さい。

表 4-1 CP15 システム制御コプロセッサレジスタの概要

CRn	名前	レジスタの説明
c0	-	*「CP15 c0 レジスタの概要」*（ページ 4-8）
	MIDR	メイン ID レジスタ、*『ARM アーキテクチャ リファレンスマニュアル』* を参照
	CTR	キャッシュタイプ レジスタ、*『ARM アーキテクチャ リファレンスマニュアル』* を参照
	TCMTR	TCM タイプレジスタ、*『ARM アーキテクチャ リファレンスマニュアル』* を参照
	TLBTR	*「TLB タイプレジスタ」*（ページ 4-9）
	MPIDR	*「マルチプロセッサ類似性レジスタ」*（ページ 4-10）
	ID_PFR0	プロセッサ機能レジスタ 0、*『ARM アーキテクチャ リファレンスマニュアル』* を参照
	ID_PFR1	プロセッサ機能レジスタ 1、*『ARM アーキテクチャ リファレンスマニュアル』* を参照
	ID_DFR0	デバッグ機能レジスタ、*『ARM アーキテクチャ リファレンスマニュアル』* を参照
	ID_MMFR0	メモリモデル機能レジスタ 0、*『ARM アーキテクチャ リファレンスマニュアル』* を参照
	ID_MMFR1	メモリモデル機能レジスタ 1、*『ARM アーキテクチャ リファレンスマニュアル』* を参照
	ID_MMFR2	メモリモデル機能レジスタ 2、*『ARM アーキテクチャ リファレンスマニュアル』* を参照
	ID_MMFR3	メモリモデル機能レジスタ 3、*『ARM アーキテクチャ リファレンスマニュアル』* を参照
	ID_ISAR0	命令セット属性レジスタ 0、*『ARM アーキテクチャ リファレンスマニュアル』* を参照
c0	ID_ISAR1	命令セット属性レジスタ 1、*『ARM アーキテクチャ リファレンスマニュアル』* を参照
	ID_ISAR2	命令セット属性レジスタ 2、*『ARM アーキテクチャ リファレンスマニュアル』* を参照
	ID_ISAR3	命令セット属性レジスタ 3、*『ARM アーキテクチャ リファレンスマニュアル』* を参照
	ID_ISAR4	命令セット属性レジスタ 4、*『ARM アーキテクチャ リファレンスマニュアル』* を参照
	CCSIDR	*「キャッシュサイズ識別レジスタ」*（ページ 4-11）
	CLIDR	*「キャッシュレベル ID レジスタ」*（ページ 4-13）
	AIDR	*「補助 ID レジスタ」*（ページ 4-13）
	CSSELR	*「キャッシュサイズ選択レジスタ」*（ページ 4-14）

表 4-1 CP15 システム制御コプロセッサレジスタの概要 （続き）

CRn	名前	レジスタの説明
c1	-	*「CP15 c1 レジスタの概要」*（ページ 4-15）
	SCTLR	*「システム制御レジスタ」*（ページ 4-15）
	ACTLR	*「補助制御レジスタ」*（ページ 4-18）
	CPACR	*「コプロセッサアクセス制御レジスタ」*（ページ 4-20）
	SCR	セキュア構成レジスタ、*『ARM アーキテクチャ リファレンスマニュアル』*を参照
	SDER	*「セキュアデバッグ イネーブルレジスタ」*（ページ 4-22）
	NSACR	*「非セキュアアクセス制御レジスタ」*（ページ 4-23）
	VCR	*「仮想化制御レジスタ」*（ページ 4-24）
c2	-	*「CP15 c2 レジスタの概要」*（ページ 4-25）
	TTBR0	変換テーブル ベースレジスタ 0、*『ARM アーキテクチャ リファレンスマニュアル』*を参照
	TTBR1	変換テーブル ベースレジスタ 1、*『ARM アーキテクチャ リファレンスマニュアル』*を参照
	TTBCR	変換テーブルベース制御レジスタ、*『ARM アーキテクチャ リファレンスマニュアル』*を参照
c3	-	*「CP15 c3 レジスタの概要」*（ページ 4-26）
	DACR	ドメインアクセス制御レジスタ、*『ARM アーキテクチャ リファレンスマニュアル』*を参照
c4	-	*「CP15 c4、未使用」*（ページ 4-26）
c5	-	*「CP15 c5 レジスタの概要」*（ページ 4-26）
	DFSR	データフォールト ステータスレジスタ、*『ARM アーキテクチャ リファレンスマニュアル』*を参照
	IFSR	命令フォールト ステータスレジスタ、*『ARM アーキテクチャ リファレンスマニュアル』*を参照
	ADFSR	補助データフォールト ステータスレジスタ、*『ARM アーキテクチャ リファレンスマニュアル』*を参照
	AIFSR	補助命令フォールト ステータスレジスタ、*『ARM アーキテクチャ リファレンスマニュアル』*を参照
c6	-	*「CP15 c6 レジスタの概要」*（ページ 4-26）
	DFAR	データフォールト アドレスレジスタ、*『ARM アーキテクチャ リファレンスマニュアル』*を参照
	IFAR	命令フォールト アドレスレジスタ、*『ARM アーキテクチャ リファレンスマニュアル』*を参照

表 4-1 CP15 システム制御コプロセッサレジスタの概要 （続き）

CRn	名前	レジスタの説明
c7	-	*「CP15 c7 レジスタの概要」*（ページ 4-27）
	ICIALLUIS	*『ARM アーキテクチャ リファレンスマニュアル』*を参照
	BPIALLIS	
	PAR	
	ICIALLU	
	ICIMVAU	
	BPIALL	
	DCIMVAC	
	DCISW	
	V2PCWPR	
	DCCVAC	
	DCCSW	
	DCCVAU	
	DCCIMVAC	
	DCCISW	
c8	-	*「CP15 c8 レジスタの概要」*（ページ 4-28）
	TLBIALLIS	*『ARM アーキテクチャ リファレンスマニュアル』*を参照
	TLBIMVAIS	
	TLBIASIDIS	
	TLBIMVAAIS	
	TLBIALL	
	TLBIMVA	
	TLBIASID	
	TLBIMVAA	

表 4-1 CP15 システム制御コプロセッサレジスタの概要 （続き）

CRn	名前	レジスタの説明
c9	-	*「CP15 c9 レジスタの概要」*（ページ 4-28）
	PMCR	パフォーマンスモニタ制御レジスタ、*『ARM アーキテクチャ リファレンスマニュアル』*を参照
	PMCNTENSET	カウントイネーブル セットレジスタ、*『ARM アーキテクチャ リファレンスマニュアル』*を参照
	PMCNTENCLR	カウントイネーブル クリアレジスタ、*『ARM アーキテクチャ リファレンスマニュアル』*を参照
	PMOVSR	オーバフローフラグ ステータスレジスタ、*『ARM アーキテクチャ リファレンスマニュアル』*を参照
	PMSWINC	ソフトウェアインクリメント レジスタ、*『ARM アーキテクチャ リファレンスマニュアル』*を参照
	PMSELR	イベントカウンタ選択レジスタ、*『ARM アーキテクチャ リファレンスマニュアル』*を参照
	PMCCNTR	サイクルカウント レジスタ、*『ARM アーキテクチャ リファレンスマニュアル』*を参照
	PMXEVTYPER	イベント選択レジスタ、*『ARM アーキテクチャ リファレンスマニュアル』*を参照
	PMXEVCNTR	パフォーマンスモニタ カウントレジスタ、*『ARM アーキテクチャ リファレンスマニュアル』*を参照
	PMUSERENR	ユーザイネーブル レジスタ、*『ARM アーキテクチャ リファレンスマニュアル』*を参照
	PMINTENSET	割り込みイネーブルセット レジスタ、*『ARM アーキテクチャ リファレンスマニュアル』*を参照
	PMINTENCLR	割り込みイネーブルクリア レジスタ、*『ARM アーキテクチャ リファレンスマニュアル』*を参照
c10	-	*「CP15 c10 レジスタの概要」*（ページ 4-29）
	-	*「TLB ロックダウンレジスタ」*（ページ 4-29）
	PRRR	1 次領域再マップレジスタ、*『ARM アーキテクチャ リファレンスマニュアル』*を参照
	NRRR	ノーマルメモリ再マップレジスタ、*『ARM アーキテクチャ リファレンスマニュアル』*を参照
c11	-	*「CP15 c11 レジスタの概要」*（ページ 4-30）
	PLEIDR	*「PLE ID レジスタ」*（ページ 4-30）
	PLEASR	*「PLE 動作ステータスレジスタ」*（ページ 4-31）
	PLEFSR	*「PLE FIFO ステータスレジスタ」*（ページ 4-32）
	PLEUAR	*「プリロードエンジン ユーザアクセス許可レジスタ」*（ページ 4-32）
	PLEPCR	*「プリロードエンジン パラメータ制御レジスタ」*（ページ 4-33）
c12	-	*「CP15 c12 レジスタの概要」*（ページ 4-34）
	VBAR	ベクタ ベースアドレス レジスタ、*『ARM アーキテクチャ リファレンスマニュアル』*を参照
	MVBAR	モニタベクタ ベースアドレス レジスタ、*『ARM アーキテクチャ リファレンスマニュアル』*を参照
	ISR	割り込みステータスレジスタ、*『ARM アーキテクチャ リファレンスマニュアル』*を参照
	-	*「仮想化割り込みレジスタ」*（ページ 4-34）
c13	-	*「CP15 c13 レジスタの概要」*（ページ 4-35）
	FCSEIDR	FCSE PID レジスタ、*『ARM アーキテクチャ リファレンスマニュアル』*を参照
	CONTEXTIDR	コンテキスト ID レジスタ、*『ARM アーキテクチャ リファレンスマニュアル』*を参照
	TPIDRURW	ユーザ読み出し / 書き込みソフトウェアスレッド レジスタ、*『ARM アーキテクチャ リファレンスマニュアル』*を参照
	TPIDRURO	ユーザ読み出し専用ソフトウェアスレッド レジスタ、*『ARM アーキテクチャ リファレンスマニュアル』*を参照

表 4-1 CP15 システム制御コプロセッサレジスタの概要 （続き）

CRn	名前	レジスタの説明
	TPIDRPRW	特権専用ソフトウェアスレッド レジスタ、*『ARM アーキテクチャ リファレンスマニュアル』*を参照
c14	-	*「CP15 c14、未使用」*（ページ 4-36）
c15	-	*「CP15 c15 レジスタの概要」*（ページ 4-36）
	-	*「電力制御レジスタ」*（ページ 4-36）
	-	*「NEON ビジーレジスタ」*（ページ 4-37）
	-	*「構成ベースアドレス レジスタ」*（ページ 4-38）
	-	*「TLB ロックダウン操作」*（ページ 4-39）

4.3 レジスタの説明

ここでは、レジスタへのアクセスに使用される 1 次 CP15 レジスタ番号 CR n でグループ分けされた、システム制御コプロセッサのレジスタ割り当てとリセット時の値の要約を、表に示します。

CRn グループごとのレジスタの要約では、次の表記が使用されます。

- CRn は、CP15 内部のレジスタ番号です。
- Op1 は、レジスタの Opcode_1 の値です。
- CRm は、操作対象のレジスタです。
- Op2 は、レジスタの Opcode_2 の値です。
- タイプは次のとおりです。
  - 読み出し専用 (RO)
  - 書き込み専用 (WO)
  - 読み出し / 書き込み (RW)
- リセット時の値は、レジスタのリセット時の値です。

「PLE の新規チャネルプログラム操作」（ページ 9-5）で説明されている新規チャネルのプログラム操作を除いて、すべてのシステム制御コプロセッサレジスタは 32 ビット幅です。予約レジスタは RAZ/WI です。

このセクションでは、*『ARM アーキテクチャ リファレンスマニュアル』* に記載されているレジスタに関する情報は繰り返しません。本章では、実装定義の制御コプロセッサレジスタについて説明します。

4.3.1 CP15 c0 レジスタの概要

CRn が c0 のときにアクセス可能なシステム制御レジスタを、表 4-2 に示します。

表 4-2 c0 システム制御レジスタ

Op1	CRm	Op2	名前	タイプ	リセット時の値	説明
0	c0	0	MIDR	RO	0x412FC090	r2p0 のメイン ID レジスタ
			MIDR	RO	0x412FC091	r2p1 のメイン ID レジスタ
			MIDR	RO	0x412FC092	r2p2 のメイン ID レジスタ
		1	CTR	RO	0x83338003	キャッシュタイプ レジスタ
		2	TCMTR	RO	0x00000000	TCM タイプレジスタ
		3	TLBTR[a]	RO	-	*「TLB タイプレジスタ」*（ページ 4-9）
		5	MPIDR	RO	-	*「マルチプロセッサ類似性レジスタ」*（ページ 4-10）
	c1	0	ID_PFR0	RO	0x00001231	プロセッサ機能レジスタ 0
		1	ID_PFR1	RO	0x00000011	プロセッサ機能レジスタ 1
		2	ID_DFR0	RO	0x00010444	デバッグ機能レジスタ 2
		4	ID_MMFR0	RO	0x00100103	メモリモデル機能レジスタ 0
		5	ID_MMFR1	RO	0x20000000	メモリモデル機能レジスタ 1
		6	ID_MMFR2	RO	0x01230000	メモリモデル機能レジスタ 2
		7	ID_MMFR3	RO	0x00102111	メモリモデル機能レジスタ 3

表 4-2 c0 システム制御レジスタ （続き）

Op1	CRm	Op2	名前	タイプ	リセット時の値	説明
	c2	0	ID_ISAR0	RO	0x00101111	命令セット属性レジスタ 0
		1	ID_ISAR1	RO	0x13112111	命令セット属性レジスタ 1
		2	ID_ISAR2	RO	0x21232041	命令セット属性レジスタ 2
		3	ID_ISAR3	RO	0x11112131	命令セット属性レジスタ 3
		4	ID_ISAR4	RO	0x00011142	命令セット属性レジスタ 4
1	c0	0	CCSIDR	RO	-	*「キャッシュサイズ識別レジスタ」*（ページ 4-11）
		1	CLIDR	RO	0x09000003	*「キャッシュレベル ID レジスタ」*（ページ 4-13）
		7	AIDR	RO	0x00000000	*「補助 ID レジスタ」*（ページ 4-13）
2	c0	0	CSSELR	RW	-	*「キャッシュサイズ選択レジスタ」*（ページ 4-14）

a. TLBSIZE によって異なります。*「TLB タイプレジスタ」*を参照して下さい。

4.3.2 TLB タイプレジスタ

TLBTR の特徴は次のとおりです。

目的 TLB のロック可能エントリの数を返します。

使用制限 TLBTR の使用制限は次のとおりです。

- セキュア状態と非セキュア状態とで共通です。
- 特権モードでのみアクセス可能です。

構成 すべての構成で使用可能です。

属性 レジスタの概要については、表 4-2 （ページ 4-8）を参照して下さい。

TLBTR のビット割り当てを、図 4-1 に示します。

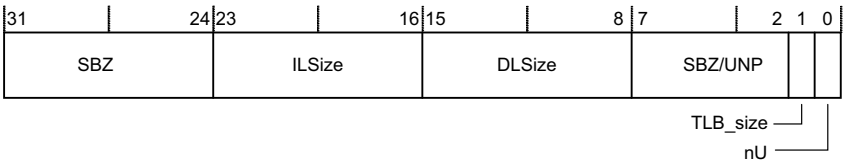


図 4-1 TLBTR のビット割り当て

TLBTR のビット割り当てを、表 4-3 に示します。

表 4-3 TLBTR のビット割り当て

ビット	名前	機能
[31:24]	SBZ	-
[23:16]	ILsize	命令 TLB のロック可能エントリの数を示します。Cortex-A9 プロセッサでは 0 です。
[15:8]	DLsize	統一 TLB またはデータ TLB のロック可能エントリの数を示します。Cortex-A9 プロセッサでは 4 です。

表 4-3 TLBTR のビット割り当て （続き）

ビット	名前	機能
[7:2]	SBZ または UNP	-
[1]	TLB_size	0 = TLB のエントリ数は 64 です。 1 = TLB のエントリ数は 128 です。
[0]	nU	TLB が統一されているか (0)、命令 TLB とデータ TLB が分離されているかを示します。 0 = Cortex-A9 プロセッサには統一 TLB が搭載されています。

TLBTR にアクセスするには、次の命令を使用します。

```
MRC p15,0,<Rd>,c0,c0,3; returns TLB details
```

4.3.3 マルチプロセッサ類似性レジスタ

MPIDR の特徴は次のとおりです。

目的 次のことを識別します。
- プロセッサが Cortex-A9 MPCore 実装の一部かどうか
- Cortex-A9 MPCore プロセッサ内部の Cortex-A9 プロセッサアクセス
- マルチプロセッサ クラスタシステム内のターゲット Cortex-A9 プロセッサ

使用制限 MPIDR の使用制限は次のとおりです。
- 特権モードでのみアクセス可能です。
- セキュア状態と非セキュア状態とで共通です。

構成 すべての構成で使用可能です。U ビット（ビット [30]）の値は、構成がマルチプロセッサ構成かユニプロセッサ構成かを示します。

属性 レジスタの概要については、表 4-2 （ページ 4-8）を参照して下さい。

MPIDR のビット割り当てを、図 4-2 に示します。

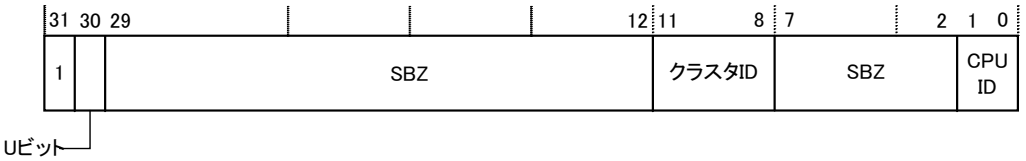


図 4-2 MPIDR のビット割り当て

MPIDR のビット割り当てを、表 4-4 に示します。

表 4-4 MPIDR のビット割り当て

ビット	名前	機能
[31]	-	レジスタが新しいマルチプロセッサ形式を使用していることを示します。このビットは常に 1 です。
[30]	U ビット	マルチプロセッシング拡張機能。 0 = プロセッサは MPCore クラスタの一部です。 1 = プロセッサはユニプロセッサです。
[29:12]	-	SBZ
[11:8]	クラスタ ID	**CLUSTERID** 構成ピンで読み取られた値[a]。複数の Cortex-A9 MPCore プロセッサが存在するシステムで、それぞれの Cortex-A9 MPCore プロセッサを識別します。ユニプロセッサ構成の場合は SBZ です。
[7:2]	-	SBZ
[1:0]	CPU ID	Cortex-A9 MPCore 構成内の CPU 番号を示します。 • 0x0 プロセッサは CPU0 です。 • 0x1 プロセッサは CPU1 です。 • 0x2 プロセッサは CPU2 です。 • 0x3 プロセッサは CPU3 です。 ユニプロセッサバージョンでは、この値が 0x0 に固定されます。

a. ユニプロセッサ実装には、**CLUSTERID** ピンは存在しません。

MPIDR にアクセスするには、次の命令を使用します。

```
MRC p15,0,<Rd>,c0,c0,5; read Multiprocessor ID register
```

4.3.4 キャッシュサイズ識別レジスタ

CCSIDR の特徴は次のとおりです。

目的 CSSELR で選択されたキャッシュのアーキテクチャに関する情報を提供します。

使用制限 CCSIDR の使用制限は次のとおりです。
- 特権モードでのみアクセス可能です。
- セキュア状態と非セキュア状態とで共通です。

構成 すべての構成で使用可能です。

属性 レジスタの概要については、表 4-2 （ページ 4-8）を参照して下さい。

CCSIDR のビット割り当てを、図 4-3 に示します。

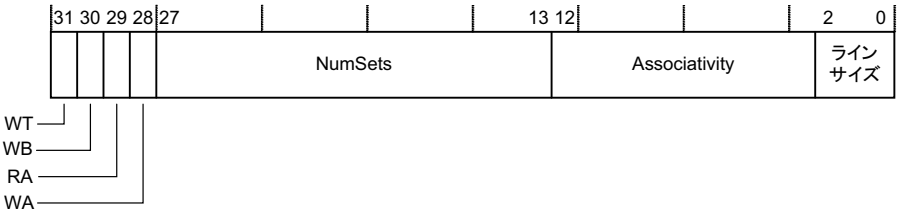


図 4-3 CCSIDR のビット割り当て

CSSIDR のビット割り当てを、表 4-5 に示します。

表 4-5 CCSIDR のビット割り当て

ビット	名前	機能
[31]	WT	ライトスルーがサポートされているかどうかを示します。 0 = ライトスルーがサポートされていません。 1 = ライトスルーがサポートされています。
[30]	WB	ライトバックがサポートされているかどうかを示します。 0 = ライトバックがサポートされていません。 1 = ライトバックがサポートされています。
[29]	RA	読み出し割り当てがサポートされているかどうかを示します。 0 = 読み出し割り当てがサポートされていません。 1 = 読み出し割り当てがサポートされています。
[28]	WA	書き込み割り当てがサポートされているかどうかを示します。 0 = 書き込み割り当てがサポートされていません。 1 = 書き込み割り当てがサポートされています。
[27:13]	NumSets	セット数を示します。 0x7F = 16KB キャッシュサイズ 0xFF = 32KB キャッシュサイズ 0x1FF = 64KB キャッシュサイズ
[12:3]	Associativity	ウェイ数を示します。 b0000000011 = 4 ウェイ
[2:0]	LineSize	ワード数を示します。 b001 = 8 ワード / 行

CCSIDR にアクセスするには、次の命令を使用します。

```
MRC p15, 1, <Rd>, c0, c0, 0; Read current Cache Size Identification Register
```

CSSELR で命令キャッシュ値を読み出した場合、ビット [31:28] は b0010 です。

CSSELR でデータキャッシュ値を読み出した場合、ビット [31:28] は b0111 です。
「キャッシュサイズ選択レジスタ」（ページ 4-14）を参照して下さい。

4.3.5 キャッシュレベル ID レジスタ

CLIDR の特徴は次のとおりです。

目的 プロセッサに実装され、システム制御コプロセッサの管理下にあるキャッシュレベルを示します。

使用制限 CLIDR の使用制限は次のとおりです。
- 特権モードでのみアクセス可能です。
- セキュア状態と非セキュア状態とで共通です。

構成 すべての構成で使用可能です。

属性 レジスタの概要については、表 4-2 （ページ 4-8）を参照して下さい。

CLIDR のビット割り当てを、図 4-4 に示します。

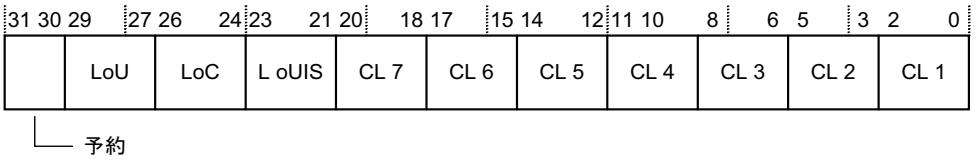


図 4-4 CLIDR のビット割り当て

CLIDR のビット割り当てを、表 4-6 に示します。

表 4-6 CLIDR のビット割り当て

ビット	名前	機能
[31:30]	-	UNP または SBZ
[29:27]	LoU	b001 = 統一レベル
[26:24]	LoC	b001 = コヒーレンシレベル
[23:21]	LoUIS	b001 = 内部共有可能統一レベル
[20:18]	CL 7	b000 = CL 7 にキャッシュが存在しない
[17:15]	CL 6	b000 = CL 6 にキャッシュが存在しない
[14:12]	CL 5	b000 = CL 5 にキャッシュが存在しない
[11:9]	CL 4	b000 = CL 4 にキャッシュが存在しない
[8:6]	CL 3	b000 = CL 3 にキャッシュが存在しない
[5:3]	CL 2	b000 = CL 2 に統一キャッシュが存在しない
[2:0]	CL 1	b011 = CL 1 に命令キャッシュとデータキャッシュが別々に存在する

CLIDR にアクセスするには、次の命令を使用します。

```
MRC p15, 1,<Rd>, c0, c0, 1; Read CLIDR
```

4.3.6 補助 ID レジスタ

AIDR の特徴は次のとおりです。

目的 実装固有の情報を提供します。

使用制限 AIDR の使用制限は次のとおりです。
- 特権モードでのみアクセス可能です。

- セキュア状態と非セキュア状態とで共通です。

構成 すべての構成で使用可能です。

属性 レジスタの概要については、表 4-2 （ページ 4-8）を参照して下さい。

補助レベル ID レジスタにアクセスするには、次の命令を使用します。

```
MRC p15,1,<Rd>,c0,c0,7; Read Auxiliary ID Register
```

注

この実装では AIDR は使用されません。

4.3.7 キャッシュサイズ選択レジスタ

CSSELR の特徴は次のとおりです。

目的 現在の CCSIDR を選択します。

使用制限 CSSELR の使用制限は次のとおりです。

- 特権モードでのみアクセス可能です。
- セキュア状態と非セキュア状態でバンクされます。

構成 すべての構成で使用可能です。

属性 レジスタの概要については、表 4-2 （ページ 4-8）を参照して下さい。

CSSELR のビット割り当てを、図 4-5 に示します。

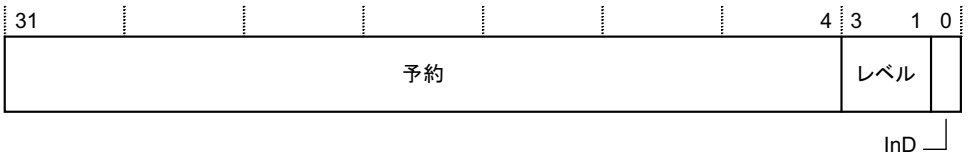


図 4-5 CSSELR のビット割り当て

CSSELR のビット割り当てを、表 4-7 に示します。

表 4-7 CSSELR のビット割り当て

ビット	名前	機能
[31:4]	-	UNP または SBZ
[3:1]	レベル	選択されたキャッシュレベル、RAZ/WI。 Cortex-A9 プロセッサ内のキャッシュレベルは 1 つしかないため、このフィールドの値は b000 です。
[0]	InD	1 = 命令キャッシュ 0 = データキャッシュ

CSSELR にアクセスするには、次の命令を使用します。

```
MRC p15, 2,<Rd>, c0, c0, 0; Read CSSELR
MCR p15, 2,<Rd>, c0, c0, 0; Write CSSELR
```

4.3.8 CP15 c1 レジスタの概要

CRn が c1 の場合にアクセス可能な CP15 レジスタを、表 4-8 に示します。

表 4-8 c1 システム制御レジスタ

Op1	CRm	Op2	名前	タイプ	リセット時の値	説明
0	c0	0	SCTLR	RW	-[a]	*「システム制御レジスタ」*
		1	ACTLR[b]	RW	0x00000000	*「補助制御レジスタ」*（ページ 4-18）
		2	CPACR	RW	[c]	*「コプロセッサアクセス制御レジスタ」*（ページ 4-20）
	c1	0	SCR[d]	RW	0x00000000	セキュア構成レジスタ
		1	SDER[c]	RW	0x00000000	*「セキュアデバッグ イネーブルレジスタ」*（ページ 4-22）
		2	NSACR	RW[e]	[f]	*「非セキュアアクセス制御レジスタ」*（ページ 4-23）
		3	VCR[c]	RW	0x00000000	*「仮想化制御レジスタ」*（ページ 4-24）

a. 入力信号によって異なります。*「システム制御レジスタ」*を参照して下さい。
b. 非セキュア状態では、NSACR[18] = 0 の場合は RO、NSACR[18] = 1 の場合は RW です。
c. NEON が存在する場合は 0x00000000 で、NEON が存在しないか、電力オフの場合は 0xC0000000 です。
d. 非セキュア状態ではアクセスできません。
e. このレジスタは、セキュア状態では読み出し / 書き込み、非セキュア状態では読み出し専用です。
f. NEON が存在する場合は 0x00000000 で、NEON が存在しない場合は 0x0000C000 です。

4.3.9 システム制御レジスタ

SCTLR の特徴は次のとおりです。

目的

次の制御と構成を提供します。

- メモリアライメントとエンディアン形式
- メモリ保護とフォールト動作
- MMU とキャッシュの稼動
- 割り込みと割り込みレイテンシ動作
- 例外ベクタの位置
- プログラムフロー予測

使用制限

SCTLR の使用制限は次のとおりです。

- 特権モードでのみアクセス可能です。
- 部分的にバンクされます。バンクビットとセキュア変更専用ビットを、表 4-9（ページ 4-16）に示します。

構成

すべての構成で使用可能です。

属性

レジスタの概要については、表 4-8 を参照して下さい。

SCTLR のビット割り当てを、図 4-6 に示します。

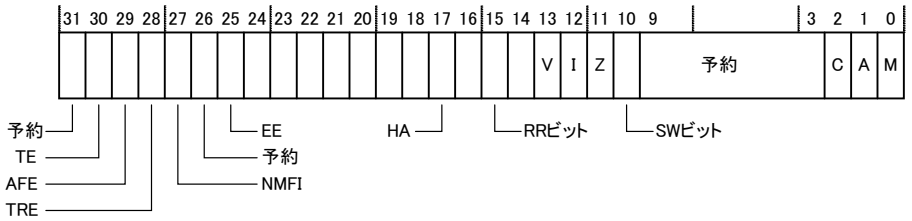


図 4-6 SCTLR のビット割り当て

SCTLR のビット割り当てを、表 4-9 に示します。

表 4-9 SCTLR のビット割り当て

ビット	名前	アクセス	機能
[31]	-	-	SBZ
[30]	TE	バンク	Thumb 例外イネーブル。 0 = 例外は、リセットも含めて、ARM 状態で処理されます。 1 = 例外は、リセットも含めて、Thumb 状態で処理されます。 リセット時の値は、**TEINIT** 信号によって定義されます。
[29]	AFE	バンク	アクセスフラグ イネーブルビット。 0 = フルアクセス許可動作。これはリセット時の値です。ソフトウェアによって、ARMv6K 動作とのバイナリ互換性が維持されます。 1 = 簡易アクセス許可動作。Cortex-A9 プロセッサは、AP[0] ビットをアクセスフラグとして再定義します。 AFE ビットの変更後は、TLB を無効化する必要があります。
[28]	TRE	バンク	このビットは、MMU 内の TEX 再マップ機能を制御します。 0 = TEX 再マップは不可能です。これはリセット時の値です。 1 = TEX 再マップが可能です。
[27]	NMFI	読み出し専用	マスク不能 FIQ のサポート。 このビットはソフトウェアで構成できません。 リセット時の値は、**CFGNMFI** 信号によって定義されます。
[26]	-	-	RAZ/SBZP
[25]	EE ビット	バンク	例外発生時の CPSR E ビットの設定方法を決定します。 0 = 例外発生時に CPSR E ビットが 0 にセットされます。 1 = 例外発生時に CPSR E ビットが 1 にセットされます。 この値は、変換テーブルルックアップ用の変換テーブルデータのエンディアン形式も示します。 0 = リトルエンディアン 1 = ビッグエンディアン リセット時の値は、**CFGEND** 信号によって定義されます。
[24]	-	-	RAZ/WI
[23:22]	-	-	RAO/SBOP
[21]	-	-	RAZ/WI
[20:19]	-	-	RAZ/SBZP
[18]	-	-	RAO/SBOP
[17]	HA	-	RAZ/WI
[16]	-	-	RAO/SBOP

表 4-9 SCTLR のビット割り当て （続き）

ビット	名前	アクセス	機能
[15]	-	-	RAZ/SBZP
[14]	RR	セキュア変更専用	キャッシュ、BTAC、マイクロ TLB の置換方式。このビットは、セキュア状態では読み出し / 書き込み、非セキュア状態では読み出し専用です。 0 = ランダム置換。これはリセット時の値です。 1 = ラウンドロビン置換
[13]	V	バンク	ベクタビット。このビットにより、次のように例外ベクタのベースアドレスが選択されます。 0 = 通常の例外ベクタ、ベースアドレス 0x00000000。セキュリティ拡張機能が実装されているため、このベースアドレスを再マッピングすることができます。 1 = 上位例外ベクタ (Hivecs)、ベースアドレス 0xFFFF0000。このベースアドレスは再マッピングされることはありません。 このビットが **VINITHI** から取得された場合は、セキュアバージョンのリセット時の値。
[12]	I ビット	バンク	命令をいずれかのキャッシュレベルにキャッシュできるかどうかを決定します。 0 = すべてのレベルで命令のキャッシュが不可能です。これはリセット時の値です。 1 = 命令のキャッシュが可能です。
[11]	Z ビット	バンク	プログラムフロー予測を可能にします。 0 = プログラムフロー予測が不可能です。これはリセット時の値です。 1 = プログラムフロー予測が可能です。
[10]	SW ビット	バンク	SWP/SWPB イネーブルビット。 0 = SWP および SWPB は未定義です。これはリセット時の値です。 1 = SWP および SWPB は正常に動作します。
[9:7]	-	-	RAZ/SBZP
[6:3]	-	-	RAO/SBOP
[2]	C ビット	バンク	データをいずれかのキャッシュレベルでキャッシュできるかどうかを決定します。 0 = すべてのレベルでデータのキャッシュが不可能です。これはリセット時の値です。 1 = データのキャッシュが可能です。
[1]	A ビット	バンク	厳格なデータアライメントによって、データアクセス時のアライメントフォールトの検出を可能にします。 0 = 厳格なアライメントフォールト チェックが不可能です。これはリセット時の値です。 1 = 厳格なアライメントフォールト チェックが可能です。
[0]	M ビット	バンク	MMU を稼働します。 0 = MMU が非稼働です。これはリセット時の値です。 1 = MMU が稼働状態です。

セキュアまたは非セキュアのユーザモードから SCTLR の読み書きを試みると、未定義命令例外が発生します。

CP15SDISABLE が HIGH のときに、セキュア特権モードでこのレジスタへの書き込みを試みると、未定義命令例外が発生します。

非セキュア特権モードでセキュア変更専用ビットに書き込みを試みても無視されます。

セキュア変更専用ビットの読み出しを試みると、セキュアビットの値が返されます。

読み出し専用ビットの変更の試みは無視されます。

SCTRL にアクセスするには、次の命令を使用します。

```
MRC p15, 0,<Rd>, c1, c0, 0; Read SCTLR
MCR p15, 0,<Rd>, c1, c0, 0; Write SCTLR
```

4.3.10 補助制御レジスタ

ACTLR の特徴は次のとおりです。

目的

次の要素を制御します。

- パリティチェック（実装されている場合）
- 1 ウェイでの割り当て
- レベル 2 キャッシュを使用した排他的キャッシュ
- コヒーレンシモード、*対称マルチプロセッシング* (SMP)、または*非対称マルチプロセッシング* (AMP)
- AXI 上での投機的アクセス
- キャッシュ、分岐予測、および TLB 保守操作のブロードキャスト
- L2C-310 キャッシュの割り当て。
  - 0 のフルライン書き込みモード

使用制限

ACTLR の使用制限は次のとおりです。

- 特権モードでのみアクセス可能です。
- セキュア状態と非セキュア状態に共通しています。
- セキュア状態では RW です。
- NSACR.NS_SMP = 0 の場合、非セキュア状態では RO です。
- NSACR.NS_SMP = 1 の場合、非セキュア状態では RW です。この場合は、SMP ビット以外のビットへの書き込みが無視されます。

構成

すべての構成で使用可能です。

- すべての構成で、SMP ビット = 0 の場合は、内部キャッシュ可能共有可能属性がキャッシュ不可として扱われます。
- マルチプロセッサ構成で、SMP ビットがセットされている場合には、次の規則が適用されます。
  - FW ビットがセットされている場合は、キャッシュ保守操作と TLB 保守操作のブロードキャストが許可されます。
  - FW ビットがセットされている場合は、同じコヒーレンシクラスタ内の他の Cortex-A9 プロセッサによる、キャッシュ保守操作と TLB 保守操作のブロードキャストの受信が許可されます。
  - Cortex-A9 プロセッサは、同じコヒーレンシクラスタ内の他の Cortex-A9 プロセッサとの間で、共有内部ライトバック、書き込み割り当てアクセスのコヒーレンシ要求を送受信できます。

属性

レジスタの概要については、表 4-8 （ページ 4-15）を参照して下さい。

ACTLR のビット割り当てを、図 4-7 に示します。

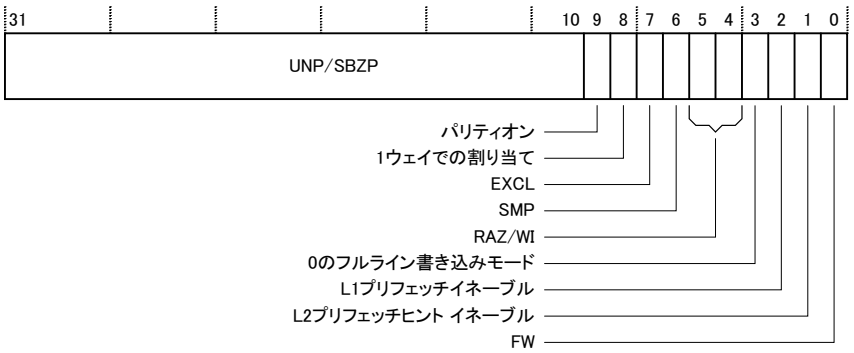


図 4-7 ACTLR のビット割り当て

ACTLR のビット割り当てを、表 4-10 に示します。

表 4-10 ACTLR のビット割り当て

ビット	名前	機能
[31:10]	-	UNP または SBZP
[9]	パリティオン	パリティチェックのサポート（実装されている場合）。 0 = 非稼働です。これはリセット時の値です。 1 = 稼働しています。 パリティチェックが実装されていない場合、このビットは 0 として読み出され、書き込みは無視されます。
[8]	1 ウェイでの割り当て	1 キャッシュウェイのみでの割り当てを可能にします。キャッシュ汚染を減らすため、メモリコピー操作と組で使用されます。リセット時の値は 0 です。
[7]	EXCL	排他キャッシュビット。 排他キャッシュ構成では、レベル 1 とレベル 2 に同時にデータを存在させることができません。また、内部キャッシュ属性がライトバック キャッシュ可能でレベル 1 に割り当てられている場合は、レベル 1 からの退出時にしかデータをキャッシュできません。キャッシュコントローラが排他キャッシュ用にも構成されていることを確認して下さい。 0 = 非稼働です。これはリセット時の値です。 1 = 稼働しています。
[6]	SMP	Cortex-A9 プロセッサがコヒーレンシの一部かどうかを示します。 ユニプロセッサ構成では、このビットがセットされていると、内部キャッシュ可能共有はキャッシュ可能として扱われます。リセット時の値は 0 です。
[5:4]	-	RAZ/WI
[3]	0 のフルライン書き込みモード	0 のフルライン書き込みモードを可能にします[a]。リセット時の値は 0 です。
[2]	レベル 1 プリフェッチ イネーブル	Dside プリフェッチ。 0 = 非稼働です。これはリセット時の値です。 1 = 稼働しています。
[1]	レベル 2 プリフェッチ イネーブル	プリフェッチヒント イネーブル[a]。リセット時の値は 0 です。
[0]	FW	キャッシュおよび TLB 保守のブロードキャスト。 0 = 非稼働です。これはリセット時の値です。 1 = 稼働しています。 Cortex-A9 プロセッサが 1 つしか存在しない場合は RAZ/WI です。

a. この機能は、Cortex-A9 AXI マスタポートに接続されたスレーブでサポートされている場合のみ、可能にする必要があります。L2-310 キャッシュコントローラはこの機能をサポートしています。*「レベル 2 メモリインタフェースへのアクセスの最適化」*（ページ 8-7）を参照して下さい。

ACTLR にアクセスするには、読み出し - 変更 - 書き込みの手法を使用する必要があります。ACTLR にアクセスするには、次の命令を使用します。

```
MRC p15, 0,<Rd>, c1, c0, 1; Read ACTLR
MCR p15, 0,<Rd>, c1, c0, 1; Write ACTLR
```

CP15SDISABLE が HIGH のときに、セキュア特権モードでこのレジスタへの書き込みを試みると、未定義命令例外が発生します。

4.3.11 コプロセッサアクセス制御レジスタ

CPACR の特徴は次のとおりです。

目的

- コプロセッサ CP11 と CP10 のアクセス権を設定します。
- 特定のコプロセッサがシステム内に存在するかどうかを、ソフトウェアで判断できるようにします。

注

このレジスタは、CP14 または CP15 へのアクセスに影響を与えません。

使用制限

CPACR の使用制限は次のとおりです。

- 特権モードでのみアクセス可能です。
- セキュア状態と非セキュア状態とで共通です。

構成

すべての構成で使用可能です。

属性

レジスタの概要については、表 4-8 （ページ 4-15）を参照して下さい。

CPACR のビット割り当てを、図 4-8 に示します。

図 4-8 CPACR のビット割り当て

CPACR のビット割り当てを、表 4-11 に示します。

表 4-11 CPACR のビット割り当て

ビット	名前	機能
[31]	ASEDIS	アドバンスト SIMD 拡張機能の非稼働。 0 = このビットによって、命令が未定義になることはありません。 1 = *『ARM アーキテクチャ リファレンスマニュアル』*で、アドバンスト SIMD 拡張機能の一部であるが、VFPv3 命令ではないとされているすべての命令エンコードが未定義になります。 詳細については、*『Cortex-A9 浮動小数点ユニット テクニカルリファレンス マニュアル』*と*『Cortex-A9 NEON メディア処理エンジン テクニカルリファレンス マニュアル』*を参照して下さい。 VFP のみが実装されている場合は、RAO/WI です。 VFP と NEON のいずれも実装されていない場合は、UNK/SBZP です。
[30]	D32DIS	VFP レジスタファイルの D16 ～ D31 の使用を不可能にします。 0 = このビットによって、命令が未定義になることはありません。 1 = レジスタの D16 ～ D31 のいずれかにアクセスする場合、*『ARM アーキテクチャ リファレンスマニュアル』*で VFPv3 命令とされているすべての命令エンコードが未定義になります。 詳細については、*『Cortex-A9 浮動小数点ユニット テクニカルリファレンス マニュアル』*と*『Cortex-A9 NEON メディア処理エンジン テクニカルリファレンス マニュアル』*を参照して下さい。 VFP のみを使用して実装されている場合は、RAO/WI です。VFP と NEON のいずれも実装されていない場合は、UNK/SBZP です。
[29:24]	-	RAZ/WI
[23:22]	cp11	コプロセッサのアクセス許可を定義します。リセット時の状態はアクセス拒否で、コプロセッサが存在しない場合もこの状態です。 b00 = アクセス拒否。これはリセット時の値です。アクセスを試みると、未定義命令例外が生成されます。 b01 = 特権モードアクセスのみ b10 = 予約 b11 = 特権モードアクセスとユーザモード アクセス
[21:20]	cp10	コプロセッサのアクセス許可を定義します。リセット時の状態はアクセス拒否で、コプロセッサが存在しない場合もこの状態です。 b00 = アクセス拒否。これはリセット時の値です。アクセスを試みると、未定義命令例外が生成されます。 b01 = 特権モードアクセスのみ b10 = 予約 b11 = 特権モードアクセスとユーザモード アクセス
[19:0]	-	RAZ/WI

非セキュア状態でのコプロセッサへのアクセスは、*「非セキュアアクセス制御レジスタ」*（ページ 4-23）で設定されているアクセス許可によって異なります。

CPACR アクセスビットの読み書きは、*「非セキュアアクセス制御レジスタ」*（ページ 4-23）のコプロセッサに対応するビットによって異なります。

CPACR にアクセスするには、次の命令を使用します。

```
MRC p15, 0,<Rd>, c1, c0, 2; Read Coprocessor Access Control Register
MCR p15, 0,<Rd>, c1, c0, 2; Write Coprocessor Access Control Register
```

CPACR 更新の直後に ISB を実行する必要があります。*『ARM アーキテクチャ リファレンスマニュアル』*の「メモリバリア」を参照して下さい。ISB とレジスタ更新との間では、アクセス権の変更の影響を受ける命令は実行しないで下さい。

システムに特定のコプロセッサが存在するかどうかを判断するには、対象のコプロセッサのアクセスビットに b11 を書き込みます。システムにそのコプロセッサが存在しない場合、アクセス権は b00 に設定されたままです。

注

NEON または VFP システムレジスタにアクセスする前に、コプロセッサ 10 とコプロセッサ 11 の両方を稼働状態にする必要があります。

4.3.12 セキュアデバッグ イネーブルレジスタ

SDER の特徴は次のとおりです。

目的 Cortex-A9 デバッグを制御します。

使用制限 SDER の使用制限は次のとおりです。

- 特権モードでのみアクセス可能です。
- セキュア状態でのみアクセス可能です。非セキュア状態でアクセスしようとすると、未定義命令例外が発生します。

構成 すべての構成で使用可能です。

属性 レジスタの概要については、表 4-8 （ページ 4-15）を参照して下さい。

SDER のビット割り当てを、図 4-9 に示します。

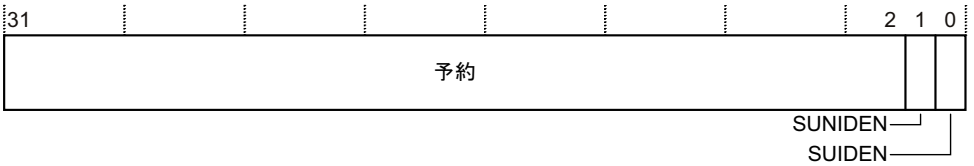


図 4-9 SDER のビット割り当て

SDER のビット割り当てを、表 4-12 に示します。

表 4-12 SDER のビット割り当て

ビット	名前	機能
[31:2]	-	予約
[1]	セキュアユーザ非侵襲性デバッグイネーブル	0 = セキュアユーザ モードで非侵襲性デバッグが許可されません。これはリセット時の値です。 1 = セキュアユーザ モードで非侵襲性デバッグが許可されます。
[0]	セキュアユーザ侵襲性デバッグイネーブル	0 = セキュアユーザ モードで侵襲性デバッグが許可されません。これはリセット時の値です。 1 = セキュアユーザ モードで侵襲性デバッグが許可されます。

SDER にアクセスするには、次の命令を使用します。

```
MRC p15,0,<Rd>,c1,c1,1; Read Secure debug enable Register
MCR p15,0,<Rd>,c1,c1,1; Write Secure debug enable Register
```

4.3.13 非セキュアアクセス制御レジスタ

NSACR の特徴は次のとおりです。

目的 コプロセッサの非セキュアアクセス許可を設定します。

使用制限 NSACR の使用制限は次のとおりです。

- 特権モードでのみアクセス可能です。
- セキュア状態では読み出し / 書き込みレジスタ
- 非セキュア状態では読み出し専用レジスタ

注

このレジスタは、デバッグ制御レジスタまたはシステム制御コプロセッサへの、非セキュアアクセス許可には影響を与えません。

構成 すべての構成で使用可能です。

属性 レジスタの概要については、表 4-8 （ページ 4-15）を参照して下さい。

NSACR のビット割り当てを、図 4-10 に示します。

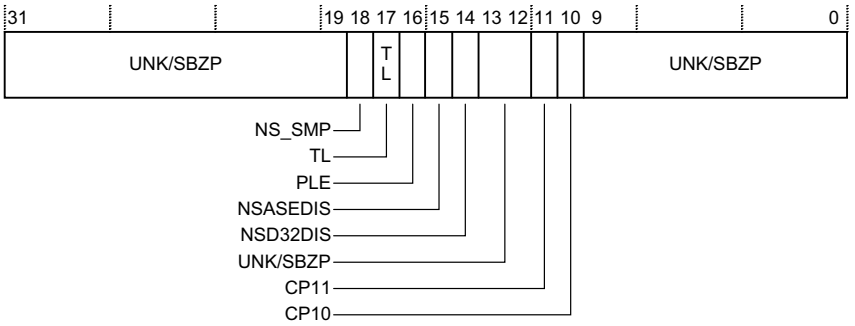


図 4-10 NSACR のビット割り当て

NSACR のビット割り当てを、表 4-13 に示します。

表 4-13 NSACR のビット割り当て

ビット	名前	機能
[31:19]	-	UNK/SBZP
[18]	NS_SMP	補助制御レジスタの SMP ビットが非セキュア状態で書き込み可能かどうかを決定します。 0 = 非セキュア状態で補助制御レジスタに書き込むと、未定義例外が取得され、SMP ビットへの書き込みは無視されます。これはリセット時の値です。 1 = 非セキュア状態で補助制御レジスタに書き込むことによって、SMP ビットの値を変更できます。他のビットへの書き込みは無視されます。
[17]	TL	ロック可能 TLB エントリが非セキュア状態で割り当て可能かどうかを決定します。 0 = ロック可能 TLB エントリは割り当てできません。これはリセット時の値です。 1 = ロック可能 TLB エントリを割り当てることができます。
[16]	PLE	プリロードエンジン リソースに対する NS アクセスを制御します。 0 = CP15 c11 へのセキュアアクセスのみが許可されます。すべての CP15 c11 への非セキュアアクセスは UNDEF にトラップされます。これはデフォルト値です。 1 = CP15 c11 ドメインへの非セキュアアクセスが許可されます。つまり、非セキュア状態で PLE リソースにアクセスできます。 プリロードエンジンが実装されていない場合、このビットは RAZ/WI です。第 9 章 *プリロードエンジン*を参照して下さい。

表 4-13 NSACR のビット割り当て （続き）

ビット	名前	機能
[15]	NSASEDIS	非セキュアアドバンスト SIMD 拡張機能を非稼働にします。 0 = このビットは、CPACR.ASEDIS への書き込みに影響しません。これはリセット時の値です。 1 = 非セキュア状態で実行している場合、CPACR.ASEDIS ビットの値が 1 に固定され、書き込みは無視されます。 詳細については、*『Cortex-A9 浮動小数点ユニット テクニカルリファレンス マニュアル』*と*『Cortex-A9 NEON メディア処理エンジン テクニカルリファレンス マニュアル』*を参照して下さい。
[14]	NSD32DIS	VFP レジスタファイルの D16 ～ D31 の非セキュア使用を不可能にします。 0 = このビットは、CPACR.D32DIS への書き込みに影響しません。これはリセット時の値です。 1 = 非セキュア状態で実行している場合、CPACR.D32DIS ビットの値が 1 に固定され、書き込みは無視されます。 詳細については、*『Cortex-A9 浮動小数点ユニット テクニカルリファレンス マニュアル』*と*『Cortex-A9 NEON メディア処理エンジン テクニカルリファレンス マニュアル』*を参照して下さい。
[13:12]	-	UNK/SBZP
[11]	CP11	非セキュア状態でのコプロセッサ 11 へのアクセス許可を決定します。 0 = セキュアアクセスのみ。これはリセット時の値です。 1 = セキュアアクセスまたは非セキュアアクセス
[10]	CP10	非セキュア状態でのコプロセッサ 10 へのアクセス許可を決定します。 0 = セキュアアクセスのみ。これはリセット時の値です。 1 = セキュアアクセスまたは非セキュアアクセス
[9:0]	-	UNK/SBZP

NSACR にアクセスするには、次の命令を使用します。

```
MRC p15, 0,<Rd>, c1, c1, 2; Read NSACR data
MCR p15, 0,<Rd>, c1, c1, 2; Write NSACR data
```

詳細については、*『Cortex-A9 浮動小数点ユニット テクニカルリファレンス マニュアル』*と*『Cortex-A9 NEON メディア処理エンジン テクニカルリファレンス マニュアル』*を参照して下さい。

4.3.14 仮想化制御レジスタ

VCR の特徴は次のとおりです。

目的 *カレントプログラム ステータスレジスタ* (CPSR) の A、I、F ビットの値に関係なく、例外を発生させます。

使用制限 VCR の使用制限は次のとおりです。

- 特権モードでのみアクセス可能です。
- セキュア状態でのみアクセス可能です。

構成 すべての構成で使用可能です。

属性 レジスタの概要については、表 4-8 （ページ 4-15）を参照して下さい。

VCR のビット割り当てを、図 4-11 に示します。

図 4-11 VCR のビット割り当て

VCR のビット割り当てを、表 4-14 に示します。

表 4-14 VCR のビット割り当て

ビット	名前	機能
[31:9]	-	UNK/SBZP
[8]	AMO	アボートマスク オーバライド。 プロセッサが非セキュア状態で、SCR.EA ビットがセットされており、AMO ビットがセットされている場合は、これによって、CPSR.A ビットの値に関係なく、非同期データアボート例外を取得できます。 プロセッサがセキュア状態、または SCR.EA ビットがセットされていない場合、AMO ビットは無視されます。
[7]	IMO	IRQ マスクオーバライド。 プロセッサが非セキュア状態で、SCR.IRQ ビットがセットされており、IMO ビットがセットされている場合は、これによって、CPSR.I ビットの値に関係なく、IRQ 例外を取得できます。 プロセッサがセキュア状態、または SCR.IRQ ビットがセットされていない場合、IMO ビットは無視されます。
[6]	IFO	FIQ マスクオーバライド。 プロセッサが非セキュア状態で、SCR.FIQ ビットがセットされており、IFO ビットがセットされている場合は、これによって、CPSR.F ビットの値に関係なく、FIQ 例外を取得できます。 プロセッサがセキュア状態、または SCR.FIQ ビットがセットされていない場合、IFO ビットは無視されます。
[5:0]	-	UNK/SBZP

VCR にアクセスするには、次の命令を使用します。

```
MRC p15, 0,<Rd>, c1, c1, 3; Read VCR data
MCR p15, 0,<Rd>, c1, c1, 3; Write VCR data
```

4.3.15 CP15 c2 レジスタの概要

CRn が c2 の場合にアクセス可能なシステム制御レジスタを、表 4-15 に示します。

表 4-15 c2 システム制御レジスタ

Op1	CRm	Op2	名前	タイプ	リセット時の値	説明
0	c0	0	TTBR0	RW	-	
		1	TTBR1	RW	-	変換テーブルベースレジスタ 1
		2	TTBCR	RW	0x00000000[a]	変換テーブルベース制御レジスタ

a. セキュア状態でのみ。非セキュアバージョンを必要な値でプログラムする必要があります。

4.3.16 CP15 c3 レジスタの概要

CRn が c3 の場合にアクセス可能なシステム制御レジスタを、表 4-16 に示します。

表 4-16 c3 システム制御レジスタ

Op1	CRm	Op2	名前	タイプ	リセット時の値	説明
0	c0	0	DACR	RW	-	ドメインアクセス制御レジスタ

4.3.17 CP15 c4、未使用

CRn を c4 に設定してアクセスする CP15 レジスタは存在しません。

4.3.18 CP15 c5 レジスタの概要

CRn が c5 の場合にアクセス可能なシステム制御レジスタを、表 4-17 に示します。

表 4-17 c5 システム制御レジスタ

Op1	CRm	Op2	名前	タイプ	リセット時の値	説明
0	c0	0	DFSR	RW	-	データフォールト ステータスレジスタ
		1	IFSR	RW	-	命令フォールト ステータスレジスタ
	c1	0	ADFSR	-	-	補助データフォールト ステータスレジスタ
		1	AIFSR	-	-	補助命令フォールト ステータスレジスタ

4.3.19 CP15 c6 レジスタの概要

CRn が c6 の場合にアクセス可能なシステム制御レジスタを、表 4-18 に示します。

表 4-18 c6 システム制御レジスタ

Op1	CRm	Op2	名前	タイプ	リセット時の値	説明
0	c0	0	DFAR	RW	-	データフォールト アドレスレジスタ
		2	IFAR	RW	-	命令フォールト アドレスレジスタ

4.3.20 CP15 c7 レジスタの概要

CRn が c7 の場合にアクセス可能なシステム制御レジスタを、表 4-19 に示します。

表 4-19 c7 システム制御レジスタ

Op1	CRm	Op2	名前	タイプ	リセット時の値	説明
0	c0	0 ～ 3	予約	WO	-	-
		4	NOP[a]	WO	-	-
	c1	0	ICIALLUIS	WO	-	キャッシュ操作レジスタ
		6	BPIALLIS	WO	-	
		7	予約	WO	-	
	c4	0	PAR	RW	_	-
	c5	0	ICIALLU	WO	-	キャッシュ操作レジスタ
		1	ICIMVAU	WO	_	
		2 ～ 3	予約	WO	_	
		4	ISB	WO	ユーザ	*「非推奨レジスタ」*（ページ 4-2）
		6	BPIALL	WO	_	キャッシュ操作レジスタ
	c6	1	DCIMVAC	WO	_	
		2	DCISW	WO	_	
0	c8	0 ～ 7	V2PCWPR	WO	-	VA から PA への変換操作
	c10	1	DCCVAC	WO	-	キャッシュ操作レジスタ
		2	DCCSW	WO	-	
		4	DSB	WO	ユーザ	*「非推奨レジスタ」*（ページ 4-2）
		5	DMB	WO	ユーザ	
	c11	1	DCCVAU	WO	-	キャッシュ操作レジスタ
	c14	1	DCCIMVAC	WO	-	
		2	DCCISW	WO	-	

a. この操作は、WFI 命令によって実行されます。*「非推奨レジスタ」*（ページ 4-2）も参照して下さい。

4.3.21 CP15 c8 レジスタの概要

CRn が c8 の場合にアクセス可能なシステム制御レジスタを、表 4-20 に示します。

表 4-20 c8 システム制御レジスタ

Op1	CRm	Op2	名前	タイプ	リセット時の値	説明
0	c3	0	TLBIALLIS[a]	WO	-	-
		1	TLBIMVAIS[b]	WO	-	-
		2	TLBIASIDIS[b]	WO	-	-
		3	TLBIMVAAIS[a]	WO	-	-
	c5、c6、または c7	0	TLBIALL[a]	WO	-	-
		1	TLBIMVA[b]	WO	-	-
		2	TLBIASID[b]	WO	-	-
		3	TLBIMVAA[a]	WO	-	-

a. ロックダウンされたエントリに影響を与えません。
b. 一致したロック済みエントリを無効化します。

「ASID の一致による TLB エントリの無効化」（ページ 4-41）も参照して下さい。

4.3.22 CP15 c9 レジスタの概要

CRn が c9 の場合にアクセス可能なシステム制御レジスタを、表 4-21 に示します。

表 4-21 c9 システム制御レジスタ

Op1	CRm	Op2	名前	タイプ	リセット時の値	説明
0	c12	0	PMCR	RW	0x41093000	パフォーマンスモニタ制御レジスタ
		1	PMCNTENSET	RW	0x00000000	カウントイネーブル セットレジスタ
		2	PMCNTENCLR	RW	0x00000000	カウントイネーブル クリアレジスタ
		3	PMOVSR	RW	-	オーバフローフラグ ステータスレジスタ
		4	PMSWINC	WO	-	ソフトウェアインクリメント レジスタ
		5	PMSELR	RW	0x00000000	イベントカウンタ選択レジスタ
	c13	0	PMCCNTR	RW	-	サイクルカウント レジスタ
		1	PMXEVTYPER	RW	-	イベント選択レジスタ
		2	PMXEVCNTR	RW	-	パフォーマンスモニタ カウントレジスタ
	c14	0	PMUSERENR	RW[a]	0x00000000	ユーザイネーブル レジスタ
		1	PMINTENSET	RW	0x00000000	割り込みイネーブルセット レジスタ
		2	PMINTENCLR	RW	0x00000000	割り込みイネーブルクリア レジスタ

a. ユーザモードでは RO

第 11 章 *パフォーマンス監視ユニット*を参照して下さい。

4.3.23 CP15 c10 レジスタの概要

CRn が c10 の場合にアクセス可能なシステム制御レジスタを、表 4-22 に示します。

表 4-22 c10 システム制御レジスタ

Op1	CRm	Op2	名前	タイプ	リセット時の値	説明
0	c0	0	TLB ロックダウンレジスタ[a]	RW	0x00000000	*「TLB ロックダウンレジスタ」*
	c2	0	PRRR	RW	0x00098AA4	1 次領域再マップレジスタ
		1	NRRR	RW	0x44E048E0	ノーマルメモリ再マップレジスタ

a. 非セキュア状態では、NSCAR.TL = 0 ならばアクセス不可、NSACR.TL = 1 ならば RW です。

4.3.24 TLB ロックダウンレジスタ

TLB ロックダウンレジスタの特徴は次のとおりです。

目的 ハードウェア変換テーブルウォークによって、どこに TLB エントリを配置するかを制御します。TLB エントリは次のいずれかの位置に配置できます。

- TLB のセットアソシエイティブ領域
- TLB のロックダウン領域と、ロックダウン領域内の書き込み対象エントリ
  TLB のロックダウン領域には 4 つのエントリがあります。

使用制限 TLB ロックダウンレジスタの使用制限は次のとおりです。

- 特権モードでのみアクセス可能
- セキュア状態と非セキュア状態とで共通です。
- NSACR.TL が 0 の場合はアクセス不可です。

構成 すべての構成で使用可能です。

属性 レジスタの概要については、表 4-22 を参照して下さい。

TLB ロックダウンレジスタのビット割り当てを、図 4-12 に示します。

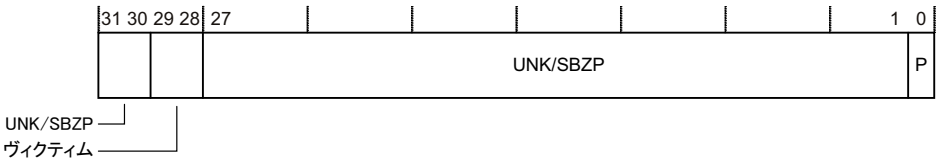


図 4-12 TLB ロックダウンレジスタのビット割り当て

TLB ロックダウンレジスタのビット割り当てを、表 4-23 に示します。

表 4-23 TLB ロックダウンレジスタのビット割り当て

ビット	名前	機能
[31:30]	-	UNK/SBZP
[29:28]	ヴィクティム	ロックダウン領域
[27:1]	-	UNK/SBZP
[0]	P	保存ビット。リセット時の値は 0 です。

TLB ロックダウンレジスタにアクセスするには、次の命令を使用します。

```
MRC p15, 0,<Rd>, c10, c0, 0; Read TLB Lockdown victim
MCR p15, 0,<Rd>, c10, c0, 0; Write TLB Lockdown victim
```

TLB ロックダウンレジスタに、保存ビット（P ビット）がセットされた状態で書き込むと、次のような効果があります。

1 後続のハードウェア変換テーブルウォークにおいて、0 ～ 3 の範囲のヴィクティムで指定されたエントリにあるロックダウン領域に TLB エントリが配置されます。

0 後続のハードウェア変換テーブルウォークにおいて、TLB のセットアソシエイティブ領域に TLB エントリが配置されます。

「ASID の一致による TLB エントリの無効化」（ページ 4-41）を参照して下さい。

4.3.25 CP15 c11 レジスタの概要

CRn が c11 の場合にアクセス可能なシステム制御レジスタを、表 4-24 に示します。

表 4-24 c11 システム制御レジスタ

Op1	CRm	Op2	名前	タイプ	リセット時の値	説明
0	c0	0	PLEIDR	RO[a]	-	*「PLE ID レジスタ」*
		2	PLEASR	RO[a]	-	*「PLE 動作ステータスレジスタ」*（ページ 4-31）
		4	PLEFSR	RO[a]	-	*「PLE FIFO ステータスレジスタ」*（ページ 4-32）
	c1	0	PLEUAR	特権 R/W ユーザ RO	-	*「プリロードエンジン ユーザアクセス許可レジスタ」*（ページ 4-32）
		1	PLEPCR	特権 R/W ユーザ RO	-	*「プリロードエンジン パラメータ制御レジスタ」*（ページ 4-33）

a. PLE が存在しない場合は RAZ です。

4.3.26 PLE ID レジスタ

PLEIDR の特徴は次のとおりです。

目的 PLE が存在するかどうか、およびその FIFO のサイズを示します。

使用制限 PLEIDR の使用制限は次のとおりです。

- セキュア状態と非セキュア状態とで共通です。
- 構成ビットに関係なく、ユーザモードと特権モードでアクセス可能です。

構成 PLE が存在するかどうかに関係なく、すべての Cortex-A9 構成で使用可能です。

属性 表 4-24 を参照して下さい。

PLEIDR のビット割り当てを、図 4-13 に示します。

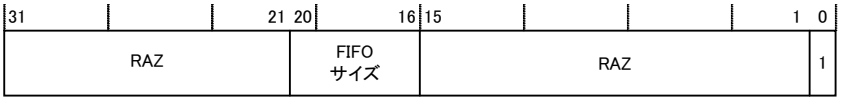


図 4-13 PLEIDR のビット割り当て

PLEIDR のビット割り当てを、表 4-25 に示します。

表 4-25 PLEIDR のビット割り当て

ビット	名前	機能
[31:21]	-	-
[20:16]	PLE FIFO サイズ	使用可能な値は次のとおりです。 • 5'b00000 は、PLE が存在しないことを示します。 • 5'b00100 は、PLE が存在し、FIFO サイズが 4 エントリであることを示します。 • 5'b01000 は、PLE が存在し、FIFO サイズが 8 エントリであることを示します。 • 5'b10000 は、PLE が存在し、FIFO サイズが 16 エントリであることを示します。
[15:1]	-	RAZ
[0]	-	1 で、この構成にプリロードエンジンが存在することを示します。

PLEIDR にアクセスするには、次の命令を使用します。

```
MRC p15, 0, <Rt>, c11, c0, 0; Read PLEIDR
```

4.3.27 PLE 動作ステータスレジスタ

PLEASR の特徴は次のとおりです。

目的 PLE エンジンが現在アクティブかどうかを示します。

使用制限 PLEASR の使用制限は次のとおりです。

- セキュア状態と非セキュア状態とで共通です。
- 構成ビットに関係なく、ユーザモードと特権モードでアクセス可能です。

構成 PLE が存在するかどうかに関係なく、すべての Cortex-A9 構成で使用可能です。

属性 表 4-24 （ページ 4-30）を参照して下さい。

PLEASR のビット割り当てを、図 4-14 に示します。

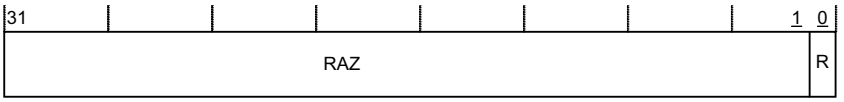


図 4-14 PLEASR のビット割り当て

PLEASR のビット割り当てを、表 4-26 に示します。

表 4-26 PLEASR のビット割り当て

ビット	名前	機能
[31:1]	-	-
[0]	R	PLE チャネル実行中 1 = プリロードエンジンが PLE 要求の処理中です。

PLEASR にアクセスするには、次の命令を使用します。

```
MRC p15, 0, <Rt>, c11, c0, 2; Read PLEASR
```

4.3.28 PLE FIFO ステータスレジスタ

PLEFSR の特徴は次のとおりです。

目的 PLE FIFO 内で使用可能なエントリがどのくらい残っているかを示します。

使用制限 PLEFSR の使用制限は次のとおりです。

- セキュア状態と非セキュア状態とで共通です。
- 構成ビットに関係なく、ユーザモードと特権モードでアクセス可能です。

NSAC.PLE が非セキュアアクセスを制御します。

構成 PLE が存在するかどうかに関係なく、すべての Cortex-A9 構成で使用可能です。

属性 表 4-24 （ページ 4-30）を参照して下さい。

PLEFSR のビット割り当てを、図 4-15 に示します。

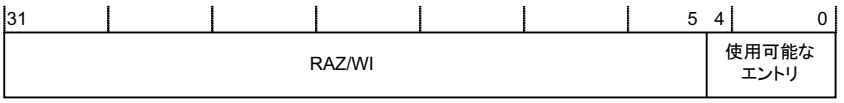


図 4-15 PLEFSR のビット割り当て

PLEFSR のビット割り当てを、表 4-27 に示します。

表 4-27 PLEFSR のビット割り当て

ビット	名前	機能
[31:5]	-	-
[4:0]	使用可能なエントリ	PLE FIFO 内で使用可能なエントリ数 これは、構成固有の FIFO 内の総エントリ数と、すでにプログラムされているエントリ数との差です。

PLEFSR は、新しい PLE チャネルをプログラムする前に、エントリが使用可能かどうかをチェックするために使用します。

PLEFSR にアクセスするには、次の命令を使用します。

```
MRC p15, 0, <Rt>, c11, c0, 4; Read the PLESFR
```

4.3.29 プリロードエンジン ユーザアクセス許可レジスタ

PLEUAR の特徴は次のとおりです。

目的 ユーザモードで PLE 操作が使用可能かどうかを制御します。

使用制限 PLEUAR の使用制限は次のとおりです。

- セキュア状態と非セキュア状態とで共通です。
- 構成ビットに関係なく、ユーザモードと特権モードでアクセス可能です。

構成 プリロードエンジンが存在する構成でのみ使用可能です。それ以外の構成では、未定義命令例外が取得されます。

属性 表 4-24 （ページ 4-30）を参照して下さい。

PLEUAR のビット割り当てを、図 4-16 に示します。

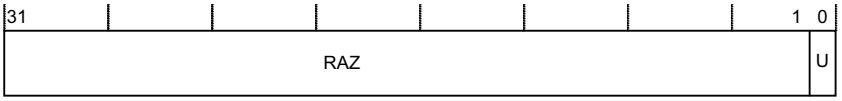


図 4-16 PLEUAR のビット割り当て

PLEUAR のビット割り当てを、表 4-28 に示します。

表 4-28 PLEUAR のビット割り当て

ビット	名前	機能
[31:1]	-	RAZ
[0]	U	ユーザアクセス許可 1 = ユーザモードで PLE 領域にアクセスして、PLE 操作を実行できます。

PLEUAR にアクセスするには、次の命令を使用します。

```
MCR p15, 0, <Rt>, c11, c1, 0; Read PLEAUR
MRC p15, 0, <Rt>, c11, c1, 0; Write PLEAUR
```

4.3.30 プリロードエンジン パラメータ制御レジスタ

PLEPCR の特徴は次のとおりです。

目的 PLE の発行レートと転送サイズを制限するために、特権モードでのみ使用可能な PLE 制御パラメータを保存します。

使用制限 PLEPCR の使用制限は次のとおりです。

- 読み出し / 書き込みレジスタ
- 特権モードでのみアクセス可能
- セキュア状態と非セキュア状態とで共通です。
- NSACR.PLE が非セキュアアクセスを制御します。

構成 プリロードエンジンが存在する構成でのみ使用可能です。それ以外の構成では、未定義命令例外が取得されます。

属性 表 4-24 （ページ 4-30）を参照して下さい。

PLEPCR のビット割り当てを、図 4-17 に示します。

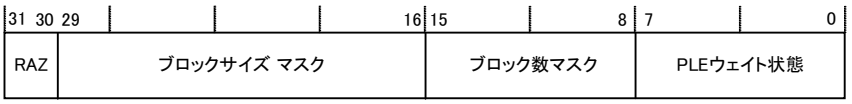


図 4-17 PLEPCR のビット割り当て

PLEPCR のビット割り当てを、表 4-29 に示します。

表 4-29 PLEPCR のビット割り当て

ビット	名前	機能
[31:30]	-	RAZ
[29:16]	ブロックサイズ マスク	特権モードで PLE 転送の最大ブロックサイズを制限できるようにします。 転送されるブロックサイズは、（ブロックサイズ）&（ブロックサイズ マスク）です。 例えば、14'b11111111111111 のブロックサイズ マスクは、最大値が 16k × 4 バイトのブロックサイズの転送を許可します。14'b00000000000000 のブロックサイズ マスクは、ブロックサイズを 1 × 4 バイトに制限します。
[15:8]	ブロック数マスク	特権モードで単一の PLE 転送の最大ブロック数を制限できるようにします。 転送されるブロック数は、（ブロック数）&（ブロック数マスク）です。 例えば、8'b11111111 のブロック数マスクは、最大 256 個のブロックの転送を許可します。8'b00000000 のブロック数マスクは、転送を 1 ブロックのデータのみに制限します。
[7:0]	PLE ウェイト状態	特権モードで、PLE エンジンで実行される PLD 要求の発行レートを制限して、外部メモリ帯域幅の飽和を回避できるようにします。 PLE ウェイト状態は、PLE エンジンで実行される 2 つの PLD 要求間に挿入されるサイクル数を指定します。 PLE ウェイト状態 = 8'b11111111 の場合、PLE エンジンで 256 サイクルごとに 1 つの PLD 要求（1 キャッシュライン）を発行できます。 PLE ウェイト状態 = 8'b000000000 の場合、PLE エンジンで 1 サイクルごとに 1 つの PLD を発行できます。

PLEPCR にアクセスするには、次の命令を使用します。

```
MCR p15, 0, <Rt>, c11, c1, 1; Read PLEPCR
MRC p15, 0, <Rt>, c11, c1, 1; Write PLEPCR
```

4.3.31 CP15 c12 レジスタの概要

CRn が c12 の場合にアクセス可能なシステム制御レジスタを、表 4-30 に示します。

表 4-30 c12 システム制御レジスタ

Op1	CRm	Op2	名前	タイプ	リセット時の値	説明
0	c0	0	VBAR	RW	0x00000000[a]	ベクタベース アドレスレジスタ
		1	MVBAR	RW	-	モニタベクタベース アドレスレジスタ
	c1	0	ISR	RO	0x00000000	割り込みステータスレジスタ
		1	仮想化割り込みレジスタ	RW	0x00000000	*「仮想化割り込みレジスタ」*

a. セキュアバージョンのみが 0 にリセットされます。非セキュアバージョンはソフトウェアでプログラムする必要があります。

4.3.32 仮想化割り込みレジスタ

VIR の特徴は次のとおりです。

目的 保留中の仮想割り込みが存在するかどうかを示します。

使用制限 VIR の使用制限は次のとおりです。

- 特権モードでのみアクセス可能
- セキュア状態でのみアクセス可能

構成 すべての構成で使用可能です。

属性 レジスタの概要については、表 4-30 を参照して下さい。

仮想割り込みは、プロセッサが NS 状態に移行した直後に発行されます。VIR のビット割り当てを、図 4-18 に示します。

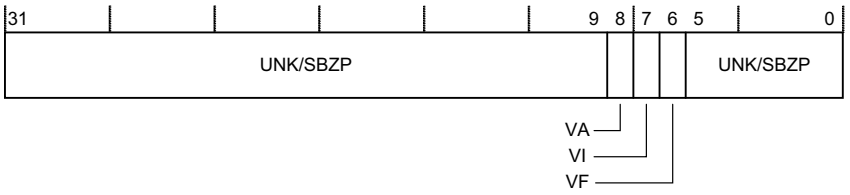


図 4-18 VIR のビット割り当て

仮想化割り込みレジスタのビット割り当てを、表 4-31 に示します。

表 4-31 仮想化割り込みレジスタのビット割り当て

ビット	名前	機能
[31:9]	-	UNK/SBZP
[8]	VA	仮想アボートビット。 セットされている場合、対応するアボートが通常のアボートと同様にソフトウェアに送信されます。仮想アボートは、プロセッサが非セキュア状態のときにのみ発生します。
[7]	VI	仮想 IRQ ビット。 セットされている場合は、対応する IRQ が通常の IRQ と同様にソフトウェアに送信されます。仮想 IRQ は、プロセッサが非セキュア状態のときにのみ発生します。
[6]	VF	仮想 FIQ ビット。 セットされている場合、対応する FIQ が通常の FIQ と同様にソフトウェアに送信されます。仮想 FIQ は、プロセッサが非セキュア状態のときにのみ発生します。
[5:0]	-	UNK/SBZP

VIR にアクセスするには、次の命令を使用します。

```
MRC p15, 0, <Rd>, c12, c1, 1 ; Read Virtualization Interrupt Register
MCR p15, 0, <Rd>, c12, c1, 1 ; Write Virtualization Interrupt Register
```

4.3.33 CP15 c13 レジスタの概要

CRn が c13 の場合にアクセス可能なシステム制御レジスタを、表 4-32 に示します。

表 4-32 c13 システム制御レジスタ

Op1	CRm	Op2	名前	タイプ	リセット時の値	説明
0	c0	0	FCSEIDR	RW	0x00000000	*「非推奨レジスタ」* (ページ 4-2)
		1	CONTEXTIDR	RW	-	コンテキスト ID レジスタ
		2	TPIDRURW	RW[a]	-	ソフトウェアスレッド ID レジスタ
		3	TPIDRURO	RO[b]	-	
		4	TPIDRPRW	RW	-	

a. ユーザモードで RW
b. ユーザモードで RO

4.3.34 CP15 c14、未使用

CRn を c14 に設定してアクセスする CP15 レジスタは存在しません。

4.3.35 CP15 c15 レジスタの概要

CRn が c15 の場合にアクセス可能なシステム制御レジスタを、表 4-33 に示します。

表 4-33 c15 システム制御レジスタ

Op1	CRm	Op2	名前	タイプ	リセット時の値	説明
0	c0	0	電力制御レジスタ	RW[a]	[b]	*「電力制御レジスタ」*
	c1	0	NEON ビジーレジスタ	RO	0x00000000	*「NEON ビジーレジスタ」*（ページ 4-37）
4	c0	0	構成ベースアドレス	RO[c]	[d]	*「構成ベースアドレス レジスタ」*（ページ 4-38）
5	c4	2	読み出し用のロックダウン TLB エントリの選択	WO[e]	-	*「TLB ロックダウン操作」*（ページ 4-39）
		4	書き込み用のロックダウン TLB エントリの選択	WO[e]	-	
	c5	2	メイン TLB VA レジスタ	RW[e]	-	
	c6	2	メイン TLB PA レジスタ	RW[e]	-	
	c7	2	メイン TLB 属性レジスタ	RW	-	

a. セキュア状態では RW です。非セキュア状態では読み出し専用です。
b. リセット時の値は、**MAXCLKLATENCY[2:0]** の値によって異なります。*「構成信号」*（ページ A-5）を参照して下さい。
c. セキュア特権モードでは RW で、非セキュア状態とユーザセキュア状態では RO です。
d. Cortex-A9 ユニプロセッサ実装では、構成ベースアドレスが 0 に設定されます。
Cortex-A9 MPCore 実装では、構成ベースアドレスが **PERIPHBASE[31:13]** にリセットされるため、ソフトウェアでプライベートメモリ領域の位置を決定できます。
e. 非セキュア状態ではアクセス不可です。

4.3.36 電力制御レジスタ

電力制御レジスタの特徴は次のとおりです。

目的 次の設定を可能にします。
- Cortex-A9 プロセッサの実装のクロックレイテンシ
- 動的クロックゲート

使用制限
- セキュア状態では読み出し / 書き込みレジスタ
- 非セキュア状態では読み出し専用レジスタ

構成 すべての構成で使用可能です。

属性 レジスタの概要については、表 4-33 を参照して下さい。

電力制御レジスタのビット割り当てを、図 4-19 に示します。

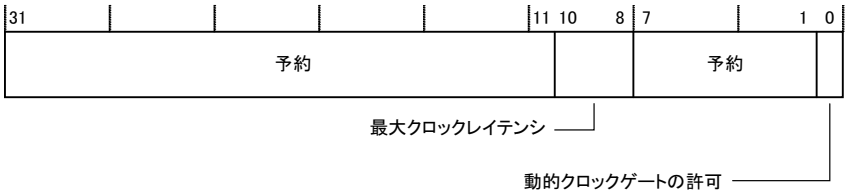


図 4-19 電力制御レジスタのビット割り当て

電力制御レジスタのビット割り当てを、表 4-34 に示します。

表 4-34 電力制御レジスタのビット割り当て

ビット	名前	機能
[31:11]	-	予約
[10:8]	最大クロックレイテンシ	リセット終了時点で **MAXCLKLATENCY** ピンに現れた値をサンプリングします。この値は実装固有のパラメータを反映しているため、ソフトウェアで変更しないことをお勧めします。
[7:1]	-	予約
[0]	動的クロックゲートイネーブル	リセット時には不可能な状態です。

電力制御レジスタにアクセスするには、次の命令を使用します。

```
MRC p15,0,<Rd>,c15,c0,0; Read Power Control Register
MCR p15,0,<Rd>,c15,c0,0; Write Power Control Register
```

4.3.37 NEON ビジーレジスタ

NEON ビジーレジスタの特徴は次のとおりです。

目的 ソフトウェアで、NEON 命令が実行中かどうかを判断できるようにします。

使用制限
- セキュア状態では読み出し専用レジスタ
- 非セキュア状態では読み出し専用レジスタ

構成 すべての構成で使用可能です。

属性 レジスタの概要については、表 4-33 （ページ 4-36）を参照して下さい。

NEON ビジーレジスタのビット割り当てを、図 4-20 に示します。

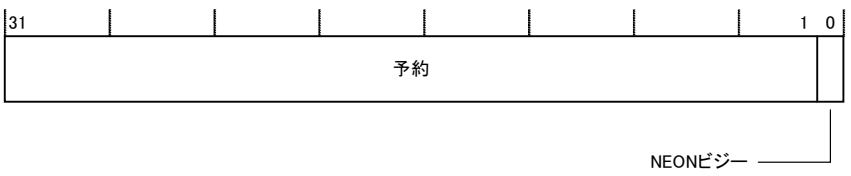


図 4-20 NEON ビジーレジスタのビット割り当て

NEON ビジーレジスタのビット割り当てを、表 4-35 に示します。

表 4-35 NEON ビジーレジスタのビット割り当て

ビット	名前	機能
[31:1]	-	予約
[0]	NEON ビジー	ソフトウェアは、このビットを使用して、NEON 命令が実行中かどうかを判断できます。NEON パイプラインまたはコアパイプライン内に NEON 命令が存在する場合、このビットが 1 にセットされます。

NEON ビジーレジスタにアクセスするには、次の命令を使用します。

```
MRC p15,0,<Rd>,c15,c1,0; Read NEON busy Register
```

4.3.38 構成ベースアドレス レジスタ

構成ベースアドレス レジスタの特徴は次のとおりです。

目的 リセット時に物理ベースアドレス値を取得します。

使用制限 構成ベースアドレス レジスタの使用制限は次のとおりです。

- セキュア特権モードでは読み出し / 書き込み
- 非セキュア状態では読み出し専用
- ユーザモードでは読み出し専用

構成 Cortex-A9 ユニプロセッサ実装では、ベースアドレスが 0 に設定されます。

Cortex-A9 MPCore 実装では、**PERIPHBASE[31:13]** にリセットされるため、ソフトウェアでプライベートメモリ領域の位置を決定できます。

属性 レジスタの概要については、表 4-33 （ページ 4-36）を参照して下さい。

構成ベースアドレス レジスタのビット割り当てを、図 4-21 に示します。

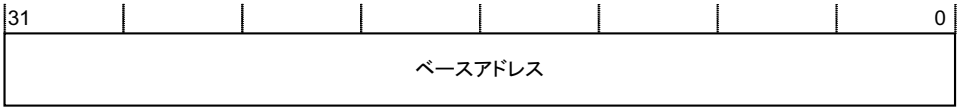


図 4-21 構成ベースアドレス レジスタのビット割り当て

構成ベースアドレス レジスタにアクセスするには、次の命令を使用します。

```
MRC p15,4,<Rd>,c15,c0,0; Read Configuration Base Address Register
MCR p15,4,<Rd>,c15,c0,0; Write Configuration Base Address Register
```

4.3.39 TLB ロックダウン操作

TLB ロックダウン操作を使用すれば、TLB のロックダウンエントリを保存または復元できます。定義済みの TLB ロックダウン操作を、表 4-36 に示します。

表 4-36 TLB ロックダウン操作

説明	データ	命令
読み出し用のロックダウン TLB エントリの選択	メイン TLB インデクス	`MCR p15,5,<Rd>,c15,c4,2`
書き込み用のロックダウン TLB エントリの選択	メイン TLB インデクス	`MCR p15,5,<Rd>,c15,c4,4`
ロックダウン TLB VA レジスタの読み出し	データ	`MRC p15,5,<Rd>,c15,c5,2`
ロックダウン TLB VA レジスタの書き込み	データ	`MCR p15,5,<Rd>,c15,c5,2`
ロックダウン TLB PA レジスタの読み出し	データ	`MRC p15,5,<Rd>,c15,c6,2`
ロックダウン TLB PA レジスタの書き込み	データ	`MCR p15,5,<Rd>,c15,c6,2`
ロックダウン TLB 属性レジスタの読み出し	データ	`MRC p15,5,<Rd>,c15,c7,2`
ロックダウン TLB 属性レジスタの書き込み	データ	`MCR p15,5,<Rd>,c15,c7,2`

読み出し用のロックダウン TLB エントリの選択操作は、ロックダウン TLB の VA/PA/ 属性の読み出し操作によってデータが読み出されるエントリを選択するために使用します。書き込み用のロックダウン TLB エントリの選択操作は、ロックダウン TLB の VA/PA/ 属性の書き込み操作によってデータが書き込まれるエントリを選択するために使用します。TLB PA レジスタは、TLB ロックダウンレジスタにアクセスするときに、最後に書き込み / 読み出しを行うレジスタにする必要があります。ロックダウン TLB エントリへのアクセスに使用するインデクスレジスタのビット割り当てを、図 4-22 に示します。

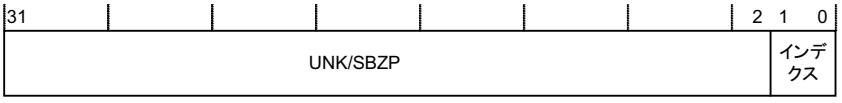


図 4-22 ロックダウン TLB インデクスのビット割り当て

TLB VA レジスタのビット割り当てを、図 4-23 に示します。

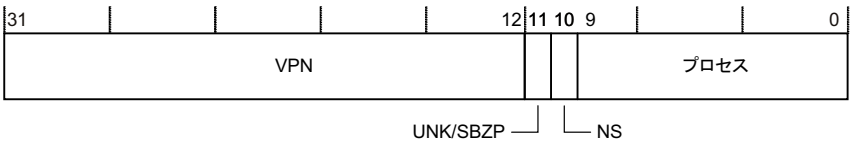


図 4-23 TLB VA レジスタのビット割り当て

TLB VA レジスタのビット割り当てを、表 4-37 に示します。

表 4-37 TLB VA レジスタのビット割り当て

ビット	名前	機能
[31:12]	VPN	仮想ページ番号。 テーブルのサイズが原因で、ページテーブル変換の一部として変換されない仮想ページ番号のビットは、読み出し時は予測不能な値になり、書き込み時は SBZ です。
[11]	-	UNK/SBZP
[10]	NS	NS ビット
[9:0]	プロセス	メモリ空間識別子

メモリ空間識別子のビット割り当てを、図 4-24 に示します。

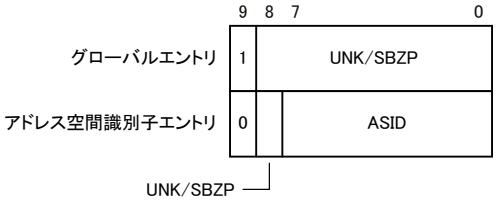


図 4-24 メモリ空間識別子の形式

TLB PA レジスタのビット割り当てを、図 4-25 に示します。

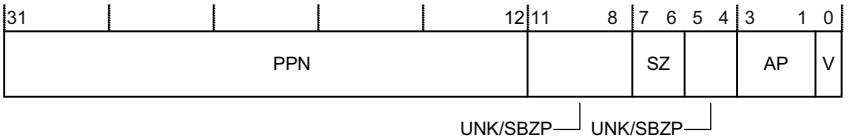


図 4-25 TLB PA レジスタのビット割り当て

TLB PA レジスタのビットの機能を、表 4-38 に示します。

表 4-38 TLB PA レジスタのビット割り当て

ビット	名前	機能
[31:12]	PPN	物理ページ番号。 ページテーブル変換の一部として変換されない物理ページ番号のビットは、読み出し時は予測不能で、書き込み時は SBZP です。
[11:8]	-	UNK/SBZP
[7:6]	SZ	領域サイズ。 b00 = 16MB のスーパーセクション b01 = 4KB のページ b10 = 64KB のページ b11 = 1MB のセクション 他の値はすべて予約されています。

表 4-38 TLB PA レジスタのビット割り当て （続き）

ビット	名前	機能
[5:4]	-	UNK/SBZP
[3:1]	AP	アクセス許可。 b000 = すべてのアクセスで許可フォールトが生成されます。 b001 = スーパバイザアクセス専用、ユーザアクセスでフォールトが生成されます。 b010 = スーパバイザ読み出し / 書き込みアクセス、ユーザ書き込みアクセスでフォールトが生成されます。 b011 = フルアクセス、フォールトは生成されません。 b100 = 予約 b101 = スーパバイザ読み出し専用 b110 = スーパバイザ / ユーザ読み出し専用 b111 = スーパバイザ / ユーザ読み出し専用
[0]	V	値ビット。 このエントリがロックされ、有効であることを示します。

TLB 属性レジスタのビット割り当てを、図 4-26 に示します。

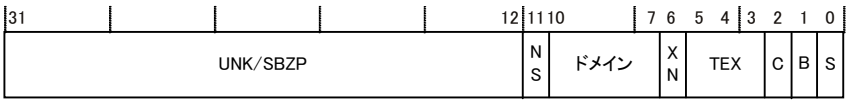


図 4-26 メイン TLB 属性レジスタのビット割り当て

TLB 属性レジスタのビット割り当てを、表 4-39 に示します。Cortex-A9 プロセッサはサブページをサポートしていません。

表 4-39 TLB 属性レジスタのビット割り当て

ビット	名前	機能
[31:12]	-	UNK/SBZP
[11]	NS	非セキュア記述
[10:7]	ドメイン	TLB エントリのドメイン番号
[6]	XN	実行不可属性
[5:3]	TEX	領域タイプのエンコード。詳細については、*『ARM アーキテクチャリファレンスマニュアル』*を参照して下さい。
[2:1]	CB	
[0]	S	共有属性

ASID の一致による TLB エントリの無効化

これは、指定された*アドレス空間識別子* (ASID) の値と一致するすべての TLB エントリを無効化する、単一の割り込み可能操作です。この機能は、ロックされたエントリの無効化を行います。グローバルとしてマークされているエントリは、この機能では無効化されません。

Cortex-A9 プロセッサでは、この操作の完了に数サイクルかかります。命令は割り込み可能です。割り込みが発生すると、r14 の状態が、MCR 命令が実行されなかったことを示すように設定されます。そのため、r14 は MCR のアドレスに 4 を加えた値を指します。その後で、割り込みルーチンは自動的に、MCR 命令の位置から再開されま

す。この操作が割り込まれ、しばらくして再開された場合は、指定された ASID を使用する割り込みによって TLB にフェッチされたすべてのエントリが、再開された無効化によって無効にされます。

第 5 章
Jazelle DBX レジスタ

本章では、CP14 コプロセッサの概要と、デバッグ以外の用途について説明します。本章は次のセクションから構成されています。

5.1 コプロセッサ CP14 について

コプロセッサ C14 のデバッグ以外の用途は、Java バイトコードのハードウェアアクセラレーションをサポートすることです。

5.2 CP14 Jazelle レジスタの概要

Jazelle 拡張機能の Cortex-A9 実装では、次の規則が適用されます。

- Jazelle 状態がサポートされます。
- BXJ 命令で Jazelle 状態に移行します。

CP14 Jazelle レジスタを、表 5-1 に示します。すべての Jazelle レジスタアクセスにおいて、CRm と Op2 は 0 です。すべての Jazelle レジスタは 32 ビット幅です。

表 5-1 CP14 Jazelle レジスタの概要

Op1	CRn	名前	タイプ	リセット時の値	ページ
7	0	JazelleID レジスタ (JIDR)	RW[a]	0xF4100168	(ページ 5-4)
7	1	Jazelle OS 制御レジスタ (JOSCR)	RW	-	(ページ 5-5)
7	2	Jazelle メイン構成レジスタ (JMCR)	RW	-	(ページ 5-6)
7	3	Jazelle パラメータレジスタ	RW	-	(ページ 5-7)
7	4	Jazelle 構成可能オペコード変換テーブルレジスタ	WO	-	(ページ 5-8)

a. 書き込み操作の影響については、*「JIDR の書き込み操作」*（ページ 5-5）を参照して下さい。

Jazelle 拡張機能の詳細については、*『ARM アーキテクチャ リファレンスマニュアル』*を参照して下さい。

5.3 CP14 Jazelle レジスタの詳細

次に示すセクションでは、表 5-1 （ページ 5-3）での番号順に、CP14 Jazelle DBX レジスタの内容を説明します。

- *「Jazelle ID レジスタ」*
- *「Jazelle オペレーティングシステム制御レジスタ」*（ページ 5-5）
- *「Jazelle メイン構成レジスタ」*（ページ 5-6）
- *「Jazelle パラメータレジスタ」*（ページ 5-7）
- *「Jazelle 構成可能オペコード変換テーブルレジスタ」*（ページ 5-8）

5.3.1 Jazelle ID レジスタ

JIDR の特徴は次のとおりです。

目的 ソフトウェアで、プロセッサから提供される Jazelle 拡張機能の実装を判断できるようにします。

使用制限 JIDR の使用制限は次のとおりです。

- 特権モードでアクセス可能
- CD ビットがクリアされている場合は、ユーザモードでもアクセス可能です。*「Jazelle オペレーティングシステム制御レジスタ」*（ページ 5-5）を参照して下さい。

構成 すべての構成で使用可能です。

属性 レジスタの概要については、表 5-1 （ページ 5-3）を参照して下さい。

JIDR のビット割り当てを、図 5-1 に示します。

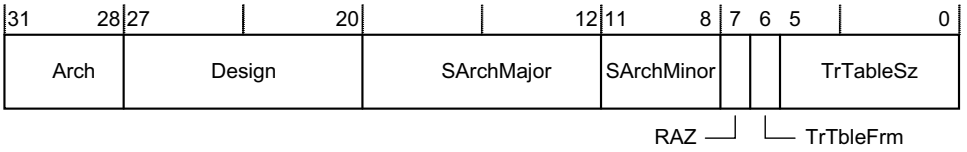


図 5-1 JIDR のビット割り当て

JIDR のビット割り当てを、表 5-2 に示します。

表 5-2 JIDR のビット割り当て

ビット	名前	機能
[31:28]	Arch	メイン ID レジスタと同じアーキテクチャコードが使用されます。
[27:20]	Design	サブアーキテクチャ設計者の実装者コードが格納されます。
[19:12]	SArchMajor	サブアーキテクチャコード
[11:8]	SArchMinor	サブアーキテクチャ マイナーコード
[7]	-	RAZ
[6]	TrTbleFrm	Jazelle 構成可能オペコード変換テーブルレジスタの形式を示します。
[5:0]	TrTbleSz	Jazelle 構成可能オペコード変換テーブルレジスタのサイズを示します。

JIDR にアクセスするには、次の命令を使用します。

```
MRC p14, 7, <Rd>, c0, c0, 0; Read Jazelle Identity Register
```

JIDR の書き込み操作

JIDR に書き込むと、変換テーブルがクリアされます。この操作によって、構成可能オペコードがすべて、ソフトウェアのみで実行されるようになります。*「Jazelle 構成可能オペコード変換テーブルレジスタ」*（ページ 5-8）を参照して下さい。

5.3.2 Jazelle オペレーティングシステム制御レジスタ

JOSCR の特徴は次のとおりです。

目的 オペレーティングシステムで、Jazelle 拡張機能ハードウェアへのアクセスを制御できるようにします。

使用制限 JOSCR の使用制限は次のとおりです。

- 特権モードでのみアクセス可能
- リセット後は 0 に設定され、特権モードで書き込む必要があります。

構成 すべての構成で使用可能です。

属性 レジスタの概要については、表 5-1 （ページ 5-3）を参照して下さい。

JOSCR のビット割り当てを、図 5-2 に示します。

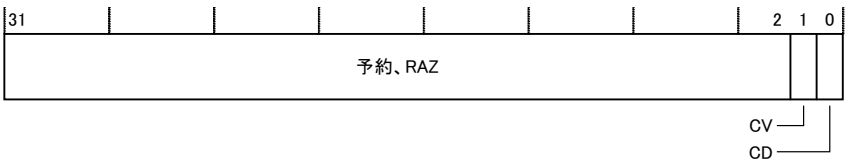


図 5-2 JOSCR のビット割り当て

JOSCR のビット割り当てを、表 5-3 に示します。

表 5-3 JOSCR のビット割り当て

ビット	名前	機能
[31:2]	-	予約、RAZ
[1]	CV	構成有効ビット。 0 = Jazelle 構成が無効です。Jazelle ハードウェアが稼働状態で Jazelle 状態への移行を試みると、次の動作が行われます。 • 構成無効 Jazelle 例外が生成されます。 • このビットがセットされ、Jazelle 構成が有効としてマークされます。 1 = Jazelle 構成が有効です。Jazelle ハードウェアが稼働状態であれば、Jazelle 状態への移行が成功します。 例外が発生すると、CV ビットが自動的にクリアされます。
[0]	CD	構成不可能ビット。 0 = ユーザモードでの Jazelle 構成が可能です。 • JIDR の読み出しが成功します。 • その他の Jazelle 構成レジスタを読み出すと、未定義命令例外が生成されます。 • JOSCR に書き込むと、未定義命令例外が生成されます。 • その他の Jazelle 構成レジスタへの書き込みは成功します。 1 = ユーザモードでの Jazelle 構成は不可能です。 • Jazelle 構成レジスタを読み出すと、未定義命令例外が生成されます。 • Jazelle 構成レジスタに書き込むと、未定義命令例外が生成されます。

JOSCR にアクセスするには、次の命令を使用します。

```
MRC p14, 7, <Rd>, c1, c0, 0; Read JOSCR
MCR p14, 7. <Rd>, c1, c0, 0; Write JOSCR
```

5.3.3 Jazelle メイン構成レジスタ

JMCR の特徴は次のとおりです。

目的 Jazelle ハードウェアの構成とその動作を示します。

使用制限 特権モードでのみアクセス可能です。

構成 すべての構成で使用可能です。

属性 レジスタの概要については、表 5-1 （ページ 5-3）を参照して下さい。

JMCR のビット割り当てを、図 5-3 に示します。

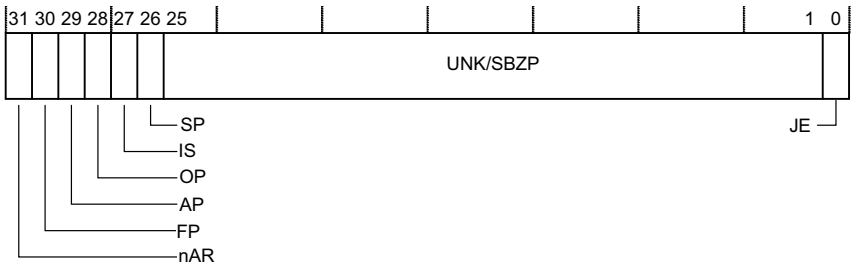


図 5-3 JMCR のビット割り当て

JMCR のビット割り当てを、表 5-4 に示します。

表 5-4 JMCR のビット割り当て

ビット	名前	機能
[31]	nAR	*非配列操作* (nAR) ビット。 0 = 実装されていれば、ハードウェアで配列操作を実行します。実装されていない場合は、VM 実装テーブル内の適切なハンドラを呼び出します。 1 = すべての配列操作を、VM 実装テーブル内の適切なハンドラを呼び出して実行します。
[30]	FP	FP ビットは、Jazelle ハードウェアによる JVM 浮動小数点オペコードの実行方法を制御します。 0 = すべての JVM 浮動小数点オペコードを、VM 実装テーブル内の適切なハンドラを呼び出して実行します。 1 = 可能であれば、VFP 命令を発行して JVM 浮動小数点オペコードを実行します。 それ以外の場合は、VM 実装テーブル内の適切なハンドラを呼び出します。 この実装では、FP は 0 に設定され、読み出し専用です。
[29]	AP	*配列ポインタ* (AP) ビットは、Jazelle ハードウェアによるオペランドスタック上の配列参照の処理方法を制御します。 0 = 配列参照がハンドルとして処理されます。 1 = 配列参照がポインタとして処理されます。
[28]	OP	*オブジェクトポインタ* (OP) ビットは、Jazelle ハードウェアによるオペランドスタック上のオブジェクト参照の処理方法を制御します。 0 = オブジェクト参照がハンドルとして処理されます。 1 = オブジェクト参照がポインタとして処理されます。
[27]	IS	*インデクスサイズ* (IS) ビットは、クイックオブジェクト フィールドアクセスに関連付けられたインデクスのサイズを示します。 0 = クイックオブジェクト フィールドインデクスは 8 ビットです。 1 = クイックオブジェクト フィールドインデクスは 16 ビットです。
[26]	SP	*静的ポインタ* (SP) ビットは、Jazelle ハードウェアによる静的参照の処理方法を制御します。 0 = 静的参照がハンドルとして処理されます。 1 = 静的参照がポインタとして処理されます。
[25:1]	-	UNK/SBZP
[0]	JE	*Jazelle イネーブル* (JE) ビットは、Jazelle ハードウェアが稼働するかどうかを制御します。 0 = Jazelle ハードウェアは非稼働状態です。 • BXJ 命令が BX 命令と同様に動作します。 • CPSR 内の J ビットをセットすると、Jazelle 非稼働 Jazelle 例外が発生します。 1 = Jazelle ハードウェアが稼働状態です。 • BXJ 命令で Jazelle 状態に移行します。 • CPSR の J ビットをセットすると、Jazelle 状態に移行します。

JMCR にアクセスするには、次の命令を使用します。

```
MRC p14, 7, <Rd>, c2, c0, 0; Read JMCR
MCR p14, 7. <Rd>, c2, c0, 0; Write JMCR
```

5.3.4 Jazelle パラメータレジスタ

Jazelle パラメータレジスタの特徴は次のとおりです。

目的 Jazelle ハードウェアの動作方法を構成するパラメータを示します。

使用制限 特権モードでのみアクセス可能です。

構成 すべての構成で使用可能です。

属性 レジスタの概要については、表 5-1 （ページ 5-3）を参照して下さい。

Jazelle パラメータレジスタのビット割り当てを、図 5-4 に示します。

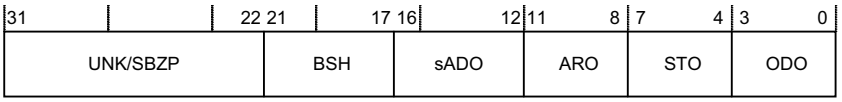


図 5-4 Jazelle パラメータレジスタのビット割り当て

Jazelle パラメータレジスタのビット割り当てを、表 5-5 に示します。

表 5-5 Jazelle パラメータレジスタのビット割り当て

ビット	名前	機能
[31:22]	-	UNK/SBZP
[21:17]	BSH	*バウンドシフト* (BSH) ビットには、配列記述子ワード内の配列バウンド （配列内の項目数）のオフセットがビット単位で格納されます。
[16:12]	sADO	*符号付き配列記述子オフセット* (sADO) ビットには、配列参照から配列記述子ワードまでのオフセットが、ワード単位で格納されます。オフセットは、符号絶対値符号付き数量です。 • ビット [16] はオフセットの符号を示します。オフセットは、このビットがクリアされている場合は正で、このビットがセットされている場合は負です。 • ビット [15:12] は、オフセットの絶対値を示します。
[11:8]	ARO	*配列参照オフセット* (ARO) ビットには、配列参照から、配列データまたは配列データポインタまでのオフセットが、ワード単位で格納されます。
[7:4]	STO	*静的オフセット* (STO) ビットには、静的参照から、静的または静的ポインタまでのオフセットが、ワード単位で格納されます。
[3:0]	ODO	*オブジェクト記述子オフセット* (ODO) ビットには、オブジェクトデータ ブロックのベースからフィールドまでのオフセットが、ワード単位で格納されます。

Jazelle パラメータレジスタにアクセスするには、次の命令を使用します。

```
MRC p14, 7, <Rd>, c3, c0, 0; Read Jazelle Parameters Register
MCR p14, 7. <Rd>, c3, c0, 0; Write Jazelle Parameters Register
```

5.3.5 Jazelle 構成可能オペコード変換テーブルレジスタ

Jazelle 構成可能オペコード変換テーブルレジスタの特徴は次のとおりです。

目的 0xCB ～ 0xFD の範囲の構成可能なオペコードと、Jazelle ハードウェアで提供される操作との間の変換を実行します。

使用制限 特権モードでのみアクセス可能です。

構成 すべての構成で使用可能です。

属性 レジスタの概要については、表 5-1 （ページ 5-3）を参照して下さい。

Jazelle 構成可能オペコード変換テーブルレジスタのビット割り当てを、図 5-5 （ページ 5-9）に示します。

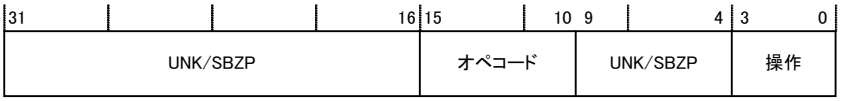


図 5-5 Jazelle 構成可能オペコード変換テーブルレジスタのビット割り当て

Jazelle 構成可能オペコード変換テーブルレジスタのビット割り当てを、表 5-6 に示します。

表 5-6 Jazelle 構成可能オペコード変換テーブルレジスタのビット割り当て

ビット	名前	機能
[31:16]	-	UNK/SBZP
[15:10]	オペコード	構成可能オペコードの下位ビットが格納されます。
[9:4]	-	UNK/SBZP
[3:0]	操作	操作 0x0 ～ 0x9 のコードが格納されます。

このレジスタにアクセスするには、次の命令を使用します。

```
MRC p14, 7, <Rd>, c4, c0, 0;  Read Jazelle Configurable Opcode Translation
                              Table Register

MCR p14, 7. <Rd>, c4, c0, 0;  Write Jazelle Configurable Opcode Translation
                              Table Register
```

第 6 章
メモリ管理ユニット

本章では、MMU について説明します。本章は次のセクションから構成されています。

6.1 MMU について

MMU は、レベル 1 およびレベル 2 メモリシステムと連携して、仮想アドレスを物理アドレスに変換します。また、外部メモリとのアクセスも制御します。

仮想メモリシステム アーキテクチャバージョン 7(VMSAv7) には次のような機能があります。

- 4KB、64KB、1MB、16MB をサポートするページテーブル エントリ
- 16 個のドメイン
- コンテキストスイッチ TLB フラッシュに関する要件を排除するための、グローバルおよびアドレス空間識別子
- アクセス許可チェック機能の拡張

VMSAv7 アーキテクチャの完全な説明については、*『ARM アーキテクチャ リファレンスマニュアル』* を参照して下さい。

プロセッサは、アドレス変換とアクセス許可チェックを提供する、セキュリティ拡張機能およびマルチプロセッサ拡張機能により拡張された、ARMv7-A MMU を実装しています。MMU は、メインメモリの変換テーブルにアクセスする、テーブルウォーク ハードウェアを制御します。MMU の使用により、一連の仮想 / 物理アドレスマッピングとメモリ属性による、きめ細かなメモリシステム制御が可能になります。

注

VMSAv7 では、レベル 1 記述子形式ページテーブル ベースアドレスのビット 9 は実装定義です。Cortex-A9 プロセッサ設計では、このビットは使用されません。

MMU には次のような機能があります。

- 命令側マイクロ TLB
  - 32 個の完全アソシエイティブエントリ
- データ側マイクロ TLB
  - 32 個の完全アソシエイティブエントリ
- 統一メイン TLB
  - 64 エントリ（2 × 32 エントリ）TLB と、128 エントリ（2 × 64 エントリ）TLB。いずれも、統一 2 ウェイアソシエイティブです。
  - エントリ単位ロックモデルを使用した、4 つのロック可能エントリ
  - レベル 1 データキャッシュ内でルックアップを実行する、ハードウェア ページテーブル ウォークをサポートします。

6.1.1 メモリ管理ユニット

MMU は次の操作を実行します。

- 仮想アドレスと ASID のチェック
- ドメインアクセス許可のチェック
- メモリ属性のチェック
- 仮想アドレスから物理アドレスへの変換
- 4 つのページ（領域）サイズのサポート
- キャッシュまたは外部メモリへのアクセスのマッピング
- ハードウェアとソフトウェアでの TLB ロード

ドメイン

Cortex-A9 プロセッサは、16 個のアクセスドメインをサポートします。

TLB

Cortex-A9 プロセッサは 2 レベルの TLB 構造を実装しています。メイン TLB の 4 つのエントリがロック可能です。

ASID

メイン TLB エントリは、グローバルにすることも、アドレス空間識別子 (ASID) を使用して特定のプロセスまたはアプリケーションに関連付けることもできます。ASID の使用により、コンテキストスイッチ中に TLB エントリの内容が保持されるため、後でロードし直す必要がなくなります。*「ASID の一致による TLB エントリの無効化」*（ページ 4-41）を参照して下さい。

システム制御コプロセッサ

TLB の保守および構成操作は、コアに統合されている専用コプロセッサの CP15 を経由して制御されます。このコプロセッサは、レベル 1 メモリシステムを構成するための標準的な機構を提供します。

6.2 TLB の構成

次に示すセクションでは、TLB の構成について説明します。

- 「マイクロ TLB」
- 「メイン TLB」

6.2.1 マイクロ TLB

ページテーブル情報のレベル 1 キャッシュは、命令側とデータ側のそれぞれに実装される 32 エントリのマイクロ TLB です。これらのブロックは、単一 CLK サイクルで仮想アドレスの、完全アソシエイティブルックアップを提供します。

マイクロ TLB は、アドレスを比較するために物理アドレスをキャッシュに返し、保護属性をチェックしてプリフェッチアボートまたはデータアボートを通知します。

メイン TLB に関連するすべての操作は、命令とデータの両方のマイクロ TLB に影響を与え、それらのフラッシュを引き起こします。同様に、コンテキスト ID レジスタを変更すると、マイクロ TLB がフラッシュされます。

6.2.2 メイン TLB

メイン TLB は、マイクロ TLB からのミスを捕捉します。また、ロック可能な変換エントリのソースの集中化を可能にします。

メイン TLB へのアクセスに必要なサイクル数は、マイクロ TLB 間の要求の競合や、他の実装固有の要因によって異なります。メイン TLB のロック可能領域内にあるエントリは、エントリ単位でロック可能です。ロック可能領域にロックされているエントリがなければ、ロックされていないエントリを割り当てて、メイン TLB ストレージの合計サイズを増やすことができます。

メイン TLB は、次の要素の組み合わせとして実装されます。

- 4 要素の完全アソシエイティブな、ロック可能配列
- 2 × 32 または 2 × 64 エントリの、2 ウェイアソシエイティブ構造

TLB 一致プロセス

各 TLB エントリには、仮想アドレス、ページサイズ、物理アドレス、一連のメモリプロパティが格納されます。各エントリは、特定のアプリケーション空間に関連付けられたもの、またはすべてのアプリケーション空間に対してグローバルなものとしてマークされます。CONTEXIDR は、現在選択されているアプリケーション空間を示します。修飾仮想アドレスのビット [31:N] が一致すれば、TLB エントリは一致します。ここで、N は TLB エントリのページサイズの $\log_2$ です。このエントリは、グローバルとして、または現在の ASID と一致する ASID としてマークされます。

TLB エントリは、次の条件が真の場合に一致します。

- エントリの仮想アドレスが、要求されたアドレスの仮想アドレスと一致する。
- エントリの非セキュア TLB ID (NSTID) が、セキュア状態または非セキュア状態の MMU 要求と一致する。
- エントリの ASID が現在の ASID と一致するか、グローバルである。

オペレーティングシステムは常に、2 つ以上の TLB エントリが一致しないようにする必要があります。

スーパーセクション、セクション、ラージページのサポートによって、TLB の単一エントリのみを使用して、メモリの大きな領域のマッピングが可能です。アドレスのマッピングが TLB に見つからない場合は、ハードウェアで自動的に変換テーブルが読み出され、マッピングが TLB 内に配置されます。

TLB ロックダウン

TLB は、*『ARM アーキテクチャ リファレンスマニュアル』*に記載されているように、TLB のエントリ単位ロックモデルをサポートします。詳細については、*「TLB ロックダウン操作」*（ページ 4-39）を参照して下さい。

6.3 メモリアクセス シーケンス

プロセッサがメモリアクセスを生成すると、MMU で次の処理が実行されます。

1. 関連する命令またはデータマイクロ TLB で、要求された仮想アドレス、現在の ASID、現在のセキュリティ状態のルックアップを実行します。
2. マイクロ TLB でミスが発生した場合は、メイン TLB で、要求された仮想アドレス、現在の ASID、現在のセキュリティ状態のルックアップを実行します。
3. メイン TLB でミスが発生した場合は、ハードウェア変換テーブルウォークを実行します。

変換テーブルベース レジスタの IRGN ビットをセットすると、キャッシュ可能領域でハードウェア変換テーブルウォークを実行するように MMU を構成できます。IRGN ビットのエンコードがライトバックの場合は、レベル 1 データキャッシュのルックアップが実行され、データキャッシュからデータが読み出されます。IRGN ビットのエンコードがライトスルーまたはキャッシュ不可の場合は、外部メモリへのアクセスが実行されます。

MMU の処理の結果、指定された仮想アドレスについて、一致する*非セキュア TLB ID*(NSTID) を持つ、グローバルマッピングまたは現在選択されている ASID のマッピングが、TLB に見つからない可能性があります。この場合、TTB 制御レジスタの PD0 または PD1 ビットにより変換テーブルウォークが可能になっていれば、ハードウェアで変換テーブルウォークが実行されます。変換テーブルウォークが不可能な場合は、プロセッサがセクション変換フォールトを返します。

MMU で一致する TLB エントリが見つかった場合は、次のようにそのエントリの情報が使用されます。

1. アクセス許可ビットとドメインによって、アクセスが可能かどうかが判断されます。一致するエントリが許可チェックに合格しなかった場合は、MMU からメモリアボートが通知されます。アクセス許可ビットの説明、アボートのタイプと優先度に関する説明、および IFSR と*データフォールト ステータスレジスタ*(DFSR) の説明については、*『ARM アーキテクチャ リファレンスマニュアル』*を参照して下さい。
2. TLB エントリと CP15 c10 再マップレジスタの両方で指定されているメモリ領域属性によって、キャッシュとライトバッファが制御され、アクセスが次のどれであるかが決定されます。
   - セキュアまたは非セキュア
   - 共有または非共有
   - ノーマルメモリ、デバイス、またはストロングリオーダ
3. MMU は、メモリアクセスを行うため、仮想アドレスを物理アドレスに変換します。

MMU で一致するエントリが見つからなかった場合は、ハードウェア テーブル ウォークが実行されます。

6.4 MMU の稼働または非稼働

『*ARM アーキテクチャ リファレンスマニュアル*』に記載されている方法で、MMU を稼働または非稼働にできます。

6.5 外部アボート

外部メモリエラーは、MMU で検出されたものではなく、メモリシステムで発生したものとして定義されます。外部メモリエラーの発生は非常に稀であると想定されます。外部アボートは、要求がプロセッサの外部に出力されたときに、AXI インタフェースでフラグが付けられたエラーによって引き起こされます。セキュア構成レジスタの EA ビットをセットして、外部アボートがモニタモードにトラップするように構成できます。

6.5.1 データの読み出し / 書き込み時の外部アボート

データの読み出し / 書き込み時に外部で生成されたエラーは、非同期な場合があります。これは、このようなアボートでアボートハンドラが開始されたとき、例外を引き起こした命令のアドレスが r14_abt に保存されていない可能性があることを意味します。

非同期アボートが発生した場合、DFAR は予測不能です。

複数ロード / ストア操作の場合、DFAR にキャプチャされるアドレスは、同期外部アボートを生成したアドレスです。

6.5.2 同期アボートと非同期アボート

フォールトタイプを判断するために、データアボートの場合は DFSR を、命令アボートの場合は IFSR を読み出します。

プロセッサは、ソフトウェアの互換性を維持する目的でのみ、補助フォールトステータス レジスタをサポートしています。アボートが生成されても、プロセッサがこのレジスタを変更することはありません。

第 7 章
レベル 1 メモリシステム

本章では、レベル 1 メモリシステムについて説明します。本章は次のセクションから構成されています。

7.1 レベル 1 メモリシステムについて

レベル 1 メモリシステムには次のような特徴と機能があります。

- ライン長が 32 バイト固定の、独立した命令キャッシュおよびデータキャッシュ
- メモリシステム全体にわたる 64 ビットのデータパス
- 4 つのメモリページ サイズのサポート
- 外部メモリシステムのメモリ属性のエクスポート
- セキュリティ拡張機能のサポート

レベル 1 メモリシステムのデータ側には、次のような特徴と機能があります。

- 2 つの 32 バイトラインフィル バッファと、1 つの 32 バイト退出バッファ
- 4 エントリの 64 ビット結合ストアバッファ

注

これらを使用する前に、命令キャッシュ、データキャッシュ、TLB、BTAC を無効化する必要があります。メイン TLB は無効化する必要がありませんが、安全上の理由から推奨されます。これによって、将来のリビジョンのプロセッサとの互換性が保証されます。

7.1.1 メモリシステム

ここでは、次のトピックについて説明します。

- *「キャッシュ機能」*
- *「ストアバッファ」*（ページ 7-3）

キャッシュ機能

Cortex-A9 プロセッサには、命令キャッシュとデータキャッシュが別々に存在します。これらのキャッシュには、次のような特徴があります。

- キャッシュ単位で無効にすることができます。*「システム制御レジスタ」*（ページ 4-15）を参照して下さい。
- キャッシュ置換方式は、疑似ラウンドロビンと疑似ランダムのどちらかです。
- どちらのキャッシュも 4 ウェイセットアソシエイティブです。
- キャッシュラインの長さは 8 ワードです。
- キャッシュミス時、重要なワードから先にキャッシュフィルが行われます。
- 実装時に、命令キャッシュとデータキャッシュを別々に 16KB、32KB、または 64KB のサイズに構成できます。
- 電力消費を抑えるため、多くのキャッシュ操作がシーケンシャルに実行される特性を活かして、キャッシュ全体の読み出し回数が削減されます。キャッシュ読み出しが前のキャッシュ読み出しからシーケンシャルに行われ、かつ同じキャッシュラインが読み出される場合、前に読み出したデータ RAM セットのみがアクセスされます。

命令キャッシュの特徴

命令キャッシュには、仮想的にインデクスが付けられ、物理的にタグが付けられます。

データキャッシュの特徴

データキャッシュには、物理的にインデクスが付けられ、物理的にタグが付けられます。

データキャッシュの読み出しミスと書き込みミスはいずれも非ブロッキングで、未解決データキャッシュ読み出しミスが最高 4 回まで、未解決データキャッシュ書き込みミスが最高 4 回までサポートされます。

ストアバッファ

Cortex-A9 CPU は、データ結合機能を持つ 4 つの 64 ビットスロットで構成されるストアバッファを備えています。

7.2 セキュリティ拡張機能のサポート

Cortex-A9 プロセッサは、セキュリティ拡張機能をサポートしており、メモリシステムに対するメモリ要求のセキュアまたは非セキュアステータスをエクスポートします。

7.3 レベル 1 命令側メモリシステムについて

レベル 1 命令側メモリシステムは、Cortex-A9 プロセッサに命令ストリームを提供します。総合的なパフォーマンスを向上し、電力消費を低減するため、次の機能が含まれています。

- 動的分岐予測
- 命令のキャッシュ

この動作を、図 7-1 に示します。

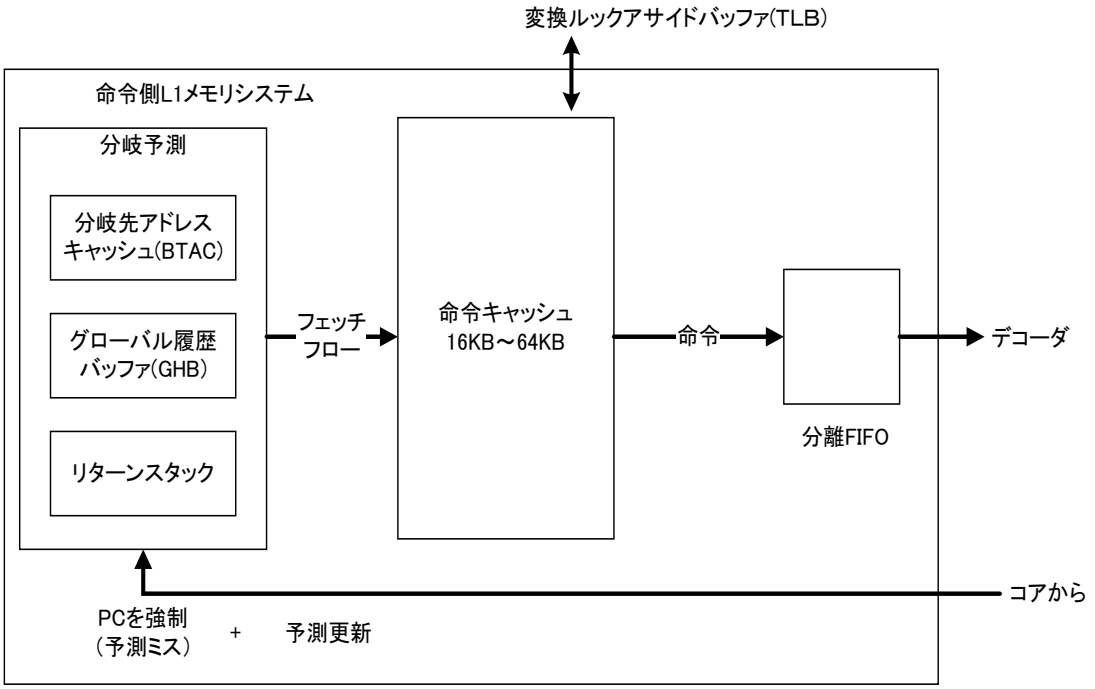


図 7-1 分岐予測と命令キャッシュ

ISide は次の要素で構成されています。

プリフェッチユニット (PFU)

プリフェッチユニットには、次の要素で構成される 2 レベルの予測機構が実装されています。

- 512 エントリの BTAC。RAM に実装され、2 ウェイ × 256 エントリとして編成されています。
- *グローバル履歴バッファ* (GHB)。4096 個の 2 ビット予測器が RAM に実装されています。
- リターンスタック。8 つの 32 ビットエントリが含まれています。

予測方式は、ARM 状態、Thumb 状態、ThumbEE 状態、Jazelle 状態で使用できます。また、ARM から Thumb へ、Thumb から ARM への状態遷移も予測できます。その他の状態遷移は予測できません。また、コアのモードを変更する命令も予測できません。*「プログラムフロー予測」*（ページ 7-6）を参照して下さい。

命令キャッシュコントローラ

命令キャッシュコントローラは、プリフェッチユニットで予測されたプログラムフローに応じて、メモリから命令をフェッチします。

命令キャッシュは 4 ウェイ セットアソシエイティブです。命令キャッシュには次のような特徴があります。

- サイズを 16KB、32KB、または 64KB に構成可能です。
- *仮想インデクス物理タグ付き* (VIPT)
- 64 ビットネイティブのアクセスにより、サイクルごとに最大 4 つの命令をプリフェッチユニットに送ることができます。
- セキュリティ拡張機能のサポート
- ロックダウンはサポートしていません。

7.3.1 プログラムフロー予測の可能化

CP15 c1 制御レジスタの Z ビットを 1 にセットすると、プログラムフロー予測が可能になります。*「システム制御レジスタ」*（ページ 4-15）を参照して下さい。プログラムフロー予測を可能にする前に、BTAC フラッシュ操作を実行する必要があります。

この操作により、GHB が既知の状態に設定される効果もあります。

7.3.2 プログラムフロー予測

次に示すセクションでは、プログラムフロー予測について説明します。

- *「予測される命令と予測されない命令」*
- *「Thumb 状態の条件付き分岐」*
- *「リターンスタック予測」*（ページ 7-7）

予測される命令と予測されない命令

ここでは、プロセッサが予測する命令を示します。特に記載がなければ、このリストは ARM 命令、Thumb 命令、ThumbEE 命令、Jazelle 命令に適用されます。通常、フロー予測ハードウェアは、次のものを含むすべての分岐命令を、アドレシングモードに関係なく予測します。

- 条件付き分岐
- 無条件分岐
- 間接分岐
- PC をデスティネーションとするデータ処理操作
- ARM 状態と Thumb 状態とを切り替える分岐

ただし、一部の分岐命令は予測されません。

- 状態を切り替える分岐（ARM から Thumb、Thumb から ARM への移行を除く）
- 接尾文字が S の命令は、一般に例外からの復帰に使用され、特権モードとセキュリティ状態を変更する副作用があるため、予測されません。
- すべてのモード変更命令

Thumb 状態の条件付き分岐

Thumb 状態では、通常は無条件としてエンコードされる分岐を、*If-Then-Else* (ITE) ブロックに含めることによって条件付きにできます。この分岐は、通常の条件付き分岐として扱われます。

リターンスタック予測

リターンスタックには、関数呼び出し型分岐命令の次の命令のアドレスと実行状態が保存されます。このアドレスは、r14 に保存されるリンクレジスタの値と同じです。次に示す命令では、予測された場合にリターンスタックのプッシュが実行されます。

- `BL` イミディエート
- `BLX(1)` イミディエート
- `BLX(2)` レジスタ
- `HBL`（ThumbEE 状態）
- `HBLP`（ThumbEE 状態）

次に示す命令では、予測された場合にリターンスタックのポップが実行されます。

- `BX r14`
- `MOV pc, r14`
- `LDM r13, {…pc}`
- `LDR pc, [r13]`

`LDR` 命令は、r13 がベースレジスタであれば、任意のアドレシングモードを使用できます。加えて、ThumbEE 状態では、r9 をスタックポインタとして使用することもできるため、デスティネーションとして PC を、ベースレジスタとして r9 を使用した `LDR` 命令と `LDM` 命令も、リターンスタック ポップとして扱われます。

例外からの復帰命令はプロセッサの特権モードとセキュリティ状態を変更する可能性があるため、予測されません。これには、`LDM(3)` 命令と `MOVS pc, r14` 命令が含まれます。

7.4 レベル 1 データ側メモリシステムについて

レベル 1 データキャッシュは、物理的にインデクスとタグが付けられたキャッシュとして編成されています。マイクロ TLB では、キャッシュアクセスを実行する前に、仮想アドレスから物理アドレスが生成されます。

7.4.1 ローカルモニタ

Cortex-A9 プロセッサのレベル 1 メモリシステムにはローカルモニタがあります。これは 2 状態の、オープンで排他的なステートマシンで、排他ロード / ストア (LDREXB、LDREXH、LDREX、LDREXD、STREXB、STREXH、STREX、STREXD) アクセスおよび排他クリア (CLREX) 命令を管理します。これらの命令を使用すれば、セマフォを構築して、CPU 上で実行されている複数のプロセスの同期と、そのセマファに対して同じ一貫したメモリ位置を使用している複数のプロセッサの同期を保証できます。

注

排他ストアでは、MMU フォールトが生成されたり、ローカルモニタの状態にかかわらず、プロセッサでデータウォッチポイント例外が取得されたりする可能性があります。表 10-8 （ページ 10-11）を参照して下さい。

これらの命令の詳細については、*『ARM アーキテクチャ リファレンスマニュアル』*を参照して下さい。

中間 STR 操作の処理

LDREX/STREX コードシーケンスに中間 STR 操作が存在しても、その中間 STR が内部排他モニタに影響を与えることはありません。ローカルモニタは、LDREX の後で排他アクセス状態になり、STR 後は排他アクセス状態を維持し、STREX の後にのみオープンアクセス状態に戻ります。

LDREX 操作と STREX 操作のサイズが異なる場合

LDREX 操作と STREX 操作のサイズが異なる場合は、タグ付きアドレスバイトがストア操作のサイズと一致する、またはサイズの範囲内であることを保証するためにチェックが行われます。

LDREX 命令のタグ付きアドレスの粒度は 8 ワードで、8 ワード境界にアラインされます。このサイズは実装定義であるため、ソフトウェアは他の ARM コアでもこの粒度が同じであることを前提とはできません。

7.4.2 外部アボート処理

レベル 1 データキャッシュは、アクセスのメモリ領域の属性に応じて、2 種類の外部アボートを処理します。

- すべてのストロングリオーダアクセスで、同期アボート機構が使用されます。
- すべてのキャッシュ可能、デバイス、ノーマルのキャッシュ不可メモリの要求では、非同期アボート機構が使用されます。例えば、読み出し時のミスでラインフィルが発行され、アボートが返された場合、そのアボートは非同期としてフラグ付けされます。

7.5 DSB について

Cortex-A9 プロセッサは、DSB 命令の SY オプションのみを実装しています。他のすべての DSB オプションは、完全なシステム DSB 操作として動作しますが、ソフトウェアはこの動作を前提とはできません。

7.6 データのプリフェッチ

ここでは、次のトピックについて説明します。

- *「PLD 命令」*
- *「データのプリフェッチと監視」*

7.6.1 PLD 命令

すべての PLD 命令は、専用リソースを持つ Cortex-A9 プロセッサの専用ユニットで処理されます。これによって、整数コアまたはロードストア ユニットのリソースを使用することが回避されます。

7.6.2 データのプリフェッチと監視

Cortex-A9 データキャッシュは、プロセッサで発生したキャッシュミスを監視する自動プリフェッチ機構を実装しています。このユニットは、2 つの独立したデータストリームを監視およびプリフェッチできます。また、ソフトウェアで CP15 補助制御レジスタのビットを使用してアクティブにすることができます。*「補助制御レジスタ」*（ページ 4-18）を参照して下さい。

ソフトウェアで PLD 命令を発行した場合、PLD プリフェッチユニットが常に、データプリフェッチ機構からの要求よりも優先されます。投機的プリフェッチ機構でプリフェッチされたラインは、割り当て前に破棄される可能性があります。PLD 命令は常に実行され、破棄されることはありません。

7.7 パリティエラーのサポート

構成にパリティエラーのサポートが実装されている場合、次のような特徴があります。

- パリティ方式は偶数パリティです。0000000 のバイトの場合、パリティは 0 です。
- 設計に含まれるそれぞれの RAM についてパリティ情報が生成されます。一般に、RAM のバイトごとに 1 つのパリティビットが生成されます。RAM のビット幅が 8 の倍数でない場合は、残りのビットで 1 つのパリティビットが生成されます。

  パリティビット書き込み可能データもサポートされます。
- パリティがサポートされている設計の RAM アレイには、RAM バンクのデータとともにパリティ情報が格納されます。そのため、設計でパリティのサポートを実装する場合、RAM アレイの幅が広くなります。
- Cortex-A9 ロジックに、パリティ生成ロジックとパリティチェック ロジックが追加されます。

パリティサポート設計の機能とステージを、図 7-2 に示します。ステージ 1 と 2 では、RAM 書き込みとパリティ生成が並列に実行されます。RAM 読み出しとパリティチェックは、ステージ 3 と 4 で並列に実行されます。

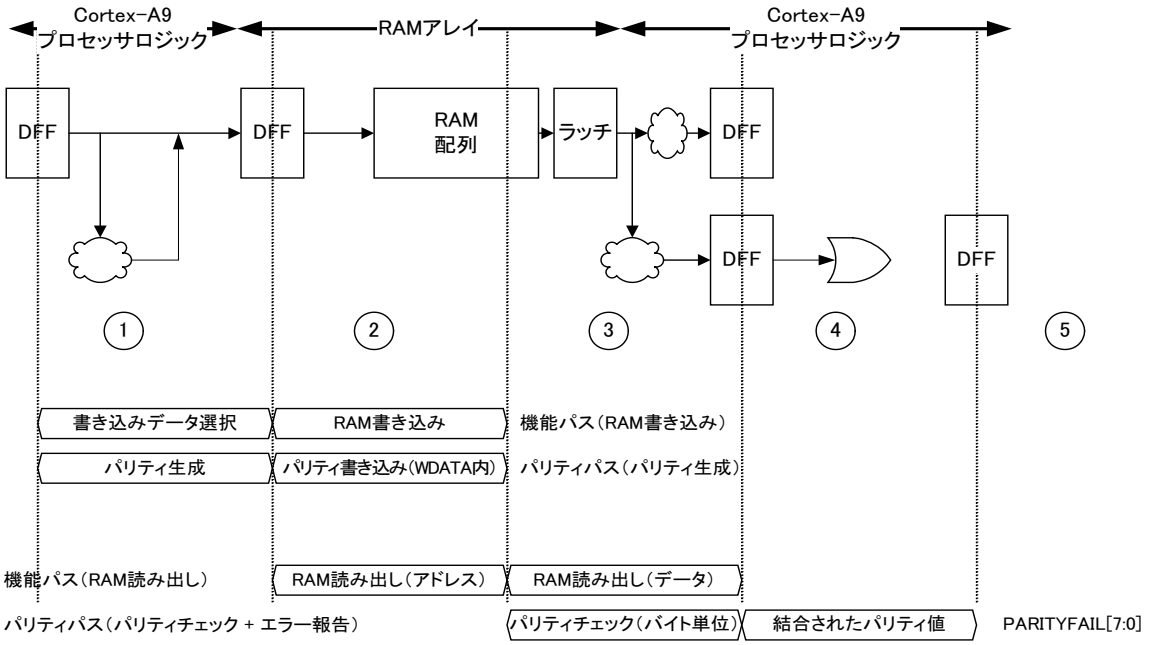


図 7-2 パリティのサポート

パリティエラーは、出力信号 **PARITYFAIL[7:0]** によって報告されます。通常は、対応する RAM 読み出しから 3 クロックサイクル後に、**PARITYFAIL[7:0]** によってパリティエラーが報告されます。

注

これは、正確なエラー検出方式ではありません。設計者は、RAM 用のアドレスレジスタ パイプラインを追加することによって、正確なエラー検出方式を実装できます。このロジックを正しく実装するのは設計者の責任です。

7.7.1 GHB と BTAC のデータ破損

この方式では、GHB RAM と BTAC RAM に対するパリティエラー サポートが提供されますが、この機能によって行える診断は限定されます。GHB データまたは BTAC データが破損した場合は、Cortex-A9 プロセッサで機能エラーが生成されません。GHB データまたは BTAC データの破損は、分岐予測ミスとして検出され、修正されます。

第 8 章
レベル 2 メモリインタフェース

本章では、レベル 2 メモリインタフェースについて説明します。本章は次のセクションから構成されています。

8.1 Cortex-A9 のレベル 2 インタフェース

ここでは、Cortex-A9 のレベル 2 インタフェースについて説明します。

- *「Cortex-A9 のレベル 2 インタフェースについて」*
- *「サポートされている AXI 転送」*（ページ 8-3）
- *「AXI トランザクション ID」*（ページ 8-3）
- *「STRT 命令」*（ページ 8-9）

8.1.1 Cortex-A9 のレベル 2 インタフェースについて

Cortex-A9 のレベル 2 インタフェースは、64 ビット幅の AXI バスマスタで構成されます。

- M0 はデータ側バスです。
- M1 は命令側バスで、書き込みチャネルは存在しません。

AXI マスタ 0 インタフェースの属性を、表 8-1 に示します。

表 8-1 AXI マスタ 0 インタフェースの属性

属性	形式
書き込み発行機能	以下を含めて 12 • 8 つのキャッシュ不可書き込み • 4 つの退出
読み出し発行機能	以下を含めて 10 • 6 つのラインフィル読み出し • 4 つのキャッシュ不可読み出し
統合発行機能	22
書き込み ID 機能	2
書き込みインターリーブ機能	1
書き込み ID 幅	2
読み出し ID 機能	3
読み出し ID 幅	2

AXI マスタ 1 インタフェースの属性を、表 8-2 に示します。

表 8-2 AXI マスタ 1 インタフェースの属性

属性	形式
書き込み発行機能	なし
読み出し発行機能	命令読み出し 4 つ
統合発行機能	4
書き込み ID 機能	なし
書き込みインターリーブ機能	なし

表 8-2 AXI マスタ 1 インタフェースの属性 （続き）

属性	形式
書き込み ID 幅	なし
読み出し ID 機能	4
読み出し ID 幅	2

注

表 8-1 （ページ 8-2）と表 8-2 （ページ 8-2）の数値は、Cortex-A9 MP プロセッサの理論上の最大値です。標準的なシステムで、これらの数値に到達する可能性はほとんどありません。システムリソースについてのプロファイリングを実行し、その結果に応じてパフォーマンスを最適化するように調整を行うことをお勧めします。

AXI プロトコルおよび各 AXI 信号の意味については、本書では解説しません。詳細については、*『AMBA AXI プロトコル v1.0 仕様』* を参照して下さい。

サポートされている AXI 転送

Cortex-A9 のマスタポートでは、可能な AXI トランザクションの一部しか生成されません。

キャッシュ可能トランザクションについは、次のとおりです。
- 読み出し転送用の WRAP4 64 ビット（ラインフィル）
- 書き込み転送用の INCR4 64 ビット（退出）

キャッシュ不可トランザクションについては、次のとおりです。
- INCR N (N : 1 ～ 9) 64 ビット読み出し転送
- 64 ビット書き込み転送用の INCR 1
- INCR N (N : 1 ～ 16) 32 ビット読み出し転送
- 32 ビット書き込み転送用の INCR N (N : 1 ～ 2)
- 8 ビットおよび 16 ビット読み出し / 書き込み転送用の INCR 1
- 8 ビット、16 ビット、32 ビット、64 ビット排他読み出し / 書き込み転送用の INCR 1
- スワップを行う 8 ビットおよび 32 ビット読み出し / 書き込み（ロック付き）の INCR 1
-
-

AXI トランザクションには、次の規則が適用されます。
- WRAP バーストは、読み出し転送、64 ビット、4 転送のみです。
- INCR 1 は、任意のサイズの読み出し / 書き込みが可能です。
- INCR バースト（複数転送）は、32 ビットまたは 64 ビットのみです。
- どのトランザクションも FIXED としてマークされません。
- すべてのバイトストローブが LOW の状態で書き込み転送が発生する可能性があります。

8.1.2 AXI トランザクション ID

AXI ID 信号は次のようにエンコードされます。

- データ側読み出しバスの場合は、**ARIDM0** が次のようにエンコードされます。
  - 2'b00: キャッシュ不可アクセス

- 2'b01: 未使用
- 2'b10: ラインフィル 0 アクセス
- 2'b11: ラインフィル 1 アクセス

- 命令側読み出しバスの場合は、**ARIDM1** が次のようにエンコードされます。
  - 2'b00: 未解決トランザクション
  - 2'b01: 未解決トランザクション
  - 2'b10: 未解決トランザクション
  - 2'b11: 未解決トランザクション
- データ側書き込みバスの場合は、**AWIDM0** が次のようにエンコードされます。
  - 2'b00: キャッシュ不可アクセス
  - 2'b01: 未使用
  - 2'b10: ラインフィル 0 退出
  - 2'b11: ラインフィル 1 退出

8.1.3 AXI USER ビット

AXI USER ビットのエンコードは次のとおりです。

データ側読み出しバス、ARUSERM0[6:0]

ARUSERM0[6:0] のビットエンコードを、表 8-3 に示します。

表 8-3 AWUSERM0[6:0] のエンコード

ビット	名前	説明
[6]	予約	b0
[5]	レベル 2 プリフェッチヒント	この読み出しアクセスがレベル 2 に対するプリフェッチヒントで、データを返すと想定されていないことを示します。
[4:1]	内部属性	b0000 = ストロングリオーダ b0001 = デバイス b0011 = ノーマルメモリ、キャッシュ不可 b0110 = ライトスルー b0111 = ライトバック、書き込み割り当てなし b1111 = ライトバック、書き込み割り当て
[0]	共有ビット	b0 = 非共有 b1 = 共有

命令側読み出しバス、ARUSERM1[6:0]

ARUSERM1[6:0] のビットエンコードを、表 8-4 に示します。

表 8-4 ARUSERM1[6:0] のエンコード

ビット	名前	説明
[6]	予約	b0
[5]	予約	b0
[4:1]	内部属性	b0000 = ストロングリオーダ b0001 = デバイス b0011 = ノーマルメモリ、キャッシュ不可 b0110 = ライトスルー b0111 = ライトバック、書き込み割り当てなし b1111 = ライトバック、書き込み割り当て
[0]	共有ビット	b0 = 非共有 b1 = 共有

データ側書き込みバス、AWUSERM0[8:0]

AWUSERM0[8:0] のビットエンコードを、表 8-5 に示します。

表 8-5 AWUSERM0[8:0] のエンコード

ビット	名前	説明
[8]	早期 BRESP イネーブルビット	レベル 2 スレーブが、書き込み要求に対する早期 BRESP 応答を送信できることを示します。*「早期 BRESP」*（ページ 8-7）を参照して下さい。
[7]	0 のフルライン書き込みビット	このアクセスが、キャッシュライン全体に 0 が書き込まれる操作であることを示します。*「0 のフルライン書き込み」*（ページ 8-8）を参照して下さい。
[6]	クリーン退出	書き込みアクセスが、クリーン キャッシュラインの退出であることを示します。
[5]	レベル 1 退出	書き込みアクセスが、レベル 1 からのキャッシュライン退出であることを示します。
[4:1]	内部属性	b0000 = ストロングリオーダ b0001 = デバイス b0011 = ノーマルメモリ、キャッシュ不可 b0110 = ライトスルー b0111 = ライトバック、書き込み割り当てなし b1111 = ライトバック、書き込み割り当て
[0]	共有ビット	b0 = 非共有 b1 = 共有

8.1.4 排他レベル 2 キャッシュ

Cortex-A9 プロセッサは、排他キャッシュモードをサポートするレベル 2 キャッシュに接続できます。このモードは、Cortex-A9 プロセッサとレベル 2 キャッシュコントローラの両方でアクティブにする必要があります。

このモードでは、Cortex-A9 プロセッサのデータキャッシュとレベル 2 キャッシュは排他です。どの時点でも、特定のアドレスはレベル 1 データキャッシュとレベル 2 キャッシュのいずれかにキャッシュされ、両方に同時にキャッシュされることはあ

りません。これは、Cortex-A9 プロセッサに接続されたレベル 2 キャッシュの使用可能空間と効率性の向上に大きな影響を与えます。排他キャッシュ構成が選択されている場合、次の規則が適用されます。

- データ キャッシュライン置換方式が、ヴィクティムラインがクリーンな場合でも常にレベル 2 メモリに退出するように変更されます。
- レベル 2 キャッシュコントローラ内のラインがダーティな場合は、このアドレスに対するプロセッサからの読み出し要求によって、外部メモリへのライトバックとプロセッサへのラインフィルが発生します。

8.2 レベル 2 メモリインタフェースへのアクセスの最適化

ここでは、レベル 2 メモリインタフェースへのアクセスの最適化について説明します。これらのアクセスの最適化により、Cortex-A9 AXI マスタポートに AXI 準拠でない要求が生成されることがあります。これらの AXI 準拠でない要求は、Cortex-A9 AXI マスタポートに接続されたスレーブでサポート可能な場合にのみ生成する必要があります。レベル 2 キャッシュコントローラは、この種の要求をサポートします。次に示すサブセクションでは、これらの要求について説明します。

- *「レベル 2 メモリインタフェースに対するプリフェッチヒント」*
- *「早期 BRESP」*
- *「0 のフルライン書き込み」*（ページ 8-8）
- *「投機的コヒーレント要求」*（ページ 8-8）

8.2.1 レベル 2 メモリインタフェースに対するプリフェッチヒント

Cortex-A9 プロセッサは、レベル 2 メモリコントローラに対するプリフェッチヒント要求を生成できます。このプリフェッチヒント要求は、データが返されることを想定していない、Cortex-A9 プロセッサで生成される AXI 準拠でない読み出し要求です。

レベル 2 へのプリフェッチヒント要求は、次の方法で生成できます。

- レベル 2 プリフェッチヒント機能（ACTLR のビット [1]）を稼働状態にする。この機能が稼働状態であれば、Cortex-A9 プロセッサは、コヒーレントメモリ上で通常のフェッチパターンを検出したときに、自動的にレベル 2 プリフェッチヒント要求を発行できます。この機能は、Cortex-A9 MPCore プロセッサでのみトリガされ、ユニプロセッサではトリガされません。
- Cortex-A9 プロセッサで利用可能であれば、PLE 動作をプログラムする。この場合、PLE エンジンが、プログラムされたアドレスで一連のレベル 2 プリフェッチヒント要求を発行します。第 9 章 *プリロードエンジン*を参照して下さい。

レベル 2 プリフェッチヒント要求は、ARUSER[5] ビットがセットされていることで識別されます。

注

L2C-310 の追加プログラミングは必要ありません。

8.2.2 早期 BRESP

AXI 仕様によれば、応答チャネル上での **BRESP** 応答は、マスタから最後のデータが送信された時点でのみマスタに返す必要があります。Cortex-A9 プロセッサは、データが送信されたかどうかにかかわらず、アドレスがスレーブで受け付けられた直後に返却された **BRESP** 応答を処理することもできます。これによって、Cortex-A9 プロセッサは、スレーブが早期 BRESP 機能をサポート可能な場合に、書き込み用の帯域幅を広げることができます。Cortex-A9 プロセッサは、**AWUSER[8]** ビットをセットすることによって、そのアクセスに対する早期 **BRESP** 応答を受け付け可能であることをスレーブに伝達します。この機能によりプロセッサの性能を最適化できますが、早期 BRESP 機能によって AXI 準拠でない要求が生成されます。スレーブは、**AWUSER[8]** がセットされた書き込み要求を受け取ったとき、最後のデータの受信後に BRESP 応答を返す（AXI 準拠）ことも、事前に応答を返す（AXI 非準拠）こともできます。L2C-310 キャッシュコントローラは、この AXI 準拠でない機能をサポートします。

Cortex-A9 で、この機能を可能にするためプログラミングの必要はありません。この機能はデフォルトで可能です。

注

この最適化からメリットが得られるように、レベル 2 キャッシュコントローラをプログラムする必要があります。*『AMBA レベル 2 キャッシュコントローラ (L2C-310) テクニカルリファレンス マニュアル』*を参照して下さい。

8.2.3 0 のフルライン書き込み

この機能が可能な場合、Cortex-A9 プロセッサは、単一の要求で、非コヒーレントキャッシュライン全体のビットを 0 として、L2C-310 キャッシュコントローラに書き込むことができます。この操作によって、性能が向上し、ある程度の電力が節約されます。この機能によりプロセッサの性能を最適化できますが、この特殊なアクセスに対してスレーブを最適化させる必要があります。この要求は、関連する **AWUSERM0[7]** ビットをセットすることによって、0 のフルライン書き込みとしてマークされます。

ACTLR のビット [3] をセットすると、この機能が可能になります。*「補助制御レジスタ」*（ページ 4-18）を参照して下さい。

この機能をサポートするには、Cortex-A9 プロセッサでこの機能を可能にする前に、L2C-310 キャッシュコントローラをプログラムする必要があります。*『AMBA レベル 2 キャッシュコントローラ (L2C-310) テクニカルリファレンス マニュアル』*を参照して下さい。

8.2.4 投機的コヒーレント要求

この最適化は、Cortex-A9 MPCore プロセッサでのみ使用できます。*『Cortex-A9 MPCore テクニカルリファレンス マニュアル』*を参照して下さい。

8.3 STRT 命令

STRT 命令を使用している場合は、キャッシュ不可書き込みアクセスに特に注意する必要があります。外部バスに正しい情報を渡すため、次のいずれかの条件が成り立っていることを確認して下さい。

- アクセスがストロングリオーダメモリに対して行われる。
  これによって、STRT 命令がストアバッファ内で結合しないことが保証されます。
- アクセスがデバイスメモリに対して行われる。
  これによって、STRT 命令がストアバッファ内で結合しないことが保証されます。
- DSB 命令が STRT の前後で発行される。
  これによって、STRT が、同じ 64 ビットアドレスの既存のスロットや、同じ 64 ビットアドレスの別の書き込みと結合することがなくなります。

Cortex-A9 のモードと、対応する **AxPROT** の値を、表 8-6 に示します。

表 8-6 Cortex-A9 のモードと AxPROT の値

プロセッサモード	アクセスタイプ	AxPROT の値
ユーザ	キャッシュ可能読み出しアクセス	ユーザ
特権		特権
ユーザ	キャッシュ不可読み出しアクセス	ユーザ
特権		特権
-	キャッシュ可能書き込みアクセス	常に特権としてマークされる
ユーザ	キャッシュ不可書き込みアクセス	ユーザ
特権	キャッシュ不可書き込みアクセス	特権（STRT を使用している場合は除きます）

第 9 章
プリロードエンジン

設計に*プリロードエンジン* (PLE) を含めることができます。PLE は、選択されたメモリ領域をレベル 2 にロードします。本章では、PLE について説明します。本章は次の項目から構成されています。

9.1 プリロードエンジンについて

実装されている場合、PLE は選択されたメモリ領域をレベル 2 にロードします。PLE をプログラムするには、MCRR プリロードチャネル操作を使用して下さい。専用のイベントにより、メモリ領域の動作を監視します。L2C-310 の追加イベントにより、PLE の動作も監視できます。

プリロード操作パラメータは、次の要素を含む PLE FIFO に入力されます。

- プログラム対象のパラメータ
  - ベースアドレス
  - ストライド長
  - ブロック数
- 有効ビット
- NS 状態ビット
- *変換テーブルベース* (TTB) アドレス
- *アドレス空間識別子* (ASID) の値

プリロードブロックは複数のページエントリにまたがることができます。コンテキストスイッチが発生しても、プログラムされたエントリは有効なまま残る可能性があります。

プリロードエンジンは、独自の TTB パラメータと ASID パラメータを使用することを除いて、標準の PLD 要求と同様にキャッシュラインのプリロード要求を処理します。変換アボートが発生した場合は、プリロード要求が無視され、プリロードエンジンから次の要求が発行されます。

すべての MMU 設定が保存されるわけではありません。ドメイン、TEX 再マップ、1 次再マップ、通常再マップ、アクセス許可のレジスタは保存されません。そのため、これらのレジスタのいずれかに書き込むと、FIFO 全体とアクティブチャネルがフラッシュされます。

さらに、*変換ルックアサイドバッファ* (TLB) 保守操作では、FIFO エントリにも保守操作を適用する必要があります。この操作は次のように実行されます。

MVA と ASID による無効化時

一致する ASID を持つすべてのエントリを無効化します。

ASID による無効化時

一致する ASID を持つすべてのエントリを無効化します。

MVA とすべての ASID による無効化時

FIFO 全体をフラッシュします。

TLB 全体の無効化時

FIFO 全体をフラッシュします。

この規則は、PLE アクティブチャネルにも適用可能です。

プリロードエンジンは、次の MCRR 命令を定義して、プリロードブロックに使用します。

```
MCRR p15, 0, <Rt>,<Rt2> c11;Program new PLE channel
```

FIFO のエントリ数は、RTL 構成設計の選択として設定できます。使用可能なサイズは次のとおりです。

- 16 エントリ
- 8 エントリ
- 4 エントリ

9.2 PLE 制御レジスタの説明

PLE 制御レジスタは、CRn が c11 のときにアクセスされる CP15 レジスタです。*「CP15 c11 レジスタの概要」*（ページ 4-30）を参照して下さい。次に示すセクションでは、PLE 制御レジスタについて説明します。

- *「PLE ID レジスタ」*（ページ 4-30）
- *「PLE 動作ステータスレジスタ」*（ページ 4-31）
- *「PLE FIFO ステータスレジスタ」*（ページ 4-32）
- *「プリロードエンジン ユーザアクセス許可レジスタ」*（ページ 4-32）
- *「プリロードエンジン パラメータ制御レジスタ」*（ページ 4-33）

すべての CP15 c11 システム制御レジスタについて、非セキュアアクセスは NSAC.PLE によって制御されます。これらの制御レジスタで使用される操作を、*「PLE 操作」*（ページ 9-4）に示します。

9.3 PLE 操作

次に示すセクションでは、PLE 操作について説明します。

- *「プリロードエンジンの FIFO フラッシュ操作」*
- *「プリロードエンジンのチャネル一時停止操作」*
- *「プリロードエンジンのチャネル再開操作」*
- *「プリロードエンジンのチャネル停止操作」*（ページ 9-5）
- *「PLE の新規チャネルプログラム操作」*（ページ 9-5）

すべてのプリロードエンジン操作には、次の規則が適用されます。

- 非セキュア実行は、NSACR.PLE によって制御されます。
- ユーザ実行は、PLEUAR.EN によって制御されます。
- この操作は、プリロードエンジンが存在する構成でのみ使用可能です。それ以外の構成では、未定義命令例外が取得されます。

9.3.1 プリロードエンジンの FIFO フラッシュ操作

PLEFF 操作の特徴は次のとおりです。

目的 現在実行中の PLE チャネルを含めて、事前にプログラムされたすべての PLE チャネルをフラッシュします。

PLE FIFO フラッシュ操作を実行するには、次の命令を使用します。

```
MCR p15, 0, <Rt>, c11, c2, 1
```

この操作では、<Rt> は無視されます。

9.3.2 プリロードエンジンのチャネル一時停止操作

PLEPC 操作の特徴は次のとおりです。

目的 PLE の動作を一時停止します。

PLEPC 操作は、アクティブな PLE チャネルが存在しない場合でも実行できます。この場合、後で新しい PLE チャネルがプログラムされても、PLE のチャネル再開操作が実行されるまで、PLE の動作は再開されません。

PLE PC 操作を実行するには、次の命令を使用します。

```
MCR p15, 0, <Rt>, c11, c3, 0
```

この操作では、<Rt> は無視されます。

9.3.3 プリロードエンジンのチャネル再開操作

PLERC 操作の特徴は次のとおりです。

目的 プリロードエンジンの動作を再開させます。

PLE が一時停止していないときに PLERC 操作を実行した場合、チャネル再開操作は無視されます。

PLERC 操作を実行するには、次の命令を使用します。

```
MCR p15, 0, <Rt>, c11, c3, 1
```

9.3.4 プリロードエンジンのチャネル停止操作

PLEKC 操作の特徴は次のとおりです。

目的 現在アクティブな PLE チャネルを停止します。

この操作は、PLE FIFO 内の PLE 要求に対しては何も動作を行いません。

PLEKC 操作を実行するには、次の命令を使用します。

```
MCR p15, 0, <Rt>, c11, c3, 2
```

9.3.5 PLE の新規チャネルプログラム操作

PLE の新規チャネルプログラム操作の特徴は次のとおりです。

目的 レベル 2 メモリにプリロードする、新しいメモリ領域をプログラムします。現在アクティブな PLE チャネルを停止します。

PLE の新規チャネルプログラム操作に関する <Rt> と <Rt2> のビット割り当てを、図 9-1 に示します。Rt は、ベースアドレスが格納されたレジスタです。Rt2 は、ブロックの長さ、ストライド、および数が格納されたレジスタです。

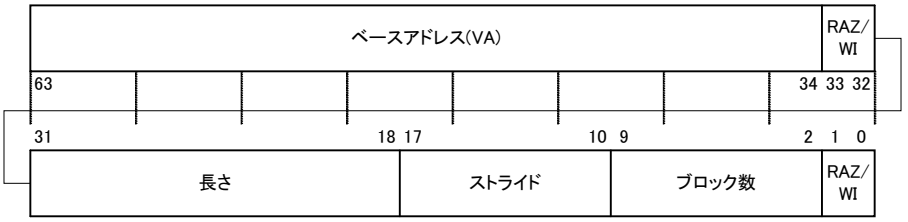


図 9-1 新規チャネルプログラム操作のビット割り当て

PLE の新規チャネルプログラム操作のビット割り当てを、表 9-1 に示します。

表 9-1 PLE の新規チャネルプログラム操作のビット割り当て

ビット	名前	説明
[63:34]	ベースアドレス (VA)	これは、プリロードする最初のメモリブロックの 32 ビットベース仮想アドレスです。このアドレスはワード境界にアラインされています。つまり、ビット [33:32] は RAZ/WI です。
[33:32]	-	RAZ/WI
[31:18]	長さ	プリロードするブロックの長さを指定します。 長さは、ワードの倍数としてエンコードされます。範囲は、14'b0000000000（単一ワードブロック）～ 14'b11111111111111（16K ワードブロック）です。
[17:10]	ストライド	ブロック間のプリロードストライドを指定します。プリロードストライドは、2 つのブロックの開始アドレスの差です。ストライドは、ワードの倍数としてエンコードされます。範囲は、8'b00000000（連続ブロック）～ 8'b11111111（256 ワード単位のプリフェッチブロック）です。
[9:2]	ブロック数	プリロードするブロックの数を指定します。 値の範囲は、単一ブロックのプリロードを示す 8'b00000000 から、256 ブロックを示す 8'b11111111 までです。
[1:0]	-	RAZ/WI

新規チャネルのプログラム操作には、次の MCRR 操作を使用します。

```
MCRR p15, 0, <Rt>,<Rt2> c11; Program new PLE channel
```

注

新しくプログラムされた PLE エントリは、PLE FIFO に使用可能なエントリがあれば、その FIFO に書き込まれます。FIFO がオーバフローした場合は、命令が警告なしに失敗して、FIFO オーバフローイベント信号がアサートされます。プリロードイベントについては、表 11-7 （ページ 11-7）を参照して下さい。*「PLE FIFO ステータスレジスタ」*（ページ 4-32）を参照して下さい。

第 10 章
デバッグ

本章では、プロセッサのデバッグユニットについて説明します。この機能は、アプリケーションソフトウェア、オペレーティングシステム、ハードウェアの開発を支援します。本章は次のセクションから構成されています。

10.1 デバッグインタフェースについて

Cortex-A9 プロセッサは、*『ARM アーキテクチャ リファレンスマニュアル』* に記載されている ARMv7 デバッグアーキテクチャを実装しています。また、*『ARM アーキテクチャ リファレンスマニュアル』* に記載されているデバッグイベントの組も実装しています。

さらに、次の機能が実装されています。

- Cortex-A9 プロセッサ固有のイベント。詳細については、*「パフォーマンス監視イベント」*（ページ 11-7）を参照して下さい。
- システムコヒーレンシ イベント

「パフォーマンス監視」（ページ 2-3）を参照して下さい。第 11 章 *パフォーマンス監視ユニット*も参照して下さい。

10.1.1 デバッグモード

デバッグモードは、認証信号によって制御されます。認証信号は、プロセッサモードの特定のサブセットとセキュリティ状態でのみ動作をデバッグまたはトレースできるように、プロセッサを構成します。*「認証信号」*（ページ 10-16）を参照して下さい。

注

Cortex-A9 プロセッサは、SPIDEN ピンによって侵襲性デバッグが可能になっている場合のみ、セキュア ユーザモードでのホールトモード デバッグをサポートします。SPIDEN が LOW の場合は、SDR.SUIDEN ビットをセットすることによって、セキュア ユーザモードでのモニタモード デバッグのみを使用できます。つまり、**SPIDEN** が LOW の場合、SDR.SUIDEN ビットが 1 にセットされていても、コアはホールデバッグ モードに移行することはできません。外部の **SPIDEN** ピンを HIGH に設定することによって、この制限を取り除くことができます。

10.1.2 ブレークポイントとウォッチポイント

次のような種類があります。

- 6 つのブレークポイント。これらのうち BRP4 と BRP5 には、コンテキスト ID 比較機能が含まれています。*「ブレークポイント値レジスタ」*（ページ 10-7）と *「ブレークポイント制御レジスタ」*（ページ 10-8）を参照して下さい。
- 4 つのウォッチポイント。

  ウォッチポイントイベントは常に同期で、同期データアボートと同じ動作を行います。デバッグ開始方法 (DBGDSCR[5:2]) の値が b0010 になることはありません。

  ウォッチポイントが設定されたアクセスで同期アボートが発生した場合、同期アボートの方がウォッチポイントよりも優先されます。

  アボートが非同期で、アクセスに関連付けることができない場合、取得される例外は予測不能です。

  キャッシュ保守操作では、ウォッチポイントイベントは生成されません。

  「ウォッチポイント値レジスタ」（ページ 10-10）と *「ウォッチポイント制御レジスタ」*（ページ 10-11）を参照して下さい。

10.1.3 非同期アボート

Cortex-A9 プロセッサは、可能性のあるすべての未解決な非同期データアボートが、デバッグ状態に入る前に認識されることを保証しています。

10.1.4 プロセッサのインタフェース

Cortex-A9 プロセッサには、デバッグとパフォーマンスモニタに関する次のインタフェースが存在します。

デバッグレジスタ

このインタフェースは、ベースライン CP14 インタフェース、拡張 CP14 インタフェース、およびメモリマップされたインタフェースです。

「CTI 信号」（ページ A-22）と *「APB インタフェース信号」*（ページ A-22）を参照して下さい。

パフォーマンスモニタ

このインタフェースは、CP15 ベースのインタフェースと、メモリマップされたインタフェースです。*「パフォーマンス監視」*（ページ 2-3）を参照して下さい。第 11 章 *パフォーマンス監視ユニット*も参照して下さい。

10.1.5 リセットのデバッグレジスタへの影響

nDBGRESET

nDBGRESET は、デバッグロジック リセット信号です。この信号は、パワーオンリセット シーケンス中にアサートする必要があります。

他のリセット信号である **nCPURESET** および **nNEONRESET** は、MPE が存在すれば、デバッグロジックに影響を与えません。

デバッグリセット時には、次の規則が適用されます。

- デバッグ状態は変更されません。つまり、DBGSCR.HALTED は変更されません。
- プロセッサは、保留されているホールトデバッグ イベントの DBGDRCR.HaltReq を削除します。

10.2 Cortex-A9 デバッグインタフェースについて

デバッグインタフェースは、次の要素で構成されています。

- ベースライン CP14 インタフェース
- 拡張 CP14 インタフェース
- *デバッグアクセス ポート* (DAP) 経由で外部のデバッガに接続される、外部デバッグインタフェース

Cortex-A9 デバッグインタフェースを、図 10-1 に示します。

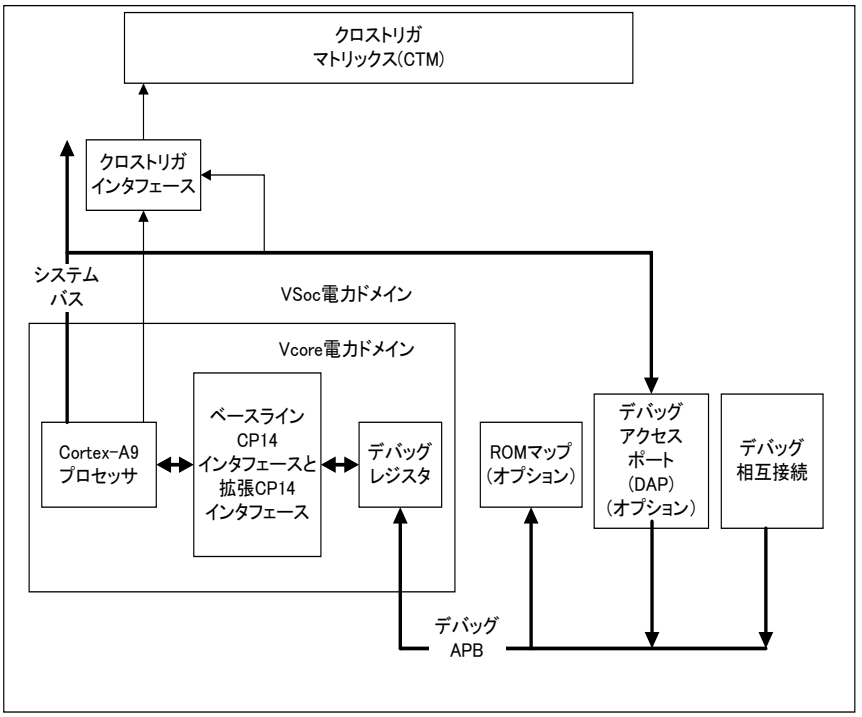


図 10-1 デバッグレジスタ インタフェースと CoreSight インフラストラクチャ

10.2.1 デバッグレジスタへのアクセス

デバッグレジスタには、次の方法でアクセスできます。

- cp14 インタフェース経由。デバッグレジスタが、コプロセッサ命令にマップされます。
- 次の例外を伴う、関連オフセットを使用した APB 経由
  - DBGRAR
  - DBGSAR
  - DBGSCR-int
  - DBGTR-int

DBSCR と DBGTR の外部ビューには、メモリマップされた APB アクセス経由でアクセスできます。

CP14 インタフェースレジスタを、表 10-1 （ページ 10-5）に示します。他のレジスタについては、*『ARM アーキテクチャ リファレンスマニュアル』*を参照して下さい。

表 10-1 CP14 インタフェースレジスタ

レジスタ番号	オフセット	CP14 命令	アクセス	レジスタ名	説明
0	0x000	0, c0, c0, 0	RO	DBGDIDR[a]	-[b]
-	-	0, c1, c0, 0	RO	DBGDRAR[a]	-
-	-	0, c2, c0, 0	RO	DBGDSAR[a]	-
-	-	0, c0, c1, 0	RO	DBGDSCRint[ab]	-
5	-	0, c0, c5, 0	R	DBGDTRRXint[a]	-
	-		W	DBGDTRTXint[a]	-
6	0x018	0, c0, c6, 0	RW	DBGWFAR	DBGWFAR の使用は、ウォッチポイントが同期しているため、ARMv7 アーキテクチャでは非推奨です。
7	0x01C	0, c0, c7, 0	RW	DBGVCR	-
8	-	-	-	予約	-
9	0x024	0, c0, c9, 0	RAZ/WI	DBGECR	実装されていない
10	0x028	0, c0, c10, 0	RAZ/WI	DBGDSCCR	*「デバッグ状態キャッシュ制御レジスタ (DBGDSCCR)」*（ページ 10-7）
11	0x02C	0, c0, c11, 0	RAZ/WI	DBGDSMCR	実装されていない
12 ～ 31	-	-	-	予約	-
32	0x080	0, c0, c0, 2	RW	DBGDTRRXext	-
33	0x084	0, c0, c1, 2	WO	DBGITR	-
33	0x084	0, c0, c1, 2	RO	DBGPCSR	-
34	0x088	0, c0, c2, 2	RW	DBGDSCRext	-
35	0x08C	0, c0, c3, 2	RW	DBGDTRTXext	-
36	0x090	0, c0, c4, 2	WO	DBGDRCR	-
37 ～ 63	-	-	-	予約	-
64 ～ 68	0x100 ～ 0x114	0, c0, c0 ～ c5, 4	RW	DBGBVRn	*「ブレークポイント値レジスタ」*（ページ 10-7）
69 ～ 79	-	-	-	予約	-
80 ～ 85	0x140 ～ 0x154	0, c0, c0 ～ c5, 5	RW	DBGBCRn	*「ブレークポイント制御レジスタ」*（ページ 10-8）
86 ～ 95	-	-	-	予約	-
96 ～ 99	0x180 ～ 0x18BC	0, c0, c0 ～ c3, 6	RW	DBGWVRn	*「ウォッチポイント値レジスタ」*（ページ 10-10）
100 ～ 111	-	-	-	予約	-
112 ～ 115	0x1C0 ～ 0x1DC	0, c0, c0 ～ c3, 7	RW	DBGWCRn	*「ウォッチポイント制御レジスタ」*（ページ 10-11）
116 ～ 191	-	-	-	予約	-

表 10-1 CP14 インタフェースレジスタ （続き）

レジスタ番号	オフセット	CP14 命令	アクセス	レジスタ名	説明
192	0x300	0, c1, c0, 4	RAZ/WI	DBGOSLAR	実装されていない
193	0x304	0, c1, c1, 4	RAZ/WI	DBGOSLSR	実装されていない
194	0x308	0, c1, c2, 4	RAZ/WI	DBGOSSRR	実装されていない
195	-	-	-	予約	-
196	0x310	0, c1, c4, 4	RO	DBGPRCR	-
197	0x314	0, c1, c5, 4	RO	DBGPRSR	-
198 ～ 511	-	-	-	予約	-
512 ～ 575	0x800 ～ 0x8FC	-	-	-	PMU レジスタ[c]
576 ～ 831	-	-	-	予約	-
832 ～ 895	0xD00 ～ 0xDFC	0, c6, c0, 15, 4 ～ 7	RW	予測不能	-
896 ～ 927	-	-		予約	-
928 ～ 959	0xE80 ～ 0xEFC0	0, c7, c0, 15, 2 ～ 3	RAZ/WI	-	-
960	0xF00	0, c7, c0, 4	RAZ/WI	DBGITCTRL	統合モード制御レジスタ
961 ～ 999	0xF04 ～ 0xF9C	-	-	-	-
1000	0xFA0	0, c7, c8, 6	RW	DBGCLAIMSET	クレームタグ セットレジスタ
1001	0xFA4	0, c7, c9, 6	RW	DBGCLAIMCLR	クレームタグ クリアレジスタ
1002 ～ 1003	-	-	-	予約	-
1004	0xFB0	0, c7, c12, 6	WO	DBGLAR	ロックアクセス レジスタ
1005	0xFB4	0, c7, c13, 6	RO	DBGLSR	ロックステータス レジスタ
1006	0xFB8	0, c7, c14, 6	RO	DBGAUTHSTATUS	認証ステータスレジスタ
1007 ～ 1009	-	-	-	予約	-
1010	0xFC8	0, c7, c2, 7	RAZ	DBGDEVID	-
1011	0xFCC	0, c7, c3, 7	RO	DBGDEVTYPE	デバイスタイプ レジスタ
1012 ～ 1016	0xFD0 ～ 0xFEC	0, c7, c4 ～ 8, 7	RO	PERIPHERALID	*「CoreSight 識別レジスタ」*（ページ 10-14）
1017 ～ 1019	-	-	-	予約	-
1020 ～ 1023	0xFF0 ～ 0xFFC	0, c7, c12 ～ 15, 7	RO	COMPONENTID	*「CoreSight 識別レジスタ」*（ページ 10-14）

a. ベースライン CP14 インタフェース。このレジスタには、メモリマップされたインタフェースと CP14 インタフェースを経由した外部ビューも存在します。
b. DBGSCR のビット [12] がクリアされている場合、ユーザモードでアクセスできます。特権モードでもアクセスできます。
c. PMU レジスタは、CP15 インタフェースの一部です。拡張 CP14 インタフェースから読み出すと、0 が返されます。*「CP15 c9 レジスタの概要」*（ページ 4-28）を参照して下さい。第 11 章 *パフォーマンス監視ユニット*も参照して下さい。

10.3 デバッグレジスタの説明

ここでは、デバッグレジスタについて説明します。

10.3.1 デバッグ状態キャッシュ制御レジスタ (DBGDSCCR)

DSCCR は、プロセッサがデバッグ状態のときに、キャッシュの動作を制御します。Cortex-A9 プロセッサは、デバッグ状態キャッシュ制御レジスタの機能を実装していません。デバッグ状態キャッシュ制御レジスタは 0 として読み出されます。

10.3.2 ブレークポイント値レジスタ

ブレークポイント値レジスタ (BVR) は、オフセットが 0x100 ～ 0x114 のレジスタ 64 ～ 68 です。各 BVR には、次のように、*ブレークポイント制御レジスタ* (BCR) が関連付けられています。

- BVR0 と BCR0
- BVR1 と BCR1

このパターンが BVR5 と BCR5 まで続きます。

ブレークポイントレジスタのペアである BVRn と BCRn は、*ブレークポイントレジスタ ペア* (BRPn) と呼ばれます。

BVR と、対応する BCR を、表 10-2 に示します。

表 10-2 BVR と対応する BCR

ブレークポイント値レジスタ			ブレークポイント制御レジスタ		
レジスタ	レジスタ番号	オフセット	レジスタ	レジスタ番号	オフセット
BVR0	64	0x100	BCR0	80	0x140
BVR1	65	0x104	BCR1	81	0x144
BVR2	66	0x108	BCR2	82	0x148
BVR3	66	0x10C	BCR3	83	0x14C
BVR4	67	0x110	BCR4	84	0x150
BVR5	68	0x114	BCR5	85	0x154

このレジスタに含まれているブレークポイント値は、*命令仮想アドレス* (IVA) またはコンテキスト ID に対応しています。ブレークポイントを設定可能な対象は次のとおりです。

- IVA
- コンテキスト ID 値
- IVA とコンテキスト ID のペア

IVA とコンテキスト ID のペアの場合は、2 つの BRP をリンクする必要があります。IVA とコンテキスト ID のペアが同時に一致したときに、デバッグイベントが生成されます。

ブレークポイント値レジスタの各ビット値の意味を、表 10-3 に示します。

表 10-3 ブレークポイント値レジスタのビットの機能

ビット	名前	説明
[31:0]	-	ブレークポイント値。リセット時の値は 0 です。

注

- BRP4 と BRP5 のみがコンテキスト ID 比較をサポートします。
- BVR0[1:0]、BVR1[1:0]、BVR2[1:0]、BVR3[1:0] は、コンテキスト ID 比較をサポートしていないため、書き込みに対しては常に 0 または保持で、読み出しに対しては読み出し値 0 です。
- BVR と照合するコンテキスト ID 値は、CP15 コンテキスト ID レジスタの内容で指定されます。

10.3.3 ブレークポイント制御レジスタ

BCR は読み出し / 書き込みレジスタで、次の設定に必要な制御ビットが含まれています。

- ブレークポイント
- リンク付きブレークポイント

BCR のビット配置を、図 10-2 に示します。

図 10-2 ブレークポイント制御レジスタのビット割り当て

ブレークポイント制御レジスタの各ビット値の意味を、表 10-4 に示します。

表 10-4 ブレークポイント制御レジスタのビット割り当て

ビット	名前	説明
[31:29]	-	読み出し時は RAZ、書き込み時は SBZP
[28:24]	ブレークポイントアドレス マスク	ブレークポイントアドレス マスク RAZ/WI b00000 = マスクなし
[23]	-	読み出し時は RAZ、書き込み時は SBZP
[22:20]	M	BVR の意味。 b000 = 命令仮想アドレス一致 b001 = リンク付き命令仮想アドレス一致 b010 = リンクなしコンテキスト ID b011 = リンク付きコンテキスト ID b100 = 命令仮想アドレス不一致 b101 = リンク付き命令仮想アドレス不一致 b11x = 予約 注 BCR0[21]、BCR1[21]、BCR2[21]、BCR3[21] は、コンテキスト ID 比較機能がないため、読み出しに対して RAZ です。

表 10-4 ブレークポイント制御レジスタのビット割り当て （続き）

ビット	名前	説明
[19:16]	リンク付き BRP	リンク付き BRP 番号。ここにエンコードされるバイナリ番号は、この BRP とリンクされる別の BRP を示しています。 **注** • BRP が自分自身とリンクされている場合、ブレークポイント デバッグイベントが生成されるかどうかは予測不能です。 • BRP が別の BRP とリンクされており、その別の BRP がリンク付きコンテキスト ID 一致用に構成されていない場合、ブレークポイント デバッグイベントが生成されるかどうかは予測不能です。
[15:14]	セキュア状態アクセス制御	セキュア状態アクセス制御。このフィールドを使用して、ブレークポイントをプロセッサのセキュリティ状態に対して条件付きにできます。 b00 = ブレークポイントは、セキュア状態と非セキュア状態の両方で一致します。 b01 = ブレークポイントは、非セキュア状態でのみ一致します。 b10 = ブレークポイントは、セキュア状態でのみ一致します。 b11 = 予約
[13:9]	-	読み出し時は RAZ、書き込み時は SBZP
[8:5]	バイトアドレス選択	バイトアドレス選択。IVA と一致するようにプログラムされたブレークポイントの場合は、ワードアラインしたアドレスを BVR に書き込む必要があります。その後でこのフィールドを使用して、特定のバイトアドレスがアクセスされた場合にのみヒットするようにブレークポイントをプログラムできます。 IVA 一致用に BRP をプログラムした場合、このフィールドは次の意味となります。 b0000 = ブレークポイントは一切ヒットしません。 b0011 = BVR & 0xFFFFFFFC + 0 のアドレスから始まる 2 バイトのいずれかがアクセスされた場合に、ブレークポイントがヒットします。 b1100 = BVR & 0xFFFFFFFC + 2 のアドレスから始まる 2 バイトのいずれかがアクセスされた場合に、ブレークポイントがヒットします。 b1111 = BVR & 0xFFFFFFFC + 0 のアドレスから始まる 4 バイトのいずれかがアクセスされた場合に、ブレークポイントがヒットします。 IVA 不一致用に BRP をプログラムした場合は、対応する IVA ブレークポイントがヒットしないブレークポイントがヒットします。つまり、IVA 不一致ブレークポイントでカバーされるアドレスの範囲は、対応する IVA ブレークポイントの否定イメージとなります。 コンテキスト ID 比較用に BRP をプログラムする場合は、このフィールドを b1111 にセットする必要があります。それ以外の場合、ブレークポイントおよびウォッチポイント デバッグイベントが期待したように生成されない場合があります。 **注** BCR[8] と BCR[7] が等しくない、または BCR[6] と BCR[5] が等しくない値を BCR[8:5] へ書き込んだ場合、結果は予測不能です。
[4:3]	-	読み出し時は RAZ、書き込み時は SBZP
[2:1]	SP	スーパバイザアクセス制御。ブレークポイントを、プロセッサのモードについて条件付きにできます。 b00 = ユーザ、システム、またはスーパバイザ b01 = 特権 b10 = ユーザ b11 = 任意
[0]	B	ブレークポイント イネーブル。 0 = ブレークポイントは不可能です。これはリセット時の値です。 1 = ブレークポイントが可能です。

BVR ビットの意味を、表 10-5 に示します。

表 10-5 BVR のビット [22:20] の意味

BVR[22:20]	意味
b000	対応する BVR[31:2] が IVA バスと比較され、プロセッサの状態がこの BCR と比較されます。IVA と状態の組み合わせが一致した場合に、ブレークポイント デバッグイベントが生成されます。
b001	対応する BVR[31:2] が IVA バスと比較され、プロセッサの状態がこの BCR と比較されます。この BRP は、BCR[19:16] リンク付き BRP フィールドで示される BRP とリンクされています。IVA、コンテキスト ID、および状態の組み合わせが一致した場合に、ブレークポイント デバッグイベントが生成されます。
b010	対応する BVR[31:0] が CP15 コンテキスト ID レジスタ、c13 と、プロセッサの状態がこの BCR と、それぞれ比較されます。この BRP は他の BRP とリンクされていません。コンテキスト ID と状態の両方が一致した場合、ブレークポイント デバッグイベントが生成されます。この BRP では、BCR[8:5] を b1111 にセットする必要があります。それ以外の場合、ブレークポイント デバッグイベントが生成されるかどうかは予測不能です。
b011	対応する BVR[31:0] が、CP15 コンテキスト ID レジスタ、c13 と比較されます。この BRP は、別の BRP（BCR[21:20] = b01 タイプ）または WRP (WCR[20] = b1) とリンクされています。IVA または DVA とコンテキスト ID の組み合わせの一致時に、ブレークポイントまたはウォッチポイントデバッグイベントを生成します。この BRP では、BCR[8:5] を b1111 に、BCR[15:14] を b00 に、BCR[2:1] を b11 にセットする必要があります。それ以外の場合、ブレークポイント デバッグイベントが生成されるかどうかは予測不能です。
b100	対応する BVR[31:2] と BCR[8:5] が IVA バスと比較され、プロセッサの状態がこの BCR と比較されます。IVA が一致せず、状態が一致した場合に、ブレークポイント デバッグイベントが生成されます。
b101	対応する BVR[31:2] と BCR[8:5] が IVA バスと比較され、プロセッサの状態がこの BCR と比較されます。この BRP は、BCR[19:16] リンク付き BRP フィールドで示される BRP とリンクされています。IVA が一致せず、状態とコンテキスト ID が一致した場合に、ブレークポイント デバッグイベントが生成されます。
b11x	予約。動作は予測不能です。

10.3.4 ウォッチポイント値レジスタ

ウォッチポイント値レジスタ (WVR) は、オフセットが 0x180 ～ 0x18C のレジスタ 96 ～ 99 です。各 WVR には、*ウォッチポイント制御レジスタ* (WCR) が関連付けられています。次に例を示します。

- WVR0 と WCR0
- WVR1 と WCR1

このパターンが WVR3 と WCR3 まで続きます。

WVR と、対応する WCR を、表 10-6 に示します。

表 10-6 WVR と対応する WCR

ウォッチポイント値レジスタ			ウォッチポイント制御レジスタ		
レジスタ	レジスタ番号	オフセット	レジスタ	レジスタ番号	オフセット
WVR0	96	0x180	WCR0	112	0x1C0
WVR1	97	0x184	WCR1	113	0x1C4
WVR2	98	0x188	WCR2	114	0x1C8
WVR3	99	0x18C	WCR3	115	0x1DC

ウォッチポイントレジスタのペアである WVRn と WCRn は、*ウォッチポイント レジスタペア* (WRPn) と呼ばれます。

WVR に格納されたウォッチポイント値は、常に*データ仮想アドレス* (DVA) に対応し、次のいずれかに設定できます。

- DVA
- DVA とコンテキスト ID のペア

DVA とコンテキスト ID のペアの場合は、WRP と、コンテキスト ID 比較機能を持つ BRP とをリンクする必要があります。デバッグイベントは、DVA とコンテキスト ID のペアが同時に一致した場合に生成されます。ウォッチポイント値レジスタの各ビット値のと機能との対応を、表 10-7 に示します。

表 10-7 ウォッチポイント値レジスタのビット機能

ビット	名前	説明
[31:2]	-	ウォッチポイントのアドレス
[1:0]	-	読み出し時は RAZ、書き込み時は SBZP

10.3.5 ウォッチポイント制御レジスタ

WCR には、次のポイントを設定するために必要な制御ビットが保持されます。

- ウォッチポイント
- リンク付きウォッチポイント

ウォッチポイント制御レジスタのビット配置を、図 10-3 に示します。

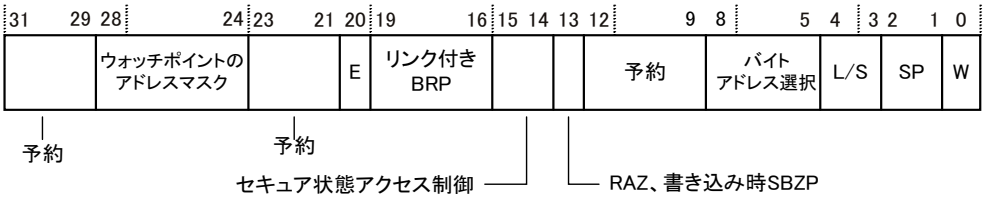


図 10-3 ウォッチポイント制御レジスタのビット割り当て

ウォッチポイント制御レジスタの各ビット値と機能との対応を、表 10-8 に示します。

表 10-8 ウォッチポイント制御レジスタのビット割り当て

ビット	名前	説明
[31:29]	-	読み出し時は RAZ、書き込み時は SBZP
[28:24]	ウォッチポイント アドレスマスク	ウォッチポイントアドレスのマスク
[23:21]	-	読み出し時は RAZ、書き込み時は SBZP
[20]	E	リンクイネーブル ビット。 0 = リンクは不可能です。 1 = リンクが可能です。 このビットがセットされている場合、このウォッチポイントは、リンク付き BRP フィールドで選択された BRP を保持しているコンテキスト ID にリンクされています。
[19:16]	リンク付き BRP	リンク付き BRP 番号。ここにエンコードされるバイナリ数値は、この WRP とリンクされる BRP を保持しているコンテキスト ID を示しています。WRP が別の BRP とリンクされており、その別の BRP がリンク付きコンテキスト ID 一致用に構成されていない場合、ウォッチポイント デバッグイベントが生成されるかどうかは予測不能です。

表 10-8 ウォッチポイント制御レジスタのビット割り当て （続き）

ビット	名前	説明
[15:14]	セキュア状態アクセス制御	セキュア状態アクセス制御。このフィールドを使用して、ウォッチポイントをプロセッサのセキュリティ状態について条件付きにできます。 b00 = ウォッチポイントは、セキュア状態と非セキュア状態の両方で一致します。 b01 = ウォッチポイントは、非セキュア状態でのみ一致します。 b10 = ウォッチポイントは、セキュア状態でのみ一致します。 b11 = 予約
[13]	-	読み出し時は RAZ、書き込み時は SBZP
[12:9]	-	RAZ/WI
[8:5]	バイトアドレス選択	バイトアドレス選択。WVR にはワードアラインしたアドレスがプログラムされます。このフィールドを使用して、特定のバイトアドレスがアクセスされた場合のみヒットするように、ウォッチポイントをプログラムできます。
[4:3]	L/S	ロード / ストアアクセス。ウォッチポイントを、実行されたアクセスのタイプについて条件付きにできます。 b00 = 予約 b01 = ロード、排他ロード、スワップ b10 = ストア、排他ストア、スワップ b11 = 両方 SWP と SWPB は、b01、b10、b11 の場合にウォッチポイントをトリガします。排他ロード命令は、b01 または b11 の場合にウォッチポイントをトリガします。排他ストア命令は、プロセッサ内部のローカルモニタを通過した場合にのみ、b10 または b11 の場合にウォッチポイントをトリガします。[a]
[2:1]	SP	特権アクセス制御。ウォッチポイントを、実行されるアクセスの特権について条件付きにできます。 b00 = 予約 b01 = 特権付き、プロセッサがメモリに特権アクセスを行った場合のみ一致します。 b10 = ユーザ、非特権アクセスでのみ一致します。 b11 = 両方、すべてのアクセスに一致します。 **注** すべての場合について、一致の条件となるのはアクセスの特権で、プロセッサのモードではありません。
[0]	W	ウォッチポイントイネーブル。 0 = ウォッチポイントが不可能です。これはリセット時の値です。 1 = ウォッチポイントが可能です。

a. 排他ストアでは、MMU フォールトが生成されたり、ローカルモニタの状態にかかわらず、プロセッサでデータウォッチポイント例外が取得されたりする可能性があります。

10.4 デバッグ管理レジスタ

この管理レジスタは、すべての CoreSight コンポーネントに実装されている、標準化されたレジスタのセットを定義します。ここでは、これらのレジスタについて説明します。これらのレジスタにアクセスするには、cp14 インタフェースを使用する必要があります。

Cortex-A9 デバッグユニットの管理レジスタの内容を、表 10-9 に示します。

表 10-9 デバッグ管理レジスタ

オフセット	レジスタ番号	アクセス	ニーモニック	説明
0xD00 ～ 0xDFC	832 ～ 895	RO	-	*「プロセッサ ID レジスタ」*
0xE00 ～ 0xEF0	854 ～ 956	-	-	RAZ
0xF00	960	RW	ITCTRL	-
0xF04 ～ 0xF9C	961 ～ 999	RAZ	-	管理レジスタの拡張用として予約
0xFA0	1000	RW	CLAIMSET	-
0xFA4	1001	RW	CLAIMCLR	-
0xFA8 ～ 0xFBC	1002 ～ 1003	-	-	RAZ
0xFB0	1004	WO	LOCKACCESS	-
0xFB4		RO	LOCKSTATUS	-
0xFB8		RO	AUTHSTATUS	-
0xFBC ～ 0xFC4	1007 ～ 1009	-	-	RAZ
0xFC8	1010	RO	DEVID	デバイス ID
0xFCC	1011	RO	DEVTYPE	-
0xFD0 ～ 0xFFC	1012 ～ 1023	R	-	*「CoreSight 識別レジスタ」*（ページ 10-14）

10.4.1 プロセッサ ID レジスタ

プロセッサ ID レジスタは読み出し専用レジスタで、対応する CP15 ID コードレジスタおよび機能 ID レジスタと同じ値を返します。

それぞれのプロセス ID レジスタに関連付けられているオフセット値、レジスタ番号、ニーモニック、および各レジスタの説明を、表 10-10 に示します。

表 10-10 プロセッサ ID レジスタ

オフセット（16 進数）	レジスタ番号	ニーモニック		レジスタ値	説明
0xD00	832	CPUID	RO	0x80000000	ID コードレジスタ[a]
0xD04	833	CTYPR	RO	0x80038003	キャッシュタイプ レジスタ
0xD08	834	-	RAZ	-	-
0xD0C	835	TTYPR	RO	0x00000400	TLB タイプレジスタ
0xD10 ～ 0xD1C	836 ～ 839	-	-	-	予約
0xD20	840	ID_PFR0	RO	0x00001231	プロセッサ機能レジスタ 0
0xD24	841	ID_PFR1	RO	0x00000011	プロセッサ機能レジスタ 1

表 10-10 プロセッサ ID レジスタ （続き）

オフセット（16 進数）	レジスタ番号	ニーモニック		レジスタ値	説明
0xD28	842	ID_DFR0	RO	0x00010444	デバッグ機能レジスタ 0
0xD2C	843	ID_AFR0	RAZ	-	補助機能レジスタ 0
0xD30	844	ID_MMFR0	RO	0x00100103	メモリモデル機能レジスタ 0
0xD34	845	ID_MMFR1	RO	0x20000000	メモリモデル機能レジスタ 1
0xD38	846	ID_MMFR2	RO	0x01230000	メモリモデル機能レジスタ 2
0xD3C	847	ID_MMFR3	RO	0x00002111	メモリモデル機能レジスタ 3
0xD40	848	ID_ISAR0	RO	0x00101111	命令セット属性レジスタ 0
0xD44	849	ID_ISAR1	RO	0x13112111	命令セット属性レジスタ 1
0xD48	850	ID_ISAR2	RO	0x21232041	命令セット属性レジスタ 2
0xD4C	851	ID_ISAR3	RO	0x11112131	命令セット属性レジスタ 3
0xD50	852	ID_ISAR4	RO	0x00011142	命令セット属性レジスタ 4
0xD54	853	ID_ISAR5	RAZ	-	命令セット属性レジスタ 5

a. ユニプロセッサバージョンの場合 = 0x80000000
マルチプロセッサバージョンの場合 = 0xC0000n0m
n = **CLUSTERID** 入力
m = CPU 番号（CPU0 の場合は 0x0、CPU1 の場合は 0x1、CPU2 の場合は 0x2、CPU3 の場合は 0x3）

10.4.2 CoreSight 識別レジスタ

この識別レジスタは読み出し専用レジスタで、ペリフェラル識別レジスタとコンポーネント識別レジスタで構成されます。ペリフェラル識別レジスタは、すべての CoreSight コンポーネントに必要な標準的な情報を提供します。各レジスタのビット [7:0] のみが使用されます。

コンポーネント識別レジスタは、プロセッサを CoreSight コンポーネントとして識別します。各レジスタのビット [7:0] のみが使用され、他のビットは RAZ です。これらのレジスタの値は固定です。

それぞれのペリフェラル識別レジスタに関連付けられているオフセット値、レジスタ番号、および各レジスタの説明を、表 10-11 に示します。

表 10-11 ペリフェラル識別レジスタ

オフセット（16 進数）	レジスタ番号	値	説明
0xFD0	1012	0x04	ペリフェラル識別レジスタ 4
0xFD4	1013	-	予約
0xFD8	1014	-	予約
0xFDC	1015	-	予約
0xFE0	1016	0x09	ペリフェラル識別レジスタ 0

表 10-11 ペリフェラル識別レジスタ （続き）

オフセット (16 進数)	レジスタ番号	値	説明
0xFE4	1017	0xBC	ペリフェラル識別レジスタ 1
0xFE8	1018	0x0B	ペリフェラル識別レジスタ 2
0xFEC	1019	0x00	ペリフェラル識別レジスタ 3

それぞれのコンポーネント識別レジスタに関連付けられているオフセット値、レジスタ番号、値を、表 10-12 に示します。

表 10-12 コンポーネント識別レジスタ

オフセット (16 進数)	レジスタ番号	値	説明
0xFF0	1020	0x0D	コンポーネント識別レジスタ 0
0xFF4	1021	0x90	コンポーネント識別レジスタ 1
0xFF8	1022	0x05	コンポーネント識別レジスタ 2
0xFFC	1023	0xB1	コンポーネント識別レジスタ 3

10.5 外部デバッグインタフェース

システムは、Cortex-A9 APB スレーブポートを経由して、メモリマップされたデバッグレジスタにアクセスできます。

この APB スレーブインタフェースは、32 ビット幅のデータ、ストール、スレーブ生成アボート、および 2 × 4KB のメモリをマップする 11 のアドレスビット ([12:2]) をサポートします。アクセスされるコンポーネントは、**PADDRDBG[12:0]** のビット [12] で選択されます。

- Cortex-A9 プロセッサのデバッグ領域にアクセスするには、**PADDRDBG[12]** = 0 を使用します。デバッグリソースのメモリマッピングについては、表 10-1（ページ 10-5）を参照して下さい。
- Cortex-A9 プロセッサのパフォーマンス監視ユニット (PMU) 領域にアクセスするには、**PADDRDBG[12]** = 1 を使用します。PMU リソースのメモリマッピングについては、第 11 章 *パフォーマンス監視ユニット*を参照して下さい。

PADDRDBG31 信号は、アクセスのソースをプロセッサに指示します。

外部デバッグ信号の完全な一覧については、付録 A *信号の説明*を参照して下さい。

外部デバッグインタフェース信号を、図 10-4 に示します。

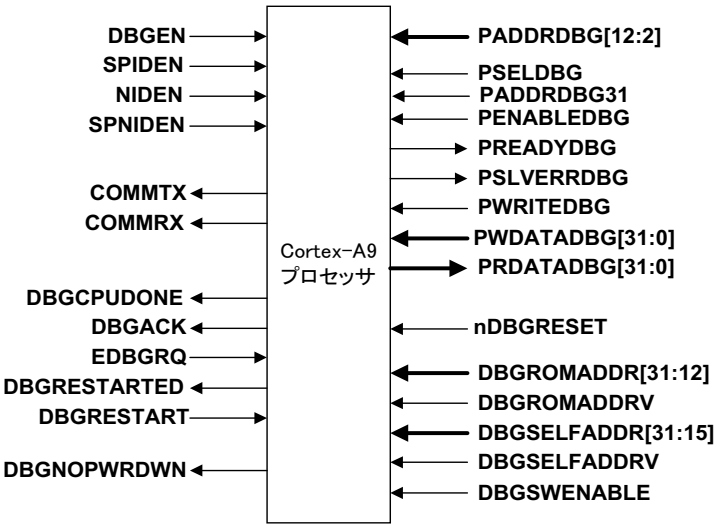


図 10-4 外部デバッグインタフェース信号

10.5.1 認証信号

認証信号と、関連するデバッグアクセス許可との有効な組み合わせの一覧を、表 10-13 に示します。

表 10-13 認証信号の制限

SPIDEN	DBGEN[a]	SPNIDEN	NIDEN	セキュア侵襲性デバッグが許可される[b]	非セキュア侵襲性デバッグが許可される	セキュア非侵襲性デバッグが許可される	非セキュア非侵襲性デバッグが許可される
0	0	0	0	いいえ	いいえ	いいえ	いいえ
0	0	0	1	いいえ	いいえ	いいえ	はい
0	0	1	0	いいえ	いいえ	いいえ	いいえ
0	0	1	1	いいえ	いいえ	はい	はい

表 10-13 認証信号の制限 （続き）

SPIDEN	DBGEN[a]	SPNIDEN	NIDEN	セキュア侵襲性デバッグが許可される[b]	非セキュア侵襲性デバッグが許可される	セキュア非侵襲性デバッグが許可される	非セキュア非侵襲性デバッグが許可される
0	1	0	0	いいえ	はい	いいえ	はい
0	1	0	1	いいえ	はい	いいえ	はい
0	1	1	0	いいえ	はい	はい	はい
0	1	1	1	いいえ	はい	はい	はい
1	0	0	0	いいえ	いいえ	いいえ	いいえ
1	0	0	1	いいえ	いいえ	はい	はい
1	0	1	0	いいえ	いいえ	いいえ	いいえ
1	0	1	1	いいえ	いいえ	はい	はい
1	1	0	0	はい	はい	はい	はい
1	1	0	1	はい	はい	はい	はい
1	1	1	0	はい	はい	はい	はい
1	1	1	1	はい	はい	はい	はい

a. **DBGEN** が LOW の場合、プロセッサは、この信号が LOW のときにホールト デバッグイベントが無視されることを除いて、DBGDSCR[15:14] が b00 であるかのように動作します。
b. 侵襲性デバッグは、コアの動作に影響を与える操作として定義されます。例えば、ブレークポイントの検出は侵襲性デバッグとして定義されますが、パフォーマンスカウンタとトレースは非侵襲性です。

10.5.2 認証信号の変更

NIDEN、**DBGEN**、**SPIDEN**、**SPNIDEN** 入力信号は、特定の値に固定されるか、特定の外部デバイスによって制御されます。

Cortex-A9 プロセッサで実行されているソフトウェアで、認証信号を駆動する外部デバイスを制御する場合は、安全なシーケンスを使用して変更を行う必要があります。

1. 実装固有の命令シーケンスを実行して、信号の値を変更します。例えば、特定の値をシステムペリフェラルの制御レジスタに書き込む、単一の `STR` 命令を実行します。
2. 手順 1 にメモリ操作が含まれる場合は、DSB を発行します。
3. DSCR または認証ステータスレジスタをポーリングし、プロセッサがこれらの信号の値の変化をすでに検出しているかどうかをチェックします。DSB が完了してから数サイクル後まで、システムが信号の変化をプロセッサに発行しない可能性があるため、この操作が必要です。
4. `ISB`、例外開始、または例外終了を実行します。

ソフトウェアは、この手順が完了するまで、認証信号の新しい値に依存するデバッグまたは分析操作を実行できません。デバッガがデバッグ状態で、ITR 経由でプロセッサを制御するときにも、同じ規則が適用されます。

関連する **DBGEN**、**NIDEN**、**SPIDEN**、**SPNIDEN** の値の組み合わせは、DSCR[17:16]、DSCR[15:14]、または認証ステータスレジスタをポーリングすることで判断できます。

10.5.3 デバッグ APB インタフェース

PMU レジスタ名と、対応するデバッグ APB インタフェース上のアドレスを、表 10-14 に示します。

表 10-14 PMU レジスタ名とデバッグ APB インタフェースアドレス

PMU レジスタ名	デバッグ APB アドレス
PMU イベントカウンタ 0	0x000
PMU イベントカウンタ 1	0x004
PMU イベントカウンタ 2	0x008
PMU イベントカウンタ 3	0x00C
PMU イベントカウンタ 4	0x010
PMU イベントカウンタ 5	0x014
pmccntr	0x07C
pmevtyper0	0x400
pmevtyper1	0x404
pmevtyper2	0x408
pmevtyper3	0x40C
pmevtyper4	0x410
pmevtyper5	0x414
pmcntenset	0xC00
pmcntenclr	0xC20
pmintenset	0xC40
pmintenclr	0xC60
pmovsr	0xC80
pmswinc	0xCA0
pmcr	0xE04
pmuserenr	0xE08

10.5.4 外部デバッグ要求インタフェース

次に示すセクションでは、外部デバッグ要求インタフェース信号について説明します。

- *「EDBGRQ」*（ページ 10-19）
- *「DBGACK」*（ページ 10-19）
- *「DBGCPUDONE」*（ページ 10-19）
- *「COMMRX と COMMTX」*（ページ 10-20）
- *「メモリマップされたアクセス、DBGROMADDR、DBGSELFADDR」*（ページ 10-20）

EDBGRQ

この信号は、ホールト デバッグイベントを生成します。つまり、デバッグ状態に移行するようプロセッサに要求します。このイベントが発生すると、デバッグ開始方法ビットの DSCR[5:2] が b0100 にセットされます。**EDBGRQ** がアサートされる場合、**DBGACK** がアサートされるまで保持する必要があります。これを行わない場合、プロセッサの動作は予測不能です。

DBGACK

プロセッサは、システムがデバッグ状態に移行したことを示すため、**DBGACK** をアサートします。これは、**EDBGRQ** 信号のハンドシェークとして機能します。**DBGACK** 信号は、デバッガにより DSCR[10] (DbgAck) ビットが 1 にセットされたときも、HIGH に駆動されます。

DBGCPUDONE

コアがデバッグ状態への移行手順の一部として*データ同期バリア* (DSB) を完了した時点で、**DBGCPUDONE** がアサートされます。

プロセッサは、非デバッグ状態でのメモリアクセスがすべて完了した後にのみ **DBGCPUDONE** をアサートします。このため、システムは、プロセッサによって発行されたすべてのメモリアクセスが、デバッガによって実行された操作の結果であることを示すインジケータとして、**DBGCPUDONE** を使用できます。

デバッグ要求と再起動に固有の Cortex-A9 接続と、CoreSight ピンを、図 10-5 に示します。

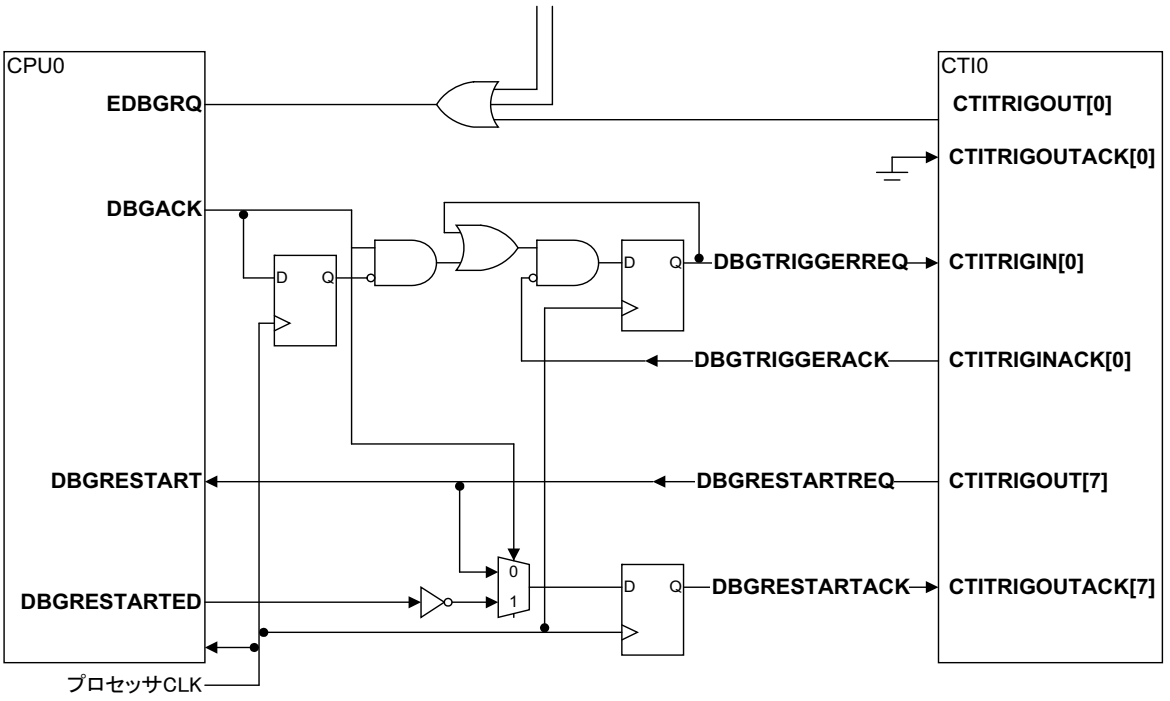


図 10-5 デバッグ要求と再起動に固有の接続

COMMRX と COMMTX

COMMRX および **COMMTX** 出力信号は、DTR 上の割り込み駆動通信を可能にします。これらの信号を割り込みコントローラに接続することによって、チャネル上に新しいデータが存在する場合、またはチャネルがクリアで転送が可能な場合であればいつでも、デバッグ通信チャネルを使用しているソフトウェアに割り込むことができます。

COMMRX は、プロセッサが読み出すデータが CP14 DTR に保持されている場合にアサートされ、プロセッサがそのデータを読み出したときにアサート解除されます。その値は、DBGDSCR[30] DTRRX フルフラグと同じです。

COMMTX は、CP14 で書き込みデータの準備ができたときにアサートされ、プロセッサがデータを書き込んだときにアサート解除されます。その値は、DBGDSCR[29] DTRTX フルフラグの反転と同じです。

メモリマップされたアクセス、DBGROMADDR、DBGSELFADDR

Cortex-A9 プロセッサには、メモリマップされたデバッグインタフェースが存在します。Cortex-A9 プロセッサは、AXI バスを通過するロード命令とストア命令を実行することによって、デバッグレジスタと PMU レジスタにアクセスできます。

DBGROMADDR は、デバッグコンポーネントの物理アドレスを示す ROM テーブルのベースアドレスを示します。

DBGSELFADDR は、ROM テーブルから、プロセッサに内蔵されたレジスタの物理アドレスまでのオフセットを示します。

第 11 章
パフォーマンス監視ユニット

本章では、*パフォーマンス監視ユニット* (PMU) と、PMU で使用可能なレジスタについて説明します。本章は次のセクションから構成されています。

- *「パフォーマンス監視ユニットについて」*（ページ 11-2）
- *「PMU 管理レジスタ」*（ページ 11-3）
- *「パフォーマンス監視イベント」*（ページ 11-7）

11.1 パフォーマンス監視ユニットについて

Cortex-A9 プロセッサの PMU は、プロセッサとメモリシステムの動作に関する統計データを収集する 6 つのカウンタを提供します。それぞれのカウンタで、Cortex-A9 プロセッサで使用可能な 58 のイベントのいずれかをカウントできます。

PMU カウンタと、関連する制御レジスタには、内部の CP15 インタフェースだけでなく、デバッグ APB インタフェースからもアクセスできます。PMU レジスタのマッピングを、表 11-1 に示します。

表 11-1 パフォーマンス監視命令とデバッグ APB のマッピング

デバッグ APB インタフェースのマッピング	CP15 命令	アクセス	リセット	名前
0x000	0, c9, c13, 2	RW	-	PMXEVCNTR0
0x004	0, c9, c13, 2	RW	-	PMXEVCNTR1
0x008	0, c9, c13, 2	RW	-	PMXEVCNTR2
0x00C	0, c9, c13, 2	RW	-	PMXEVCNTR3
0x010	0, c9, c13, 2	RW	-	PMXEVCNTR4
0x014	0, c9, c13, 2	RW	-	PMXEVCNTR5
0x07C	0, c9, c13, 0	RW	-	PMCCNTR
0x400	0, c9, c13, 1	RW	-	PMXEVTYPER0
0x404	0, c9, c13, 1	RW	-	PMXEVTYPER1
0x408	0, c9, c13, 1	RW	-	PMXEVTYPER2
0x40C	0, c9, c13, 1	RW	-	PMXEVTYPER3
0x410	0, c9, c13, 1	RW	-	PMXEVTYPER4
0x414	0, c9, c13, 1	RW	-	PMXEVTYPER5
0xC00	0, c9, 0 c12, 1	RW	-	PMCNTENSET
0xC20	0, c9, c12, 2	RW	-	PMCNTENCLR
0xC40	0, c9, c14, 1	RW	-	PMINTENSET
0xC60	0, c9, c14, 2	RW	-	PMINTENCLR
0xC80	0, c9, c12, 3	RW	-	PMOVSR
0xCA0	0, c9, c12, 4	WO	-	PMSWINC
0xE04	0, c9, c12, 0	RW	0x41093000	PMCR
0xE08	0, c9, c14, 0	RW[a]	0x00000000	PMUSERENR
-	0, c9, c12, 5	RW	-	PMSELR

a. ユーザモードでは読み出し専用です。

11.2 PMU 管理レジスタ

PMU 管理レジスタは、すべての CoreSight コンポーネントに実装された、標準化されたレジスタのセットを定義します。ここでは、これらのレジスタについて説明します。これらのレジスタにアクセスするには、cp14 インタフェースを使用する必要があります。

Cortex-A9 デバッグユニットの管理レジスタの内容を、表 11-2 に示します。

表 11-2 PMU 管理レジスタ

オフセット	レジスタ番号	アクセス	ニーモニック	説明
0xD00 ~ 0xDFC	832 ~ 895	RO	-	*「プロセッサ ID レジスタ」*
0xE00 ~ 0xEF0	854 ~ 956	-	-	RAZ
0xF00	960	RW	ITCTRL	-
0xF04 ~ 0xF9C	961 ~ 999	RAZ	-	管理レジスタの拡張用として予約
0xFA0	1000	RW	CLAIMSET	-
0xFA4	1001	RW	CLAIMCLR	-
0xFA8 ~ 0xFBC	1002 ~ 1003	-	-	RAZ
0xFB0	1004	WO	LOCKACCESS	-
0xFB4		RO	LOCKSTATUS	-
0xFB8		RO	AUTHSTATUS	-
0xFBC ~ 0xFC4	1007 ~ 1009	-	-	RAZ
0xFC8	1010	RO	DEVID	デバイス ID
0xFCC	1011	RO	DEVTYPE	-
0xFD0 ~ 0xFFC	1012 ~ 1023	R	-	*「CoreSight 識別レジスタ」*（ページ 11-4）

11.2.1 プロセッサ ID レジスタ

プロセッサ ID レジスタは読み出し専用レジスタで、対応する CP15 ID コードレジスタおよび機能 ID レジスタと同じ値を返します。

それぞれのプロセッサ ID レジスタに関連付けられたオフセット値、レジスタ番号、ニーモニック、および説明を、表 11-3 に示します。

表 11-3 プロセッサ ID レジスタ

オフセット（16 進数）	レジスタ番号	ニーモニック	アクセス	レジスタ値	説明
0xD00	832	CPUID	RO	0x80000000	ID コードレジスタ[a]
0xD04	833	CTYPR	RO	0x80038003	キャッシュタイプ レジスタ
0xD08	834	-	RAZ	-	-
0xD0C	835	TTYPR	RO	0x00000400	TLB タイプレジスタ
0xD10 ~ 0xD1C	836 ~ 839	-	-	-	予約
0xD20	840	ID_PFR0	RO	0x00001231	プロセッサ機能レジスタ 0
0xD24	841	ID_PFR1	RO	0x00000011	プロセッサ機能レジスタ 1

表 11-3 プロセッサ ID レジスタ （続き）

オフセット（16 進数）	レジスタ番号	ニーモニック	アクセス	レジスタ値	説明
0xD28	842	ID_DFR0	RO	0x00010444	デバッグ機能レジスタ 0
0xD2C	843	ID_AFR0	RAZ	-	補助機能レジスタ 0
0xD30	844	ID_MMFR0	RO	0x00100103	メモリモデル機能レジスタ 0
0xD34	845	ID_MMFR1	RO	0x20000000	メモリモデル機能レジスタ 1
0xD38	846	ID_MMFR2	RO	0x01230000	メモリモデル機能レジスタ 2
0xD3C	847	ID_MMFR3	RO	0x00002111	メモリモデル機能レジスタ 3
0xD40	848	ID_ISAR0	RO	0x00101111	命令セット属性レジスタ 0
0xD44	849	ID_ISAR1	RO	0x13112111	命令セット属性レジスタ 1
0xD48	850	ID_ISAR2	RO	0x21232041	命令セット属性レジスタ 2
0xD4C	851	ID_ISAR3	RO	0x11112131	命令セット属性レジスタ 3
0xD50	852	ID_ISAR4	RO	0x00011142	命令セット属性レジスタ 4
0xD54	853	ID_ISAR5	RAZ	-	命令セット属性レジスタ 5

a. ユニプロセッサバージョンの場合 = 0x80000000
マルチプロセッサバージョンの場合 = 0xC0000n0m
n = **CLUSTERID** 入力
m = CPU 番号（CPU0 の場合は 0x0、CPU1 の場合は 0x1、CPU2 の場合は 0x2、CPU3 の場合は 0x3）

11.2.2 CoreSight 識別レジスタ

この識別レジスタは読み出し専用レジスタで、ペリフェラル識別レジスタとコンポーネント識別レジスタで構成されます。ペリフェラル識別レジスタは、すべての CoreSight コンポーネントに必要な標準的な情報を提供します。各レジスタのビット [7:0] のみが使用されます。

コンポーネント識別レジスタは、プロセッサを CoreSight コンポーネントとして識別します。各レジスタのビット [7:0] のみが使用され、他のビットは RAZ です。これらのレジスタの値は固定です。

それぞれのペリフェラル識別レジスタに関連付けられたオフセット値、レジスタ番号、値、および説明を、表 11-4 に示します。

表 11-4 ペリフェラル識別レジスタ

オフセット（16 進数）	レジスタ番号	値	説明
0xFD0	1012	0x04	ペリフェラル識別レジスタ 4
0xFD4	1013	-	予約
0xFD8	1014	-	予約
0xFDC	1015	-	予約
0xFE0	1016	0xA0	ペリフェラル識別レジスタ 0

表 11-4 ペリフェラル識別レジスタ （続き）

オフセット (16 進数)	レジスタ番号	値	説明
0xFE4	1017	0xB9	ペリフェラル識別レジスタ 1
0xFE8	1018	0x0B	ペリフェラル識別レジスタ 2
0xFEC	1019	0x00	ペリフェラル識別レジスタ 3

それぞれのコンポーネント識別レジスタに関連付けられたオフセット値、レジスタ番号、値を、表 11-5 に示します。

表 11-5 コンポーネント識別レジスタ

オフセット (16 進数)	レジスタ番号	値	説明
0xFF0	1020	0x0D	コンポーネント識別レジスタ 0
0xFF4	1021	0x90	コンポーネント識別レジスタ 1
0xFF8	1022	0x05	コンポーネント識別レジスタ 2
0xFFC	1023	0xB1	コンポーネント識別レジスタ 3

11.2.3 PMU APB インタフェース

PMU レジスタ名と、対応する APB インタフェースのアドレスを、表 11-6 に示します。

表 11-6 PMU レジスタ名と APB アドレス

PMU レジスタ名	デバッグ APB アドレス
PMU イベントカウンタ 0	0x000
PMU イベントカウンタ 1	0x004
PMU イベントカウンタ 2	0x008
PMU イベントカウンタ 3	0x00C
PMU イベントカウンタ 4	0x010
PMU イベントカウンタ 5	0x014
pmccntr	0x07C
pmevtyper0	0x400
pmevtyper1	0x404
pmevtyper2	0x408
pmevtyper3	0x40C
pmevtyper4	0x410
pmevtyper5	0x414
pmcntenset	0xC00
pmcntenclr	0xC20
pmintenset	0xC40
pmintenclr	0xC60

表 11-6 PMU レジスタ名と APB アドレス （続き）

PMU レジスタ名	デバッグ APB アドレス
pmovsr	0xC80
pmswinc	0xCA0
pmcr	0xE04
pmuserenr	0xE08

11.3 パフォーマンス監視イベント

Cortex-A9 プロセッサは、次の例外を除いて、『ARM アーキテクチャ リファレンスマニュアル』に記載されているアーキテクチャイベントを実装しています。

0x08 アーキテクチャ的に実行されたメモリ読み出し命令

0x0E アーキテクチャ的に実行された、例外からの復帰以外のプロシージャからの復帰

イベントと、対応する **PMUEVENT** 信号については、表 A-18 （ページ A-14）を参照して下さい。

PMU は、この他に、Cortex-A9 固有のイベントのセットを提供します。

11.3.1 Cortex-A9 固有のイベント

Cortex-A9 固有のイベントを、表 11-7 に示します。表 11-7 の値列の「正確」は、イベントが正確にカウントされることを意味します。ストールや投機的な命令に関連したイベントは、同じ列の「近似」で示されます。

表 11-7 Cortex-A9 固有のイベント

イベント	説明	値
0x40	Java バイトコードの実行[a] 投機的なものも含めて、デコードされた Java バイトコードの数をカウントします。	近似
0x41	実行されたソフトウェア Java バイトコード[a] 投機的なものも含めて、デコードされたソフトウェア Java バイトコードの数をカウントします。	近似
0x42	実行された Jazelle 後方分岐[a] Jazelle で、分岐を行うものとして実行された分岐の数をカウントします。これには、過去のロード / ストアの遅延アボートが原因でフラッシュされた分岐が含まれます。	近似
0x50	コヒーレントラインフィル ミス[b] Cortex-A9 プロセッサで実行され、他の Cortex-A9 プロセッサすべてでミスしたコヒーレントラインフィル要求の数をカウントします。これは、その要求が外部メモリに送信されたことを意味します。	正確
0x51	コヒーレントラインフィル ヒット[b] Cortex-A9 プロセッサで実行され、他の Cortex-A9 プロセッサでヒットした、コヒーレントラインフィル要求の数をカウントします。これは、ラインフィルデータが該当の Cortex-A9 キャッシュから直接フェッチされたことを意味します。	正確
0x60	命令キャッシュ依存ストールサイクル プロセッサで新しい命令を受け入れる準備はできているが、命令側の準備ができていないためにプロセッサで命令が受信されず、命令キャッシュが 1 つ以上のラインフィルを実行しているサイクル数をカウントします。	近似
0x61	データキャッシュ依存ストールサイクル コアにどのパイプラインにも発行できない命令が存在し、ロード / ストアユニットに保留中の TLB 要求はないが、保留中のラインフィル要求が 1 つ以上存在するサイクル数をカウントします。	近似
0x62	メイン TLB ミスストール サイクル プロセッサが、メイン TLB からの変換テーブルウォークの完了を待ちながらストールしているサイクル数をカウントします。プロセッサのストールは、命令側で命令が提供できない、データ側で必要なデータが提供できない、またはメイン TLB 変換テーブルウォークの完了を待っている場合に発生する可能性があります。	近似
0x63	STREX 成功 アーキテクチャ的に実行され、成功した STREX 命令の数をカウントします。	正確

表 11-7 Cortex-A9 固有のイベント （続き）

イベント	説明	値
0x64	STREX 失敗 アーキテクチャ的に実行され、失敗した STREX 命令の数をカウントします。	正確
0x65	データ退出 データキャッシュのラインフィルが原因の退出要求の数をカウントします。	正確
0x66	発行で命令がディスパッチされない 発行ステージが空か、命令をディスパッチできないために、どの命令もディスパッチされなかったサイクル数をカウントします。	正確
0x67	発行が空 発行ステージが空であるサイクル数をカウントします。	正確
0x68	コア名前変更ステージからの命令 レジスタ名変更ステージを通過した命令の数をカウントします。この数は、投機的に発行された命令の総数の近似で、アーキテクチャ的に発行された命令の総数のより正確な近似です。この近似は、主に、分岐予測失敗率に依存します。 名前変更ステージは、1 つのサイクルで 2 つの命令を処理できるため、イベントは 2 ビット長です。 • b00 = コア名前変更ステージからの命令はありません。 • b01 = コア名前変更ステージからの命令が 1 つ存在します。 • b10 = コア名前変更ステージからの命令が 2 つ存在します。 これらの値の PMUEVENT バスビットへのマップ方法については、表 A-17 （ページ A-14）を参照して下さい。	近似
0x6E	予測可能関数からの復帰 例外からの復帰以外で、条件コードが失敗しなかったプロシージャからの復帰の数をカウントします。このカウントには、以前のロード / ストアの遅延アボートが原因でフラッシュされた、プロシージャからの復帰が含まれます。 次の命令のみが報告されます。 • `BX R14` • `MOV PC LR` • `POP {..,pc}` • `LDR pc,[sp],#offset` 次の命令は報告されません。 • `LDMIA R9!,{..,PC}`（ThumbEE 状態のみ） • `LDR PC,[R9],#offset`（ThumbEE 状態のみ） • `BX R0 (Rm != R14)` • `MOV PC,R0 (Rm != R14)` • `LDM SP,{...,PC}`（ライトバックが未指定） • `LDR PC,[SP,#offset]`（不正なアドレシングモード）	近似
0x70	メイン実行ユニット命令 プロセッサのメイン実行パイプライン、乗算パイプライン、算術論理演算ユニットパイプラインで実行された命令の数をカウントします。カウントされた命令は投機的でもあります。	近似
0x71	第 2 実行ユニット命令 プロセッサの第 2 実行パイプライン (ALU) で実行された命令の数をカウントします。カウントされた命令は投機的でもあります。	近似
0x72	ロード / ストア命令 ロード / ストアユニットで実行された命令の数をカウントします。カウントされた命令は投機的でもあります。	近似

表 11-7 Cortex-A9 固有のイベント （続き）

イベント	説明	値
0x73	浮動小数点命令 レジスタ名変更ステージを通過した浮動小数点命令の数をカウントします。命令はこのステージでも投機的です。 1 つのサイクルで 2 つの浮動小数点命令を名前変更できるため、イベントは 2 ビット長です。 0b00 = どの浮動小数点命令も名前変更されません。 0b01 = 1 つの浮動小数点命令が名前変更されます。 0b10 = 2 つの浮動小数点命令が名前変更されます。 これらの値の PMUEVENT バスビットへのマップ方法については、表 A-17 （ページ A-14）を参照して下さい。	近似
0x74	NEON 命令 レジスタ名変更ステージを通過した NEON 命令の数をカウントします。命令はこのステージでも投機的です。 1 つのサイクルで 2 つの NEON 命令を名前変更できるため、イベントは 2 ビット長です。 0b00 = どの NEON 命令も名前変更されません。 0b01 = 1 つの NEON 命令が名前変更されます。 0b10 = 2 つの NEON 命令が名前変更されます。 これらの値の PMUEVENT バスビットへのマップ方法については、表 A-17 （ページ A-14）を参照して下さい。	近似
0x80	PLD が原因のプロセッサストール PLD スロットがすべていっぱいなため、プロセッサがストールしたサイクル数をカウントします。	近似
0x81	メモリへの書き込みが原因のプロセッサストール データ側がいっぱいで、外部メモリへの書き込みを実行中なため、プロセッサとデータ側の両方がストールしたサイクル数をカウントします。	近似
0x82	命令側のメイン TLB ミスが原因のプロセッサストール 命令側で発行された要求に対するメイン TLB ミスが原因の、ストールサイクル数をカウントします。	近似
0x83	データ側のメイン TLB ミスが原因のプロセッサストール データ側で発行された要求に対するメイン TLB ミスが原因の、ストールサイクル数をカウントします。	近似
0x84	命令マイクロ TLB ミスが原因のプロセッサストール 命令側のマイクロ TLB ミスが原因の、ストールサイクル数をカウントします。このイベントには、対応するメイン TLB イベントでカウント済みの、メイン TLB ミスによるストールサイクルは含まれません。	近似
0x85	データマイクロ TLB ミスが原因のプロセッサストール データ側のマイクロ TLB ミスが原因の、ストールサイクル数をカウントします。このイベントには、対応するメイン TLB イベントでカウント済みの、メイン TLB ミスによるストールサイクルは含まれません。	近似
0x86	DMB が原因のプロセッサストール DMB メモリバリアの実行が原因のストールサイクル数をカウントします。これには、投機的であっても、実行されたすべての DMB 命令が含まれます。	近似
0x8A	整数クロック稼働 整数コアクロックが稼働しているサイクル数をカウントします。	近似
0x8B	データエンジン クロック稼働 データエンジン クロックが稼働しているサイクル数をカウントします。	近似
0x90	ISB 命令 アーキテクチャ的に実行された ISB 命令の数をカウントします。	正確

表 11-7 Cortex-A9 固有のイベント （続き）

イベント	説明	値
0x91	DSB 命令 アーキテクチャ的に実行された DSB 命令の数をカウントします。	正確
0x92	DMB 命令 アーキテクチャ的に実行された DMB 命令の数をカウントします。	近似
0x93	外部割り込み プロセッサで実行された外部割り込みの数をカウントします。	近似
0xA0	完了した PLE キャッシュライン要求[c]	正確
0xA1	スキップされた PLE キャッシュライン要求[c]	正確
0xA2	PLE FIFO フラッシュ[c]	正確
0xA3	完了した PLE 要求[c]	正確
0xA4	PLE FIFO オーバフロー[c]	正確
0xA5	プログラムされた PLE 要求[c]	正確

a. 設計に Jazelle 拡張機能が実装されている場合のみ。そうでない場合は、0 として読み出されます。
b. Cortex-A9 のマルチプロセッサバリアントで使用されます。
c. PLE が存在する場合にのみアクティブになります。そうでない場合は、0 として読み出されます。

付録 A
信号の説明

この付録では、Cortex-A9 の信号を一覧表で説明します。本章は次のセクションから構成されています。

A.1 クロック信号とクロック制御信号

Cortex-A9 プロセッサには、外部的に生成されるグローバルクロックが 1 つ存在します。クロック信号とクロック制御信号を、表 A-1 に示します。

表 A-1 Cortex-A9 のクロック信号とクロック制御信号

名前	I/O	ソース	説明
CLK	I	クロックコントローラ	グローバルクロック。 *「クロックとリセット」*（ページ 2-6）を参照して下さい。
MAXCLKLATENCY[2:0]	I	実装固有の静的な値	動的なクロックゲート遅延を制御します。 このピンは、プロセッサのリセット中にサンプリングされます。 *「動的高水準クロックゲート」*（ページ 2-8）を参照して下さい。

A.2 リセットとリセット制御

リセット信号とリセット制御信号を、表 A-2 に示します。

表 A-2 Cortex-A9 プロセッサのリセット信号

名前	I/O	ソース	説明
nCPURESET	I	リセットコントローラ	Cortex-A9 プロセッサのリセット
nDBGRESET	I		Cortex-A9 プロセッサのデバッグロジックのリセット
NEONCLKOFF[a]	I		MPE SIMD ロジッククロック制御。 0 = MPE SIMD ロジッククロックを停止しません。 1 = MPE SIMD ロジッククロックを停止します。
nNEONRESET[a]	I		Cortex-A9 MPE SIMD ロジックのリセット

a. MPE が存在する場合のみ

「リセット」（ページ 2-6）を参照して下さい。

A.3 割り込み

割り込みライン信号を、表 A-3 に示します。

表 A-3 割り込みライン信号

名前	I/O	ソース	説明
nFIQ	I	割り込みソース	Cortex-A9 プロセッサの FIQ 要求入力ライン。 アクティブ LOW の高速割り込み要求が、次のように設定されます。 0 = 高速割り込みをアクティブにします。 1 = 高速割り込みを非アクティブにします。 プロセッサは、**nFIQ** 入力をレベル感知として扱います。
nIRQ	I	割り込みソース	Cortex-A9 プロセッサの IRQ 要求入力ライン。 アクティブ LOW の割り込み要求が、次のように設定されます。 0 = 割り込みをアクティブにします。 1 = 割り込みを非アクティブにします。 プロセッサは、**nIRQ** 入力をレベル感知として扱います。

A.4 構成信号

プロセッサのリセット中にのみサンプリングされる構成信号を、表 A-4 に示します。

表 A-4 構成信号

名前	I/O	ソース	説明
CFGEND	I	システム構成制御	リセット時に、SCTLR の EE ビットの状態を制御します。 0 = EE ビットは LOW です。 1 = EE ビットは HIGH です。
CFGNMFI	I		高速割り込みをマスク不能に構成します。 0 = CP15 c1 制御レジスタの NMFI ビットをクリアします。 1 = CP15 c1 制御レジスタの NMFI ビットをセットします。
TEINIT	I		デフォルトの例外処理状態。 0 = ARM 1 = Thumb リセット時に、SCTLR.TE ビットがセットされます。
VINITHI	I		リセット時の例外ベクタの位置を制御します。 0 = アドレス `0x00000000` で例外ベクタを開始します。 1 = アドレス `0xFFFF0000` で例外ベクタを開始します。 SCTLR.V ビットがセットされます。

CP15SDISABLE 信号を、表 A-5 に示します。

表 A-5 CP15SDISABLE 信号

名前	I/O	ソース	説明
CP15SDISABLE	I	セキュリティコントローラ	セキュア状態で、一部のシステム制御プロセッサレジスタへの書き込みアクセスを不可能にします。 0 = 不可能 1 = 可能 *「システム制御レジスタ」*（ページ 4-15）を参照して下さい。

A.5 スタンバイ信号とイベント待ち信号

スタンバイ信号とイベント待ち信号を、表 A-6 に示します。

表 A-6 スタンバイ信号とイベント待ち信号

名前	I/O	ソースまたはデスティネーション	説明
EVENTI	I	外部コヒーレントエージェント	Cortex-A9 プロセッサを WFE 状態からウェークアップさせるためのイベント入力
EVENTO	O		イベント出力。この信号は、SEV 命令が実行された 1 プロセッサクロック サイクルについて、アクティブ HIGH になります。
STANDBYWFI	O	電力コントローラ	プロセッサが WFI 状態かどうかを示します。 0 = プロセッサがイベント待ち状態ではありません。 1 = プロセッサがイベント待ち状態です。
STANDBYWFE	O		プロセッサが WFE 状態かどうかを示します。 0 = プロセッサがイベント待ち状態ではありません。 1 = プロセッサがイベント待ち状態です。

「スタンバイモード」（ページ 2-11）を参照して下さい。

A.6 電力管理信号

電力管理信号を、表 A-7 に示します。

表 A-7 電力管理信号

名前	I/O	ソース	説明
CPURAMCLAMP	I	電力コントローラ	CPU RAM インタフェースクランプをアクティブにします。 0 = クランプは非アクティブです。 1 = クランプはアクティブです。
NEONCLAMP[a]	I		Cortex-A9 MPE SIMD ロジッククランプをアクティブにします。 0 = クランプは非アクティブです。 1 = クランプはアクティブです。

a. MPE が存在する場合のみ

「電力管理」（ページ 2-10）を参照して下さい。

A.7 AXI インタフェース

Cortex-A9 設計には、2 つの AXI マスタポートを含めることができます。次に示すセクションでは、AXI インタフェースについて説明します。

- *「AXI Master0 信号のデータアクセス」*
- *「AXI Master1 信号の命令アクセス」*（ページ A-11）

A.7.1 AXI Master0 信号のデータアクセス

次に示すセクションでは、データの読み出し / 書き込みアクセスに使用される AXI Master0 インタフェース信号について説明します。

- *「AXI Master0 の書き込みアドレス信号」*
- *「書き込みデータチャネル信号」*（ページ A-9）
- *「書き込み応答チャネル信号」*（ページ A-10）
- *「読み出しデータチャネル信号」*（ページ A-10）
- *「読み出しデータチャネル信号」*（ページ A-11）
- *「AXI Master0 のクロックイネーブル信号」*（ページ A-11）

AXI Master0 の書き込みアドレス信号

AXI Master0 の AXI 書き込みアドレス信号を、表 A-8 に示します。

表 A-8 AXI Master0 の AXI-AW 信号

名前	I/O	ソースまたはデスティネーション	説明
AWADDRM0[31:0]	O	AXI システムデバイス	アドレス
AWBURSTM0[1:0]	O		バーストタイプ = b01、INCR インクリメントバースト
AWCACHEM0[3:0]	O		キャッシュタイプ。メモリ領域のメモリタイプと外部キャッシュ方式によって決定される、キャッシュ可能属性に関する追加情報を提供します。
AWIDM0[1:0]	O		要求 ID

表 A-8 AXI Master0 の AXI-AW 信号 （続き）

名前	I/O	ソースまたはデスティネーション	説明
AWLENM0[3:0]	O	AXI システムデバイス	各バーストで生成可能なデータ転送数
AWLOCKM0[1:0]	O		ロックタイプ
AWPROTM0[2:0]	O		保護タイプ
AWREADYM0	I		アドレス準備完了
AWSIZEM0[1:0]	O		データ転送サイズ。 b000 = 8 ビット転送 b001 = 16 ビット転送 b010 = 32 ビット転送 b011 = 64 ビット転送
AWUSERM0[8:0]	O		[8] 早期 **BRESP**。L2C-310 で使用されます。 [7] 0 のフルライン書き込み。L2C-310 で使用されます。 [6] クリーニング退出 [5] レベル 1 退出 [4:1] メモリタイプと内部キャッシュ方式 b0000 = ストロングリオーダ b0001 = デバイス b0011 = ノーマルメモリ、キャッシュ不可 b0110 = ライトスルー b0111 = ライトバック、書き込み割り当てなし b1111 = ライトバック、書き込み割り当て [0] 共有
AWVALIDM0	O		アドレス有効

書き込みデータチャネル信号

AXI Master0 の AXI 書き込みデータ信号を、表 A-9 に示します。

表 A-9 AXI Master0 の AXI-W 信号

名前	I/O	ソースまたは デスティネーション	説明
WDATAM0[63:0]	O	AXI システムデバイス	書き込むデータ
WIDM0[1:0]	O		書き込み ID
WLASTM0	O		書き込み最終指示
WREADYM0	I		書き込み準備完了
WSTRBM0[7:0]	O		書き込みバイトレーン ストローブ
WVALIDM0	O		書き込み有効

書き込み応答チャネル信号

AXI Master0 の AXI 書き込み応答信号を、表 A-10 に示します。

表 A-10 AXI Master0 の AXI-B 信号

名前	I/O	ソースまたはデスティネーション	説明
BIDM0[1:0]	I	AXI システムデバイス	応答 ID
BREADYM0	O		応答準備完了
BRESPM0[1:0]	I		書き込み応答
BVALIDM0	I		応答有効

読み出しデータチャネル信号

AXI Master0 の AXI 読み出しアドレス信号を、表 A-11 に示します。

表 A-11 AXI Master0 の AXI-AR 信号

名前	I/O	ソースまたはデスティネーション	説明
ARADDRM0[31:0]	O	AXI システムデバイス	アドレス
ARBURSTM0[1:0]	O		バーストタイプ。 b01 = INCR インクリメントバースト b10 = WRAP ラップバースト
ARCACHEM0[3:0]	O		キャッシュタイプで、キャッシュ可能属性に関する追加情報を提供します。
ARIDM0[1:0]	O		要求 ID
ARLENM0[3:0]	O		各バースト内で発生可能なデータ転送の数
ARLOCKM0[1:0]	O		ロックタイプ
ARPROTM0[2:0]	O		保護タイプ
ARREADYM0	I		アドレス準備完了
ARSIZEM0[1:0]	O	AXI システムデバイス	バーストサイズ。 b000 = 8 ビット転送 b001 = 16 ビット転送 b010 = 32 ビット転送 b011 = 64 ビット転送
ARUSERM0[4:0]	O		[4:1] メモリタイプと内部キャッシュ方式。 b0000 = ストロングリオーダ b0001 = デバイス b0011 = ノーマルメモリ、キャッシュ不可 b0110 = ライトスルー b0111 = ライトバック、書き込み割り当てなし b1111 = ライトバック、書き込み割り当て [0] 共有
ARVALIDM0	O		アドレス有効

読み出しデータチャネル信号

AXI Master0 の AXI 読み出しデータ信号を、表 A-12 に示します。

表 A-12 AXI Master0 の AXI-R 信号

名前	I/O	ソースまたはデスティネーション	説明
RVALIDM0	I	AXI システムデバイス	読み出し有効
RDATAM0[63:0]	I		読み出しデータ
RRESPM0[1:0]	I		読み出し応答
RLASTM0	I		読み出し最終指示
RIDM0[1:0]	I		読み出し ID
RREADYM0	O		読み出し準備完了

AXI Master0 のクロックイネーブル信号

ここでは、AXI Master0 のクロックイネーブル信号について説明します。AXI Master0 のクロックイネーブル信号を、表 A-13 に示します。

表 A-13 AXI Master0 のクロックイネーブル信号

名前	I/O	ソース	説明
ACLKENM0	I	クロックコントローラ	AXI バスのクロックイネーブルで、AXI インタフェースをシステムクロックの整数比で動作可能にします。 *「クロックとリセット」*（ページ 2-6）を参照して下さい。

A.7.2 AXI Master1 信号の命令アクセス

次に示すセクションでは、命令アクセスに使用される AXI Master1 インタフェース信号について説明します。

- *「読み出しデータチャネル信号」*
- *「読み出しデータチャネル信号」*（ページ A-13）
- *「AXI Master1 のクロックイネーブル信号」*（ページ A-13）

読み出しデータチャネル信号

AXI Master1 の AXI 読み出しアドレス信号を、表 A-14 （ページ A-12）に示します。

表 A-14 AXI Master1 の AXI-AR 信号

名前	I/O	デスティネーション	説明
ARADDRM1[31:0]	O	AXI システムデバイス	アドレス
ARBURSTM1[1:0]	O		バーストタイプ。 b01 = INCR インクリメントバースト b10 = WRAP ラップバースト
ARCACHEM1[3:0]	O		キャッシュタイプで、キャッシュ可能属性に関する追加情報を提供します。
ARIDM1[5:0]	O		要求 ID
ARLENM1[3:0]	O		各バースト内で発生可能なデータ転送の数
ARLOCKM1[1:0]	O		ロックタイプ。 b00 = 通常アクセス
ARPROTM1[2:0]	O		保護タイプ
ARREADYM1	I		アドレス準備完了
ARSIZEM1[1:0]	O	AXI システムデバイス	バーストサイズ。 b000 = 8 ビット転送 b001 = 16 ビット転送 b010 = 32 ビット転送 b011 = 64 ビット転送
ARUSERM1[4:0]	O		[4:1] = 内部属性 b0000 = ストロングリオーダ b0001 = デバイス b0011 = ノーマルメモリ、キャッシュ不可 b0110 = ライトスルー b0111 = ライトバック、書き込み割り当てなし b1111 = ライトバック、書き込み割り当て [0] = 共有
ARVALIDM1	O		アドレス有効

読み出しデータチャネル信号

AXI Master1 の AXI 読み出しデータ信号を、表 A-15 に示します。

表 A-15 AXI Master1 の AXI-R 信号

名前	I/O	ソースまたはデスティネーション	説明
RVALIDM1	I	AXI システムデバイス	読み出し有効
RDATAM1[63:0]	I		読み出しデータ
RRESPM1[1:0]	I		読み出し応答
RLASTM1	I		読み出し最終指示
RIDM1[5:0]	I		読み出し ID
RREADYM1	O		読み出し準備完了

AXI Master1 のクロックイネーブル信号

ここでは、AXI Master1 のクロックイネーブル信号について説明します。AXI Master1 のクロックイネーブル信号を、表 A-16 に示します。

表 A-16 AXI Master1 のクロックイネーブル信号

名前	I/O	ソース	説明
ACLKENM1	I	クロックコントローラ	AXI バスのクロックイネーブルで、AXI インタフェースをシステムクロックの整数比で動作可能にします。 *「クロックとリセット」*（ページ 2-6）を参照して下さい。

第 8 章 *レベル 2 メモリインタフェース*を参照して下さい。

A.8 パフォーマンス監視信号

パフォーマンス監視信号を、表 A-17 に示します。

表 A-17 パフォーマンス監視信号

名前	I/O	デスティネーション	説明
PMUEVENT[57:0]	O	PTM または外部監視ユニット	パフォーマンス監視ユニット イベントバス。表 A-18 を参照して下さい。
PMUIRQ	O		パフォーマンス監視ユニットの割り込み信号
PMUSECURE	O		Cortex-A9 プロセッサの状態を示します。 0 = 非セキュア状態 1 = セキュア状態 この信号は、CoreSight トレース配信インフラストラクチャに入力されません。
PMUPRIV	O		Cortex-A9 プロセッサの状態を示します。 0 = ユーザモード 1 = 特権モード この信号は、CoreSight トレース配信インフラストラクチャに入力されません。

PMUEVENT 信号と、そのイベント番号との関係を、表 A-18 に示します。

表 A-18 イベント信号とイベント番号

名前	イベント番号	説明
PMUEVENT[0]	0x00	ソフトウェアインクリメント
PMUEVENT[1]	0x01	命令キャッシュミス
PMUEVENT[2]	0x02	命令マイクロ TLB ミス
PMUEVENT[3]	0x03	データキャッシュ ミス
PMUEVENT[4]	0x04	データキャッシュ アクセス
PMUEVENT[5]	0x05	データマイクロ TLB ミス
PMUEVENT[6]	0x06	データ読み出し
PMUEVENT[7]	0x07	データ書き込み
-	0x08	未使用[a]
PMUEVENT[8]	0x68	b00 = どの命令も名前変更されません。 b01 = 1 つの命令が名前変更されます。 b10 = 2 つの命令が名前変更されます。
PMUEVENT[9]		
PMUEVENT[10]	0x09	例外検出
PMUEVENT[11]	0x0A	例外からの復帰
PMUEVENT[12]	0x0B	書き込みコンテキスト ID
PMUEVENT[13]	0x0C	ソフトウェアでの PC 変更
PMUEVENT[14]	0x0D	イミディエート分岐
-	0x0E	未使用[b]
PMUEVENT[15]	0x6E	予測可能な関数からの復帰[b]

表 A-18 イベント信号とイベント番号 （続き）

名前	イベント番号	説明
PMUEVENT[16]	0x0F	アンアラインド
PMUEVENT[17]	0x10	予測に失敗したか、予測されなかった分岐
エクスポートなし	0x11	サイクルカウント
PMUEVENT[18]	0x12	予測可能な分岐
PMUEVENT[19]	0x40	Java バイトコード
PMUEVENT[20]	0x41	ソフトウェア Java バイトコード
PMUEVENT[21]	0x42	Jazelle 後方分岐
PMUEVENT[22]	0x50	コヒーレントラインフィル ミス[c]
PMUEVENT[23]	0x51	コヒーレントラインフィル ヒット[c]
PMUEVENT[24]	0x60	命令キャッシュ依存ストール
PMUEVENT[25]	0x61	データキャッシュ依存ストール
PMUEVENT[26]	0x62	メイン TLB ミスストール
PMUEVENT[27]	0x63	STREX 成功
PMUEVENT[28]	0x64	STREX 失敗
PMUEVENT[29]	0x65	データ退出
PMUEVENT[30]	0x66	発行により命令がディスパッチされない
PMUEVENT[31]	0x67	発行が空
PMUEVENT[32]	0x70	メイン実行ユニットパイプ
PMUEVENT[33]	0x71	第 2 実行ユニットパイプ
PMUEVENT[34]	0x72	ロード / ストアパイプ
PMUEVENT[35]	0x73	b00 = どの浮動小数点命令も名前変更されません。 b01 = 1 つの浮動小数点命令が名前変更されます。 b10 = 2 つの浮動小数点命令が名前変更されます。
PMUEVENT[36]		
PMUEVENT[37]	0x74	b00 = どの NEON 命令も名前変更されません。 b01 = 1 つの NEON 命令が名前変更されます。 b10 = 2 つの NEON 命令が名前変更されます。
PMUEVENT[38]		
PMUEVENT[39]	0x80	PLD ストール
PMUEVENT[40]	0x81	書き込みストール
PMUEVENT[41]	0x82	命令メイン TLB ミスストール
PMUEVENT[42]	0x83	データメイン TLB ミスストール
PMUEVENT[43]	0x84	命令マイクロ TLB ミスストール
PMUEVENT[44]	0x85	データマイクロ TLB ミスストール
PMUEVENT[45]	0x86	DMB ストール
PMUEVENT[46]	0x8A	整数コアクロック稼働
PMUEVENT[47]	0x8B	データエンジン クロック稼働
PMUEVENT[48]	0x90	ISB

表 A-18 イベント信号とイベント番号 （続き）

名前	イベント番号	説明
PMUEVENT[49]	0x91	DSB
PMUEVENT[50]	0x92	DMB
PMUEVENT[51]	0x93	外部割り込み
PMUEVENT[52]	0xA0	PLE キャッシュライン要求の完了
PMUEVENT[53]	0xA1	PLE キャッシュライン要求のスキップ
PMUEVENT[54]	0xA2	PLE FIFO フラッシュ
PMUEVENT[55]	0xA3	完了した PLE 要求
PMUEVENT[56]	0xA4	PLE FIFO オーバフロー
PMUEVENT[57]	0xA5	プログラムされた PLE 要求

a. Cortex-A9 プロセッサでは生成されません。同様のイベント 0x68 に置き換えられています。
b. Cortex-A9 プロセッサでは生成されません。同様のイベント 0x6E に置き換えられています。
c. マルチプロセッサ構成で使用されます。

「Cortex-A9 固有のイベント」（ページ 11-7）を参照して下さい。

A.9 例外フラグ信号

DEFLAGS 信号を、表 A-19 に示します。

表 A-19 DEFLAGS 信号

名前	I/O	デスティネーション	説明
DEFLAGS[6:0]	O	例外監視ユニット	データエンジン出力フラグ。Cortex-A9 プロセッサにデータエンジン（MPE または FPU）が内蔵されている場合にのみ実装されます。 DE が MPE の場合、ビットは次の意味です。 • ビット [6] が FPSCR[27] の値を示します。 • ビット [5] が FPSCR[7] の値を示します。 • ビット [4:0] が FPSCR[4:0] の値を示します。 DE が FPU の場合、ビットは次の意味です。 • ビット [6] は 0 です。 • ビット [5] が FPSCR[7] の値を示します。 • ビット [4:0] が FPSCR[4:0] の値を示します。

FPSCR の詳細については、*『Cortex-A9 浮動小数点ユニット (FPU) テクニカルリファレンス マニュアル』*と*『Cortex-A9 NEON® メディア処理エンジン テクニカルリファレンス マニュアル』*を参照して下さい。

A.10 パリティ信号

パリティ信号を、表 A-20 に示します。この信号はパリティが定義されている場合のみ存在します。*「パリティエラーのサポート」*（ページ 7-11）を参照して下さい。

表 A-20 パリティ信号

名前	I/O	デスティネーション	説明
PARITYFAIL[7:0]	O	パリティ監視デバイス	RAM アレイからのパリティ出力ピン。 0 = パリティエラーなし 1 = パリティエラーあり ビット [7] BTAC パリティエラー ビット [6] GHB パリティエラー ビット [5] 命令タグ RAM パリティエラー ビット [4] 命令データ RAM パリティエラー ビット [3] メイン TLB パリティエラー ビット [2] D 外部 RAM パリティエラー ビット [1] データタグ RAM パリティエラー ビット [0] データデータ RAM パリティエラー

A.11 MBIST インタフェース

MBIST インタフェース信号を、表 A-21 に示します。これらの信号は、BIST インタフェースが存在する場合にのみ存在します。

表 A-21 MBIST インタフェース信号

名前	I/O	ソース	説明
MBISTADDR[10:0]	I	MBIST コントローラ	MBIST アドレスバス
MBISTARRAY[19:0]	I		RAM のテストに使用される MBIST アレイ
MBISTENABLE	I		MBIST テストイネーブル
MBISTWRITEEN	I		グローバル書き込みイネーブル
MBISTREADEN	I		グローバル読み出しイネーブル

一部の MBIST 信号のサイズは、実装でパリティサポートがサポートされているかどうかによって異なります。パリティサポートが実装されている場合のこれらの信号を、表 A-22 に示します。

表 A-22 パリティサポートが実装されている場合の MBIST 信号

名前	I/O	ソースまたは デスティネーション	説明
MBISTBE[32:0]	I	MBIST コントローラ	MBIST 書き込みイネーブル
MBISTINDATA[71:0]	I		MBIST データ入力
MBISTOUTDATA[71:0]	O		MBIST データ出力

パリティサポートが実装されていない場合のこれらの信号を、表 A-23 に示します。

表 A-23 パリティサポートが実装されていない場合の MBIST 信号

名前	I/O	ソースまたは デスティネーション	説明
MBISTBE[25:0]	I	MBIST コントローラ	MBIST 書き込みイネーブル
MBISTINDATA[63:0]	I		MBIST データ入力
MBISTOUTDATA[63:0]	O		MBIST データ出力

MBIST については、*『Cortex-A9 r0p0 MBIST テクニカルリファレンス マニュアル』*を参照して下さい。

A.12 スキャンテスト信号

スキャンテスト信号を、表 A-24 に示します。

表 A-24 スキャンテスト信号

名前	I/O	デスティネーション	説明
SE	I	DFT コントローラ	スキャンイネーブル。 0 = 不可能 1 = 可能

A.13 外部デバッグインタフェース

次に示すセクションでは、外部デバッグインタフェース信号について説明します。

- *「認証インタフェース」*
- *「APB インタフェース信号」*（ページ A-22）
- *「CTI 信号」*（ページ A-22）
- *「その他のデバッグインタフェース信号」*（ページ A-23）

A.13.1 認証インタフェース

認証インタフェース信号を、表 A-25 に示します。

表 A-25 認証インタフェース信号

名前	I/O	ソース	説明
DBGEN	I	セキュリティコントローラ	侵襲性デバッグイネーブル。 0 = 不可能 1 = 可能
NIDEN	I		非侵襲性デバッグイネーブル。 0 = 不可能 1 = 可能
SPIDEN	I		セキュア特権侵襲性デバッグイネーブル。 0 = 不可能 1 = 可能
SPNIDEN	I		セキュア特権非侵襲性デバッグイネーブル。 0 = 不可能 1 = 可能

A.13.2 APB インタフェース信号

APB インタフェース信号を、表 A-26 に示します。

表 A-26 APB インタフェース信号

名前	I/O	ソースまたは デスティネーション	説明
PENABLEDBG	I	CoreSight APB デバイス	APB クロックイネーブル
PRDATADBG[31:0]	O		APB 読み出しデータバス
PSELDBG	I		デバッグレジスタの選択。 0 = デバッグレジスタが選択されません。 1 = デバッグレジスタが選択されます。
PSLVERRDBG	O		APB スレーブエラー信号
PWRITEDBG	I		APB 読み出し / 書き込み信号
PADDRDBG[12:2]	I		プログラミングアドレス
PADDRDBG31	I		APB アドレスバスのビット [31]。 0 = 外部デバッガのアクセスではありません。 1 = 外部デバッガのアクセスです。
PREADYDBG	O		APB スレーブ準備完了。APB スレーブは **PREADY** をアサートして転送を延長できます。
PWDATADBG[31:0]	I		APB 書き込みデータ

A.13.3 CTI 信号

CTI 信号を、表 A-27 に示します。

表 A-27 CTI 信号

名前	I/O	ソースまたは デスティネーション	説明
EDBGRQ	I	外部デバッガまたは CoreSight 相互接続	外部デバッグ要求。 0 = 外部デバッグ要求なし 1 = 外部デバッグ要求あり プロセッサは **EDBGRQ** 入力をレベル感知として扱います。**EDBGRQ** 入力は、プロセッサで **DBGACK** がアサートされるまでアサートしておく必要があります。
DBGACK	O		デバッグ応答信号
DBGCPUDONE	O		Cortex-A9 プロセッサから発行されたすべてのメモリアクセスが、デバッガで実行された操作によるものであることを示します。アクティブ HIGH。
DBGRESTART	I		コアがデバッグ状態を終了します。**DBGRESTARTED** がアサート解除されるまで HIGH に保つ必要があります。 0 = 不可能 1 = 可能
DBGRESTARTED	O		デバッグ状態と通常状態との間を移行するため、**DBGRESTART** とともに使用されます。 0 = 不可能 1 = 可能

A.13.4 その他のデバッグインタフェース信号

その他のデバッグインタフェース信号を、表 A-28 に示します。

表 A-28 その他のデバッグ信号

名前	I/O	ソースまたは デスティネーション	説明
COMMRX	O	デバッグ通信チャネル	通信チャネル受信 データ転送レジスタ フルフラグの受信部分。 0 = 空 1 = フル
COMMTX	O	デバッグ通信チャネル	通信チャネル送信 データ転送レジスタ フルフラグの送信部分。 0 = 空 1 = フル
DBGNOPWRDWN	O	デバッガ	デバッガが、Cortex-A9 プロセッサを電力オフにしないよう要求しました。
DBGSWENABLE	I	外部デバッガ	LOW の場合は、外部デバッグエージェントでのみデバッグレジスタを変更できます。 0 = 不可能 1 = 可能
DBGROMADDR[31:12]	I	システム構成	ROM テーブルの物理アドレスのビット [31:12] を示します。 アドレスが決定できない場合は、この信号を 0 に固定します。
DBGROMADDRV	I		**DBGROMADDR** の有効信号。 アドレスが決定できない場合は、この信号を LOW に固定します。
DBGSELFADDR[31:15]	I		ROM テーブルの物理アドレスから、デバッグレジスタがメモリマップされる物理アドレスまでの、符号付き 2 の補数形式のオフセットの、ビット [31:15] を示します。 オフセットが決定できない場合は、この信号を 0 に固定します。
DBGSELFADDRV	I		**DBGSELFADDR** の有効信号。 オフセットが決定できない場合は、この信号を LOW に固定します。

第 10 章 *デバッグ*を参照して下さい。

A.14 PTM インタフェース信号

PTM インタフェース信号を、表 A-29 に示します。これらの信号は、PTM インタフェースが存在する場合にのみ存在します。

I/O 列の I は、PTM インタフェースから Cortex-A9 プロセッサへの入力を示します。O は、Cortex-A9 プロセッサから PTM への出力を示します。これらの信号はすべて、Cortex-A9 クロックドメインに存在します。

表 A-29 PTM インタフェース信号

名前	I/O	ソースまたは デスティネーション	説明
WPTCOMMIT[1:0]	O	PTM デバイス	このサイクルをコミットしたウェイポイントの数。有効なウェイポイントを示し、同じサイクル内でコミットすることができます。
WPTCONTEXTID[31:0]	O		ウェイポイントのコンテキスト ID。 この信号は、ウェイポイントの条件コードに関係なく、TRUE にする必要があります。 コアコンテキスト ID が設定されていない場合は、**WPTCONTEXTID[31:0]** で 0 を報告する必要があります。
WPTENABLE	I		ウェイポイントイネーブル
WPTEXCEPTIONTYPE[3:0]	O		例外タイプ。 b0001 = ホールト デバッグモード b0010 = セキュアモニタ b0100 = 不正確データアボート b0101 = T2EE トラップ b1000 = リセット b1001 = 未定義 b1010 = SVC b1011 = プリフェッチアボート / ソフトウェアブレークポイント b1100 = 正確データアボート / ソフトウェアウォッチポイント b1110 = IRQ b1111 = FIQ
WPTFLUSH	O		ウェイポイントフラッシュ信号
WPTLINK	O		ウェイポイントは分岐で、リンクレジスタを更新します。**WPTTYPE** が直接分岐または間接分岐の場合のみ HIGH になります。

表 A-29 PTM インタフェース信号 （続き）

名前	I/O	ソースまたは デスティネーション	説明
WPTPC[31:0]	O	PTM デバイス	ウェイポイントの最終実行アドレスインジケータ。 これは、例外が発生した場合のベースリンク レジスタです。 ウェイポイントがリセット例外の場合は 0 と一致します。
WPTT32LINK	O		Thumb 状態で最後に実行されたアドレスのサイズを示します。 0 = 16 ビット命令 1 = 32 ビット命令
WPTTAKEN	O		ウェイポイントの条件コードが成功しました。アドレスは、この信号の値にかかわらず使用されます。 すべてのウェイポイント例外分岐についてセットする必要があります。
WPTTARGETJBIT	O		ウェイポイントデスティネーションの **J** ビット
WPTTARGETPC[31:0]	O		ウェイポイント ターゲットアドレス。 **T** ビットが 0 の場合、ビット [1] を 0 にする必要があります。 **J** ビットが 0 の場合、ビット [0] を 0 にする必要があります。 **WPTTYPE** が禁止されているか、デバッグの場合、この値は 0 です。
WPTTARGETTBIT	O		ウェイポイントデスティネーションの **T** ビット
WPTTRACEPROHIBITED	O	PTM デバイス	現在のウェイポイントターゲットに対するトレースが禁止されます。 禁止された領域へのエントリを示します。トレースが再開されるまで、ウェイポイントはトレースされません。 **NIDEN** と **DBGEN** の両方が LOW の場合、フライト中のウェイポイントがコアを出た後に、この信号を恒久的にアサートする必要があります。入力の変化がサンプリングされたことを保証するには、例外またはシリアル分岐が必要です。 **WPTTRACEPROHIBITED** がセットされた状態で、**WPTVALID** が観測されるのは 1 サイクルだけにする必要があります。 このウェイポイントでトレースは停止し、観測される次のウェイポイントは Isync パケットです。 トレースで使用されるパケットについては、*『CoreSight PTM アーキテクチャ仕様』*を参照して下さい。
WPTTYPE[2:0]	O		ウェイポイントタイプ b000 = 直接分岐 b001 = 間接分岐 b010 = 例外 b011 = DMB/DSB/ISB b100 = デバッグ開始 b101 = デバッグ終了 b110 = 無効 b111 = 無効 デバッグ開始の後には、デバッグ終了が続く必要があります。 **注** デバッグ終了は命令の実行を反映しません。

表 A-29 PTM インタフェース信号 （続き）

名前	I/O	ソースまたは デスティネーション	説明
WPTVALID	O	PTM デバイス	ウェイポイントが有効であることが確認されます。
WPTnSECURE	O		現在のウェイポイントに続く命令が、非セキュア状態で実行されます。NS ビットがセットされ、プロセッサがセキュア モニタ モードでない場合、命令は非セキュア状態です。 セキュリティ拡張機能については、*「システム制御について」*（ページ 4-2）を参照して下さい。
WPTFIFOEMPTY	O		PTM インタフェースの FIFO に、投機的ウェイポイントが存在しません。

「インタフェース」（ページ 2-4）を参照して下さい。

付録 B
命令サイクルタイミング

本章では、Cortex-A9 プロセッサの整数命令のサイクルタイミングについて説明します。本章は次のセクションから構成されています。

B.1 命令のサイクルタイミングについて

本章では、特定のコードシーケンスに必要な実行時間を見積もるための情報を提供します。Cortex-A9 プロセッサの複雑さのため、手動で正確なタイミング情報を計算することは不可能です。命令タイミングは、多くの場合、同時に実行されている命令、メモリシステムの動作、および命令フロー以外のイベントの影響を受けます。可能性のある命令の相互作用と、プロセッサ内で発生する可能性のあるイベントの詳細な説明は、本書の範囲を越えています。

B.2 データ処理命令

データ処理命令の実行ユニットサイクル時間を、表 B-1 に示します。

表 B-1 に示されているのは、次のような場合です。

ソースレジスタ上のシフトなし

例：`ADD r0, r1, r2`

イミディエートソース レジスタによるシフト

例：`ADD r0, r1, r2 LSL #2`

レジスタによるシフト

例：`ADD r0, r1, r2 LSL r3`

表 B-1 データ処理命令のサイクルタイミング

命令	シフトなし	シフトの種類	
		定数	レジスタ
MOV	1	1	2
AND, EOR, SUB, RSB, ADD, ADC, SBC, RSC, CMN, ORR, BIC, MVN, TST, TEQ, CMP	1	2	3
QADD, QSUB, QADD8, QADD16, QSUB8, QSUB16, SHADD8, SHADD16, SHSUB8, SHSUB16, UQADD8, UQADD16, UQSUB8, UQSUB16, UHADD8, UHADD16, UHSUB8, UHSUB16, QASX, QSAX, SHASX, SHSAX, UQASX, UQSAX, UHASX, UHSAX	2	-	-
QDADD, QDSUB, SSAT, USAT	3	-	-
PKHBT, PKHTB	1	2	-
SSAT16, USAT16, SADD8, SADD16, SSUB8, SSUB16,UADD8, UADD16, USUB8, USUB16, SASX, SSAX, UASX, USAX	1	-	-
SXTAB, SXTAB16, SXTAH, UXTAB, UXTAB16, UXTAH	3	-	-
SXTB, STXB16, SXTH, UXTB, UTXB16, UXTH	2	-	-
BFC, BFI, UBFX, SBFX	2	-	-
CLZ, MOVT, MOVW, RBIT, REV, REV16, REVSH, MRS	1	-	-
モードビットまたは制御ビットを変更しない MSR *「直列化命令」*（ページ B-9）を参照して下さい。	1	-	-

B.3 ロード / ストア命令

ロード / ストア命令は次のように分類されます。

- LDR 命令などの単一ロード / ストア命令
- LDM 命令などの複数ロード / ストア命令

複数ロードおよび複数ストア命令では、通常は、レジスタリスト内のレジスタの数によって、ロードまたはストア命令の実行に必要なサイクル数が決定されます。

Cortex-A9 プロセッサには、ロード命令から後続のデータ処理命令にデータを直接転送する特別なパスが、実行ユニットに存在します。

このパスは、次の条件が満たされたときに使用されます。

- データ処理命令が、SUB、RSB、ADD、ADC、SBC、RSC、CMN、MVN、CMP のいずれかである。
- 転送されるソースレジスタが、シフト操作に含まれていない。

単一ロード / ストア操作のサイクルタイミングを、表 B-2 に示します。

結果レイテンシは、最初にロードされるレジスタのレイテンシです。

表 B-2 単一ロード / ストア操作のサイクルタイミング

命令サイクル	AGU サイクル	結果レイテンシ	
		高速転送の場合	その他の場合
LDR ,[reg] LDR ,[reg imm] LDR ,[reg reg] LDR ,[reg reg LSL #2]	1	2	3
LDR ,[reg reg LSL reg] LDR ,[reg reg LSR reg] LDR ,[reg reg ASR reg] LDR ,[reg reg ROR reg] LDR ,[reg reg, RRX]	1	3	4
LDRB ,[reg] LDRB ,[reg imm] LDRB ,[reg reg] LDRB ,[reg reg LSL #2] LDRH ,[reg] LDRH ,[reg imm] LDRH ,[reg reg] LDRH ,[reg reg LSL #2]	2	3	4
LDRB ,[reg reg LSL reg] LDRB ,[reg reg ASR reg] LDRB ,[reg reg LSL reg] LDRB ,[reg reg ASR reg] LDRH ,[reg reg LSL reg] LDRH ,[reg reg ASR reg] LDRH ,[reg reg LSL reg] LDRH ,[reg reg ASR reg]	2	4	5

Cortex-A9 プロセッサは、サイクルごとに 2 つの 32 ビットレジスタをロードまたはストアできます。ただし、64 ビットにアクセスするには、アドレスを 64 ビットアラインドにする必要があります。

このスケジューリングは、*アドレス生成ユニット* (AGU) で実行されます。AGU で複数ロードまたは複数ストア操作を処理するために必要なサイクル数は、レジスタリストの長さと、アドレスが 64 ビットアラインドかどうかによって異なります。結果レイテンシは、最初にロードされるレジスタのレイテンシです。複数ロード操作のサイクルタイミングを、表 B-3 に示します。

表 B-3 複数ロード操作のサイクルタイミング

命令	命令を処理するための AGU サイクル数		結果レイテンシ	
	64 ビット境界にアラインされたアドレス		高速転送の場合	その他の場合
	はい	いいえ		
LDM ,{ 対象レジスタ 1 個 }	1	1	2	3
LDM ,{ 対象レジスタ 2 個 } LDRD RFE	1	2	2	3
LDM ,{ 対象レジスタ 3 個 }	2	2	2	3
LDM ,{ 対象レジスタ 4 個 }	2	3	2	3
LDM ,{ 対象レジスタ 5 個 }	3	3	2	3
LDM ,{ 対象レジスタ 6 個 }	3	4	2	3
LDM ,{ 対象レジスタ 7 個 }	4	4	2	3
LDM ,{ 対象レジスタ 8 個 }	4	5	2	3
LDM ,{ 対象レジスタ 9 個 }	5	5	2	3
LDM ,{ 対象レジスタ 10 個 }	5	6	2	3
LDM ,{ 対象レジスタ 11 個 }	6	6	2	3
LDM ,{ 対象レジスタ 12 個 }	6	7	2	3
LDM ,{ 対象レジスタ 13 個 }	7	7	2	3
LDM ,{ 対象レジスタ 14 個 }	7	8	2	3
LDM ,{ 対象レジスタ 15 個 }	8	8	2	3
LDM ,{ 対象レジスタ 16 個 }	8	9	2	3

複数ストア操作のサイクルタイミングを、表 B-4 に示します。

表 B-4 複数ストア操作のサイクルタイミング

命令	AGU サイクル 64 ビット境界にアラインしている	
	はい	いいえ
STM ,{ 対象レジスタ 1 個 }	1	1
STM ,{ 対象レジスタ 2 個 } STRD SRS	1	2
STM ,{ 対象レジスタ 3 個 }	2	2
STM ,{ 対象レジスタ 4 個 }	2	3
STM ,{ 対象レジスタ 5 個 }	3	3
STM ,{ 対象レジスタ 6 個 }	3	4
STM ,{ 対象レジスタ 7 個 }	4	4
STM ,{ 対象レジスタ 8 個 }	4	5
STM ,{ 対象レジスタ 9 個 }	5	5
STM ,{ 対象レジスタ 10 個 }	5	6
STM ,{ 対象レジスタ 11 個 }	6	6
STM ,{ 対象レジスタ 12 個 }	6	7
STM ,{ 対象レジスタ 13 個 }	7	7
STM ,{ 対象レジスタ 14 個 }	7	8
STM ,{ 対象レジスタ 15 個 }	8	8
STM ,{ 対象レジスタ 16 個 }	8	9

B.4 乗算命令

乗算命令のサイクルタイミングを、表 B-4 （ページ B-6）に示します。

表 B-5 乗算命令のサイクルタイミング

命令	サイクル数	結果レイテンシ
MUL(S), MLA(S)	2	4
SMULL(S), UMULL(S), SMLAL(S), UMLAL(S)	3	4（最初に書き込まれるレジスタ） 5（2 番目に書き込まれるレジスタ）
SMULxy, SMLAxy, SMULWy, SMLAWy	1	3
SMLALxy	2	3（最初に書き込まれるレジスタ） 4（2 番目に書き込まれるレジスタ）
SMUAD, SMUADX, SMLAD, SMLADX, SMUSD, SMUSDX, SMLSD, SMLSDX	1	3
SMMUL, SMMULR, SMMLA, SMMLAR, SMMLS, SMMLSR	2	4
SMLALD, SMLALDX, SMLSLD, SMLDLDX	2	3（最初に書き込まれるレジスタ） 4（2 番目に書き込まれるレジスタ）
UMAAL	3	4（最初に書き込まれるレジスタ） 5（2 番目に書き込まれるレジスタ）

B.5 分岐命令

分岐命令にはさまざまなタイミング特性があります。

- イミディエート位置への分岐命令は、実行ユニットサイクルを消費しません。
- PC レジスタに対するデータ処理命令は、標準の命令として実行ユニットで処理されます。*「データ処理命令」*（ページ B-3）を参照して下さい。
- PC レジスタへのロード命令は、標準の命令として実行ユニットで処理されます。*「ロード / ストア命令」*（ページ B-4）を参照して下さい。

また、動的分岐予測の詳細については、*「レベル 1 命令側メモリシステムについて」*（ページ 7-5）を参照して下さい。

B.6 直列化命令

アウトオブオーダ実行はいつでも使用できるわけではありません。命令によっては直列化する場合があります。直列化命令は、プロセッサに対して、次の命令が実行される前に、フラグと汎用レジスタに対するすべての変更を完了するように強制します。

ここでは、直列化命令のタイミングを一覧表で示します。

B.6.1 直列化命令

次の例外開始命令は直列化命令です。

- SVC
- SMC
- BKPT
- プリフェッチアボート ハンドラを呼び出す命令
- 未定義命令例外ハンドラを呼び出す命令

モードまたはプログラム制御を変更する次の命令は、直列化命令です。

- MSR CPSR（制御ビットまたはモードビットを変更する場合）
- S ビットがセットされた、PC に対するデータ処理（MOVS pc, r14 など）
- LDM pc ^.
- CPS
- SETEND
- RFE

次の命令は直列化命令です。

- ISB と DMB を除く、cp14 または cp15 に対するすべての MCR
- デバッグレジスタの MRC p14
- WFE、WFI、SEV
- CLREX
- DSB

r1p0 実装では、DMB は、すべての命令の終了ではなく、以前のすべての LDR/STR 命令の終了を待ちます。

SPSR を変更する次の命令は、直列化命令です。

- MSR SPSR

付録 C
リビジョン

この付録では、本書の各版の技術的な相違点について説明します。

表 C-1 A 版

変更内容	場所
初版	-

表 C-2 A 版と B 版の相違点

変更内容	場所
ロード / ストアユニットとアドレス生成の明確化	図 1-1 （ページ 1-2）
高速ループモードを小ループモードに変更	• 図 1-1 （ページ 1-2） • 小ループモード （ページ 1-3） • *「命令キャッシュの特徴」*（ページ 7-2） • 電力消費制御について（ページ 12-6）
分岐予測を動的分岐予測に変更	• *「機能」*（ページ 1-6） • *「レベル 1 命令側メモリシステムについて」*（ページ 7-5） • *「分岐命令」*（ページ B-8）
L1 キャッシュコヒーレンシを L1 データキャッシュコヒーレンシに変更	*「Cortex-A9 のバリアント」*（ページ 1-4）
プロセッサ機能レジスタ 0 のリセット時の値を修正	表 4-29 （ページ 4-46）
PMSWINC の説明を整合するように変更	• 表 4-29 （ページ 4-46） • ソフトウェアインクリメント レジスタ（ページ 4-100）

表 C-2 A 版と B 版の相違点 （続き）

変更内容	場所
MIDR ビット [3:0] を 0 から 1 に更新	表 4-1 （ページ 4-5）
ID_MMFR3 [23:20] ビットの値を 0x1 に訂正	表 4-42 （ページ 4-50）
AFE ビットの説明の訂正	表 4-51 （ページ 4-62）
補助制御レジスタのビットフィールドの訂正	• 表 4-52 （ページ 4-66） • 図 4-36 （ページ 4-66）
S パラメータの値の訂正	セット / ウェイ形式（ページ 4-83）
ビット [11]、[10]、[8] の説明を、表と整合するように修正	図 4-41 （ページ 4-87）
アーキテクチャ的に削除されたイベント 0x68 の説明の訂正	表 4-80 （ページ 4-123）
TLB ロックダウンエントリ番号を 8 から 4 に訂正	c10、TLB ロックダウンレジスタ（ページ 4-134）
A、I、F ビットの説明の訂正	c12、割り込みステータスレジスタ（ページ 4-147）
マイクロ TLB エントリ数を 8 から 32 に変更	*「マイクロ TLB」*（ページ 6-4）
キャッシュタイプに関する重複情報の削除	*「マイクロ TLB」*（ページ 6-4）
IRGN ビットの説明を、TTBCR から TTBR0/TTRBR1 に修正	*「メイン TLB」*（ページ 6-4）
使用前のキャッシュと BTAC の無効化に関する注を追加	*「レベル 1 メモリシステムについて」*（ページ 7-2）
パリティサポート方式の情報のセクションを追加	*「パリティエラーのサポート」*（ページ 7-11）
L2 マスタインタフェース M0 および M1 の一覧表と説明を追加	*「Cortex-A9 のレベル 2 インタフェースについて」*（ページ 8-2）
DBSCR の外部説明への相互参照を追加。DBSCR 外部ビューへの参照を含めるように脚注を拡張。	表 10-1 （ページ 10-5）
DBGDSCR の説明を訂正し、内部ビューと外部ビューの説明を追加	CP14 c1、デバッグステータスおよび制御レジスタ（DBGDSCR）（ページ 8-9）
MOE ビットの説明の順序変更と拡張	表 8-2 （ページ 8-10）
表 10-1 からの相互参照の追加	• デバッグ状態キャッシュ制御レジスタ（DBGDSCCR）（ページ 8-8） • CP14 c1、デバッグステータスおよび制御レジスタ（DBGDSCR）（ページ 8-9） • デバイス電力オフおよびリセットステータス レジスタ（DBGPRSR）（ページ 8-27） • 統合モード制御レジスタ（DBGITCTRL）（ページ 8-45） • クレームタグ クリアレジスタ（DBGCLAIMCLR）（ページ 8-47） • ロックアクセス レジスタ（DBGLAR）（ページ 8-48） • ロックステータス レジスタ（DBGLSR）（ページ 8-49） • 認証ステータスレジスタ（DBGAUTHSTATUS）（ページ 8-49） • デバイスタイプ レジスタ（DBGDEVTYPE）（ページ 8-50）

表 C-2 A 版と B 版の相違点 （続き）

変更内容	場所
表 10-1 の脚注の訂正	表 10-1 （ページ 10-5）
バイトアドレス フィールドのエントリの訂正	表 10-8 （ページ 10-11）
割り込み信号の説明の訂正	表 A-3 （ページ A-4）
AXI USER の説明の拡張	• 表 A-8 （ページ A-8） • 表 A-11 （ページ A-10） • 表 A-14 （ページ A-12）

表 C-3 B 版と C 版の相違点

変更内容	場所
「2.8.1 64 ビット幅のバス上の LE アクセスと BE-8 アクセス」の削除	-
「第 4 章 アンアライインド データアクセスとエンディアン混在データアクセスのサポート」の削除	-
電力管理信号 **BISTSCLAMP** の削除	-
動的高水準クロックゲートの追加	動的高水準クロックゲート（ページ 2-9）
TLB 情報の更新	表 1-1 （ページ 1-10）、表 4-10 （ページ 4-15）、表 4-37 （ページ 4-44）
ID_MMF3[15:12] の説明の短縮	メモリモデル機能レジスタ 3 （ページ 4-49）
PL310 最適化への参照を含めるように ACTLR を更新	補助制御レジスタ（ページ 4-64）
2 つ目の置換方式に関する情報の追加。選択は SCTLR.RR ビットで行われます。	*「システム制御レジスタ」*（ページ 4-15）
イベント情報の拡張	Cortex-A9 固有のイベント （ページ 4-32）
DEFLAGS[6:0] の追加	DEFLAGS[6:0] （ページ 4-37、*「パフォーマンス監視信号」*（ページ A-14））
電力制御レジスタの説明の追加	電力制御レジスタ（ページ 4-63）
L2 メモリインタフェースに対する PL310 の最適化の説明の追加	*「レベル 2 メモリインタフェースへのアクセスの最適化」*（ページ 8-7）
ウォッチポイントアドレスのマスク処理の追加	*「ウォッチポイント制御レジスタ」*（ページ 10-11）
デバッグ要求再起動の図の追加	*「リセットのデバッグレジスタへの影響」*（ページ 10-3）
CPUCLKOFF 情報の追加	表 A-4 （ページ A-5）、非レジスト信号（ページ B-3）
DECLKOFF 情報の追加	表 A-4 （ページ A-5）、非レジスト信号（ページ B-3）
MAXCLKLATENCY[2:0] 情報の追加	*「構成信号」*（ページ A-5）

表 C-3 B 版と C 版の相違点 （続き）

変更内容	場所
PMUEVENT バスの説明の拡張	*「パフォーマンス監視信号」*（ページ A-14）
PMUSECURE と **PMUPRIV** の追加	*「パフォーマンス監視信号」*（ページ A-14）
DMB の直列化動作の説明の更新	*「直列化命令」*（ページ B-9）

表 C-4 C 版と D 版の相違点

変更内容	場所
ブロック図に*プリロードエンジン*（PE) を追加	図 1-1 （ページ 1-2）
割り込み信号の修正	
データエンジン オプションの明確化	*「データエンジン」*（ページ 1-2）
システム設計コンポーネントの明確化	*「システム設計コンポーネント」*（ページ 1-3）
*準拠性*の明確化	*「準拠性」*（ページ 1-5）
PE を機能に追加	*「機能」*（ページ 1-6）
構成オプションに PE と PE FIFO サイズを追加	*「構成可能なオプション」*（ページ 1-8）
NEON SIMD および FPU オプションの明確化	表 1-1 （ページ 1-8）
*テスト機能*セクションの追加	*「テスト機能」*（ページ 1-9）
「2.1.3 PTM インタフェース」の書き直し	*「パフォーマンス監視」*（ページ 2-3）
「2.1.5 割り込みの仮想化」の追加	*「割り込みの仮想化」*（ページ 2-3）
電力制御の説明に NEON SIMD クロックゲートを追加	*「電力制御レジスタ」*（ページ 2-9）
nDERESET を **nNEONRESET** に変更	*「リセットモード」*（ページ 2-7）
nWDRESET の追加	
nPERIPHRESET の追加	
電圧ドメイン境界と説明の変更	図 2-4 （ページ 2-14）
「2.1.5 データエンジン ロジックリセット」の置き換え	*「MPE SIMD ロジックリセット」*（ページ 2-8）
Cortex-A9 入力信号 DECLAMP の削除、レベルシフタ基準の削除	*「電力管理コントローラとの通信」*（ページ 2-13）
「表 3-1 J ビットと T ビットのエンコード」の削除	-
Jazelle 拡張機能（ページ 3-3）の移動	*「Jazelle 拡張機能」*（ページ 3-7）
NEON テクノロジ（ページ 3-4）の名前変更と更新	*「アドバンスト SIMD アーキテクチャ」*（ページ 3-4）
「3.4 プロセッサの動作状態」の削除	-

表 C-4 C 版と D 版の相違点 （続き）

変更内容	場所
「3.5 データ型」の削除	-
「マルチプロセッシング拡張機能」セクションの追加	*「マルチプロセッシング拡張機能」*（ページ 3-6）
「3.6 メモリフォーマット」の名前変更と移動	*「メモリモデル」*（ページ 3-8）
「3.8 セキュリティ拡張機能の概要」の名前変更と移動	*「セキュリティ拡張機能アーキテクチャ」*（ページ 3-5）
『ARM アーキテクチャ リファレンスマニュアル』 資料と重複している内容、表、図を 4.1 から削除	*「システム制御について」*（ページ 4-2）
『ARM アーキテクチャ リファレンスマニュアル』 資料と重複している 4.2 の内容を削除、セクションの名前変更	*「レジスタの概要」*（ページ 4-3）
『ARM アーキテクチャ リファレンスマニュアル』 資料と重複している 4.3 の内容を削除、セクションの名前変更	*「レジスタの説明」*（ページ 4-8）
脚注 e の削除	表 4-8 （ページ 4-15）
プリロードエンジン レジスタの追加	*「CP15 c11 レジスタの概要」*（ページ 4-30）
-	*「PLE ID レジスタ」*（ページ 4-30）
-	*「PLE 動作ステータスレジスタ」*（ページ 4-31）
-	*「PLE FIFO ステータスレジスタ」*（ページ 4-32）
-	*「プリロードエンジン ユーザアクセス許可レジスタ」*（ページ 4-32）
-	*「プリロードエンジン パラメータ制御レジスタ」*（ページ 4-33）
新しい章に「4.4 CP14 Jazelle レジスタ」と「4.5 CP14 Jazelle レジスタの説明」を追加	第 5 章 *Jazelle DBX レジスタ*
「第 5 章 メモリ管理ユニット」で、「5.6 MMU のソフトウェアアクセス可能レジスタ」セクションを削除	-
「レベル 1 メモリシステム」章で、Cortex-A9 キャッシュ方式のセクションを削除	-
L2 メモリインタフェースに関するプリフェッチヒントの説明を更新して拡張	*「レベル 2 メモリインタフェースに対するプリフェッチヒント」*（ページ 8-7）
BRESP とキャッシュコントローラの動作の明確化	*「早期 BRESP」*（ページ 8-7）
0 のフルライン書き込み、信号名を **AWUSERM0[7]** に訂正	*「0 のフルライン書き込み」*（ページ 8-8）
投機的コヒーレント要求のセクションを追加	*「投機的コヒーレント要求」*（ページ 8-8）
PARITYFAIL の未使用ビットを HIGH に固定することに関する文の削除	*「パリティエラーのサポート」*（ページ 7-11）

表 C-4 C 版と D 版の相違点 （続き）

変更内容	場所
PE の説明の追加	第 9 章 *プリロードエンジン*
PMU の説明の追加	第 11 章 *パフォーマンス監視ユニット*
デバッグの章、「デバッグシステムについて」の削除	-
デバッグの章、「デバッグモード」の削除	-
『ARM アーキテクチャ リファレンスマニュアル』 資料と重複する部分の削除	-
外部デバッグインタフェース、**PADDRDBG[12:0]** の説明の追加	*「外部デバッグインタフェース」*（ページ 10-16）
デバッグ APB インタフェースのセクションの追加	*「デバッグ APB インタフェース」*（ページ 10-18）
信号の説明の修正と拡張、ソース / デスティネーション列の追加	付録 A *信号の説明*
PMUEVENT[46] の説明の訂正	表 A-17 （ページ A-14）
PMUEVENT[47] の説明の訂正	
*AC 特性*付録の削除	-

D 版と E 版の相違点はありません。

表 C-5 D 版と F 版の相違点

変更内容	場所
PL310 の名前を L2C-310 に変更	本書全体
VFPv3 を VFPv3 D-32 に訂正	*「メディア処理エンジン」*（ページ 1-2）
Cortex-A9 FPU ハードウェアの説明を明確化のため書き直し	*「浮動小数点ユニット」*（ページ 1-2）
SCU の説明の拡張	*「Cortex-A9 のバリアント」*（ページ 1-4）
動的分岐予測の説明の追加	*「動的分岐予測」*（ページ 2-3）
最後の段落を削除	*「エネルギー効率機能」*（ページ 2-10）
WFI/WFE を「スタンバイ」に訂正	表 2-2 （ページ 2-10）
明確化のため名前の変更と書き直し	*「スタンバイモード」*（ページ 2-11）
休眠モードクランプ情報の削除	*「休眠モード」*（ページ 2-12）
IEM サポートの名前変更と書き直し	*「電力ドメイン」*（ページ 2-13）
重複資料の削除	*「プログラマモデルについて」*（ページ 3-2）
デバッグレジスタの説明の訂正	表 4-1 （ページ 4-3）
r2p1 と r2p2 のメイン ID レジスタ値の追加	表 4-2 （ページ 4-8）
デバッグレジスタ名の訂正	表 4-2 （ページ 4-8）

表 C-5 D 版と F 版の相違点 （続き）

変更内容	場所
説明の明確化と脚注の追加	表 4-4 （ページ 4-11）
目的の説明の拡張	*「キャッシュサイズ識別レジスタ」* （ページ 4-11）
システム制御レジスタの値の訂正。脚注の修正。	表 4-8 （ページ 4-15）
ビット [17] の機能の訂正	表 4-9 （ページ 4-16）
脚注 d の訂正	表 4-33 （ページ 4-36）
目的の説明の拡張	*「電力制御レジスタ」*（ページ 4-36）
構成の説明の訂正	*「構成ベースアドレス レジスタ」*（ページ 4-38）
章の名前変更	第 5 章 *Jazelle DBX レジスタ*
「6.1 アプリケーション固有」を「アドレス空間固有」に訂正	*「MMU について」*（ページ 6-2）
統一メイン TLB の説明の明確化	*「メモリ管理ユニット」*（ページ 6-2）
ページサイズに関する重複情報の削除	
ASID の説明の訂正と拡張。相互参照の追加。	
TLB 一致プロセスのページサイズに関する重複情報の削除	*「TLB 一致プロセス」*（ページ 6-4）
同期および非同期アボートの不正な相互参照の削除	*「同期アボートと非同期アボート」* （ページ 6-8）
キャッシュ機能の相互参照の訂正	*「キャッシュ機能」*（ページ 7-2）
実装情報の削除	
リターンスタック予測の ARM または Thumb 状態を命令状態に変更	*「リターンスタック予測」*（ページ 7-7）
DSB のセクションの追加	*「DSB について」*（ページ 7-9）
AXI Master0 インタフェース属性の値の訂正	表 8-1 （ページ 8-2）
デバッグの章を PMU の章の前に移動	
図の描き直し	図 10-1 （ページ 10-4）
ビット形式の訂正	表 10-1 （ページ 10-5）
CLUSTERID 値に関する脚注の追加	表 10-10 （ページ 10-13）
値列の追加	表 10-11 （ページ 10-14）
DBGCPUDONE の説明の拡張	*「DBGCPUDONE」*（ページ 10-19）
PMU 管理レジスタのセクションの追加	*「PMU 管理レジスタ」*（ページ 11-3）
信号の説明の拡張	*「構成信号」*（ページ A-5）

表 C-5 D 版と F 版の相違点 （続き）

変更内容	場所
信号の説明の拡張、AXI からの重複情報の削除	表 A-8 （ページ A-8）
AWBURSTM0[1:0]	
AWLENM0[3:0]	
AWLOCKM0[1:0]	
信号の説明の拡張、AXI からの重複情報の削除	表 A-11 （ページ A-10）
ARLENM0[3:0]	
ARLOCKM0[1:0]	
タイトルの変更	*「AXI Master1 信号の命令アクセス」* （ページ A-11）
AXI からの重複情報の削除	表 A-14 （ページ A-12）
ARLENM1[3:0]	
PMUEVENT[46] と **PMUEVENT[47]** の訂正	表 A-17 （ページ A-14）
「はじめに」の短縮。DSB 動作に関する注の追加。	*「直列化命令」*（ページ B-9）

用語集

この用語集では、ARM のマニュアルで使用されている用語の一部について説明します。複数の意味を持つ用語については、本書では用語集に示す意味で使用されています。

AHB アドバンスト ハイパフォーマンスバス*参照。*

AHB-AP AHB アクセスポート*参照。*

AHB-Lite 完全な AMBA AHB プロトコル仕様のサブセット。大部分の AMBA AHB マスタ / スレーブ設計に必要なすべての基本機能を提供しており、特に複数レイヤの AMBA 相互接続で使用されます。ほとんどの場合、完全な AMBA AHB インタフェースで提供されている追加の機能は、AMBA AXI プロトコルインタフェースで実装するとより効率的になります。

AHB アクセスポート (AHB-AP)
DAP のオプションコンポーネントで、SoC への AHB インタフェースを提供します。

AMBA アドバンスト マイクロコントローラバス アーキテクチャ*参照。*

APB アドバンスト ペリフェラルバス*参照。*

ARM 状態 ARM（32 ビット）ワードアラインド命令を実行しているプロセッサは、ARM 状態で動作しています。

ARM 命令 ARM プロセッサが実行する操作を示すワード。ARM 命令はワードアラインしている必要があります。

ATB アドバンスト トレースバス*参照。*

ATB ブリッジ 同期 ATB ブリッジはレジスタスライスを提供するため、パイプラインステージの追加によって、タイミング収束が容易になります。また、2 つの同期 ATB ドメイン間の単方向リンクを提供します。

非同期 ATB ブリッジは、非同期クロックを使用する 2 つの ATB ドメイン間の単方向リンクを提供します。これは、異なるクロックドメインに存在するコンポーネントを ATB ポートで接続するのをサポートすることが目的です。

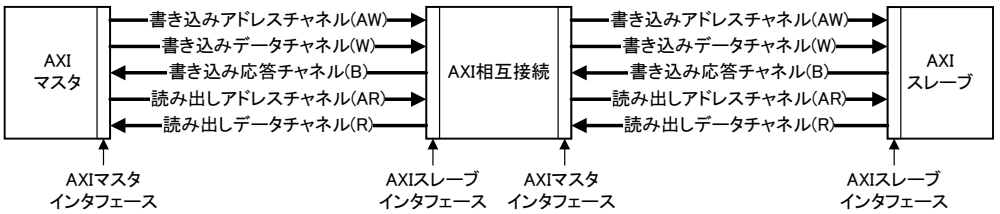
ATPG 自動テストパターン生成*参照*。

AXI アドバンスト エクステンシブルインタフェース*参照*。

AXI のチャネル順序とインタフェース

このブロック図は、次のことを示しています。

- AXI チャネル信号が記述される順序
- AXI コンポーネントのマスタ / スレーブインタフェース表記規則



AXI 用語 次の AXI 用語が一般的に使用されています。これらは、マスタとスレーブの両方に適用されます。

アクティブな書き込みトランザクション

書き込みアドレスまたは先頭の書き込みデータは転送されたが、書き込み応答がまだ転送されていない状態のトランザクション。

アクティブな転送

xVALID[1] ハンドシェークはアサートされたが、**xREADY** がまだアサートされていない状態の転送。

アクティブな読み出しトランザクション

読み出しアドレスは転送されたが、最後の読み出しデータがまだ転送されていない状態のトランザクション。

完了した転送

xVALID/xREADY ハンドシェークが完了した状態の転送。

送信 ペイロードを駆動し、関連する **xVALID** 信号をアサートする動作。

転送 単一の情報交換。つまり、1 回の **xVALID/xREADY** ハンドシェークによる処理。

トランザクション

転送のバースト全体で、アドレス、1 つまたは複数のデータ転送、応答転送（書き込みのみ）で構成されます。

ペイロード 転送に含まれる、ハンドシェーク以外の信号。

---

1. 信号名に **x** が含まれている場合、次の AXI チャネルを意味します。
   AW 書き込みアドレスチャネル
   W 書き込みデータチャネル
   B 書き込み応答チャネル
   AR 読み出しアドレスチャネル
   R 読み出しデータチャネル

次の AXI 用語は、マスタインタフェース属性です。最高のパフォーマンスを引き出すには、AXI マスタインタフェースを持つすべてのコンポーネントに対して、これらを指定する必要があります。

書き込み ID 機能

マスタインタフェースで、アクティブなすべての書き込みトランザクションについて同時に生成可能な、異なる **AWID** 値の最大数。

書き込み ID 幅

AWID バスと **WID** バスのビット数。

書き込みインターリーブ機能

マスタインタフェースでデータ送信が可能な、アクティブな書き込みトランザクションの数。この数は、最初のトランザクションからカウントされます。

書き込み発行機能

マスタインタフェースで生成可能な、アクティブな書き込みトランザクションの最大数。

統合発行機能

マスタインタフェースで生成可能なアクティブトランザクションの最大数。これは、アクティブな書き込みトランザクションと読み出しトランザクション用に同じ記憶域を使用するマスタインタフェースに対して、書き込み発行機能、または読み出し発行機能の代わりに指定されます。

読み出し ID 機能

マスタインタフェースで、アクティブなすべての読み出しトランザクションについて同時に生成可能な、異なる **ARID** 値の最大数。

読み出し ID 幅

ARID バスのビット数。

読み出し発行機能

マスタインタフェースで生成可能な、アクティブな読み出しトランザクションの最大数。

次に示す AXI 用語は、スレーブインタフェース属性です。最高のパフォーマンスを引き出すには、AXI スレーブインタフェースを持つすべてのコンポーネントに対して、これらを指定する必要があります。

書き込み受け付け機能

スレーブインタフェースで受け付け可能な、アクティブな書き込みトランザクションの最大数。

書き込みインターリーブ深度

スレーブインタフェースでデータ受信可能な、アクティブな書き込みトランザクションの数。この数は、最初のトランザクションからカウントされます。

統合受け付け機能

スレーブインタフェースで受け付け可能な、アクティブなトランザクションの最大数。これは、アクティブな書き込みトランザクションと読み出しトランザクションの両方に同じ記憶域を使用するスレーブインタフェースについて、書き込み受け付け機能、または読み出し受け付け機能の代わりに指定されます。

読み出し受け付け機能

スレーブインタフェースで受け付け可能な、アクティブな読み出しトランザクションの最大数。

読み出しデータ再順序付け深度

スレーブインタフェースでデータ送信可能な、アクティブな読み出しトランザクションの数。この数は、最初のトランザクションからカウントされます。

BE-32 ワード不変システムでの、ビッグエンディアン形式のメモリビュー。

BE-8、LE、バイト不変、ワード不変も*参照*。

BE-8 バイト不変システムでの、ビッグエンディアン形式のメモリビュー。

BE-32、LE、バイト不変、ワード不変も*参照*。

CPI 命令あたりのサイクル数*参照*。

CPSR カレントプログラム ステータスレジスタ*参照*。

DBGTAP デバッグテスト アクセスポート*参照*。

DNM 変更不可*参照*。

EmbeddedICE-RT デバッグ可能な ARM プロセッサでリアルタイム デバッグのために使用される、JTAG ベースのハードウェア。

EmbeddedICE ロジック

ARM プロセッサコアに対する TAP ベースのデバッグサポートを提供するオンチップのロジックブロック。JTAG インタフェースを使用して、ARM コア上の TAP コントローラ経由でアクセスされます。

IEEE 754 規格 *IEEE Standard for Binary Floating-Point Arithmetic, ANSI/IEEE Std. 754-1985*。浮動小数点システムに関するデータ型、正常な動作、例外のタイプと処理、エラーバウンドを規定している規格です。ほとんどのプロセッサが、ハードウェア単体またはハードウェアとソフトウェアの組み合わせによって、この規格に準拠するように構築されています。

IEM インテリジェント電力管理*参照*。

IGN 無視*参照*。

JTAG ジョイントテスト アクショングループ*参照*。

LE バイト不変とワード不変の両方のシステムにおける、リトルエンディアン形式のメモリビュー。バイト不変とワード不変も参照。

LSU ロードストア ユニット*参照*。

MMU メモリ管理ユニット*参照*。

MPU メモリ保護ユニット*参照*。

PA 物理アドレス*参照*。

RealView ICE JTAG インタフェースを使用して、組み込みプロセッサコアをデバッグするためのシステム。

SBO 常に 1 *参照*。

SBZ 常に 0 *参照*。

SBZP 常に 0 または保持*参照*。

SCREG ARM TAP コントローラ内で現在選択されているスキャンチェイン番号。

SPSR 保存プログラムステータス レジスタ*参照*。

TAP テスト アクセスポート*参照*。

Thumb 状態 Thumb（16 ビット）ハーフワードアラインド命令を実行しているプロセッサは、Thumb 状態で動作しています。

Thumb 命令 ARM プロセッサが Thumb 状態で実行する動作を指定するハーフワード。Thumb 命令は、ハーフワードアラインドの必要があります。

TLB 変換ルックアサイドバッファ*参照。*

UNP 予測不能*参照。*

VA 修飾仮想アドレス*参照。*

VA 仮想アドレス*参照。*

WB ライトバック*参照。*

WT ライトスルー*参照。*

アーキテクチャ プロセッサとその付属コンポーネントを特徴付け、同様の特徴を持つデバイスを、ハーバードアーキテクチャ、命令セットアーキテクチャ、ARMv6 アーキテクチャなどのように、その動作を記述するときにグループ化することを可能にする、ハードウェアおよびソフトウェアの編成。

アドバンスト エクステンシブルインタフェース (AXI)
独立したアドレス / 制御フェーズとデータフェーズ、バイトストローブによるアンアラインドデータ転送、開始アドレスのみの発行によるバーストベースのトランザクション、低コスト DMA を可能にする独立した読み出しおよび書き込みデータチャネル、複数の未解決アドレスの発行機能、アウトオブオーダートランザクションの実行、レジスタステージの追加が容易なことによるタイミングクロージャの提供をサポートするバスプロトコル。AXI プロトコルには、低消費電力動作の信号処理をカバーするオプションの拡張機能も含まれます。

AXI は、パフォーマンスが高く、クロック周波数が高いシステムの設計を目的としており、高速のサブミクロン相互接続に最適な多くの機能が含まれています。

アドバンスト トレースバス (ATB)
CoreSight のトレースキャプチャ資源を共有するため、トレースデバイスによって使用されるバス。

アドバンスト ハイパフォーマンスバス (AHB)
アドレス / 制御フェーズとデータフェーズとの間で 1 つの固定パイプラインを使用するバスプロトコル。AMBA AXI プロトコルで提供されている機能のサブセットのみをサポートします。完全な AMBA AHB プロトコル仕様には、一般的なマスタ / スレーブ IP 開発では必要とされない機能が多く含まれているため、通常はプロトコルのサブセットだけを使用することをお勧めします。このサブセットは、AMBA AHB-Lite プロトコルとして定義されています。

アドバンスト マイクロコントローラバス アーキテクチャと AHB-Lite *も参照。*

アドバンスト ペリフェラルバス (APB)
AXI や AHB よりも単純なバスプロトコル。タイマ、割り込みコントローラ、UART、I/O ポートなどの補助的な、または汎用のペリフェラルで使用するために設計されています。メインのシステムバスへの接続は、システムとペリフェラルとの間のバスブリッジを経由して行われるため、システムの消費電力を抑えられます。

アドバンスト マイクロコントローラバス アーキテクチャ (AMBA)
相互接続のための方針が記載された、プロトコル仕様のファミリ。AMBA は、オンチップバスに関する ARM のオープンな規格です。*システムオンチップ*(SoC) を構築する機能ブロックの相互接続と管理のための方針が記載された、オンチップバスの仕様です。1 つまたは複数の CPU や信号プロセッサ、および複数のペリフェラルを含む組み込みプロセッサの開発に役立ちます。AMBA は、SoC モジュール用の共通バックボーンを定義することによって、再利用可能な設計手法をより完全なものにします。

アドレシングモード
命令で使用する値を生成するために、多くの命令で共有される機構。ARM アドレシングモードのうち 4 つでは、値としてメモリアドレスが生成されます。これは、アドレシングモードの従来の役割です。5 番目のアドレシングモードでは、データ処理命令のオペランドとして使用される値が生成されます。

アボート
メモリアクセスに関連する値が無効であることをコアに通知する機構。アボートは、無効な命令またはデータメモリへのアクセスを実行した結果として、外部または内部のメモリシステムで発生する可能性があります。アボートは、プリフェッチアボートとデータアボート、内部アボートと外部アボートに分類されます。

データアボート、外部アボート、プリフェッチアボートも*参照*。

アボートモデル
アボートモデルは、データアボート例外に対する ARM プロセッサの応答として定義された動作です。アボートモデルによって、ベースレジスタのライトバックを指定するロード / ストア命令に関する動作が異なります。

アラインド
データサイズを定義しているバイト数で割り切れるアドレスに格納されているデータ項目を、アラインド、またはアラインしていると呼びます。アラインしているワードとハーフワードのアドレスは、それぞれ 4 と 2 で割り切れます。したがって、ワードアラインドとハーフワードアラインドという用語は、それぞれ 4 と 2 で割り切れるアドレスを意味します。

インデクス
キャッシュインデクス*参照*。

インデクスレジスタ
一部のロード / ストア命令で指定されるレジスタ。このレジスタの値は、メモリに送信される仮想アドレスを生成するために、オフセットとしてベースレジスタの値に加算または減算されます。一部のアドレシングモードでは、必要に応じて、インデクスレジスタの値をシフトしてから加算または減算できます。

インテリジェント電力管理（IEM）
デバイスの消費電力を低減するために使用される技術で、動的な電圧スケーリングとクロック周波数の変更を可能にします。

ヴィクティム
キャッシュミスのためにキャッシュラインの置き換えが必要となった場合、必要な領域を確保するために破棄されるキャッシュライン。取り出すヴィクティムを選択する方法は、プロセッサによって異なります。ヴィクティムはキャストアウトと呼ばれることもあります。

ウェイ
キャッシュウェイ*参照*。

ウォームリセット
コアリセットと呼ばれることもあります。デバッグコントローラとデバッグロジックを除く、プロセッサの大部分を初期化します。この種類のリセットは、プロセッサのデバッグ機能を使用している場合に便利です。

ウォッチポイント
ウォッチポイントは、デバッガで提供されている機構で、特定のメモリアドレスに保存されているデータが変更されたときにプログラムの実行を停止します。プログラマは、ウォッチポイントを挿入することによって、メモリが書き込まれたときのレジスタの内容、メモリの位置、変数の値を検査して、プログラムが正常に動作しているかどうかをテストできます。プログラムのテストが完了した後で、ウォッチポイントは削除されます。ブレークポイントも*参照*。

エンディアン形式
バイトの順序。データワードの連続するバイトがメモリに保存される順序を決定する方式。システムのメモリマッピングの見え方。

リトルエンディアンとビッグエンディアンも*参照*。

外部アボート
不正なメモリアクセスの実行を停止しなければならないことを、外部メモリシステムからコアに通知する手段。外部アボートは、無効なメモリへのアクセスを試みた結果として、外部メモリシステムによって引き起こされます。

アボート、データアボート、プリフェッチアボートも*参照*。

書き込み
書き込みは、ストアの意味を持つ操作として定義されます。ARM 命令の SRS、STM、STRD、STC、STRT、STRH、STRB、STRBT、STREX、SWP、SWPB と、Thumb 命令の STM、STR、STRH、STRB、PUSH が該当します。ハードウェアで高速化される Java バイトコードの場合は、Java スタックの状態と Java ハードウェアアクセラレーションの実装によっては、大量の書き込みが発生する可能性があります。

書き込み完了
メモリシステムは、トランザクション内で、書き込みの効果がシステム内のすべてのプロセッサに可視になったことを保証できる時点で、書き込みが完了したことをプロセッサに通知します。書き込みが、メモリ同期化基本命令に関連している場合や、デバイス領域またはストロングリオーダ領域を対象としている場合、これには該当しません。このような場合、書き込みの効果が可視なことと、ターゲットの状態が更新されたこととの区別が不可能である場合を除いて、メモリシステムは、アクセスによってターゲットの状態が変更されたときだけ、書き込みの完了を通知できます。

一部の種類のメモリに対するこの厳格な要件によって、メモリアクセスの副作用がすべて発生したことをプロセッサで保証できます。この機能を使用すると、副作用が可視になるまで、プログラムの順序でその後に行われる動作の開始を保留することができます。

仮想アドレス (VA)
MMU は、変換テーブルを使用して、仮想アドレスを物理アドレスに変換します。プロセッサは仮想アドレスでコードを実行し、そのアドレスは物理メモリでは別の場所に存在する可能性があります。

修飾仮想アドレスと物理アドレスも*参照*。

カレントプログラム ステータスレジスタ (CPSR)
現在動作中のプロセッサのステータスを保持しているレジスタ。

完全アソシエイティブキャッシュ
キャッシュ全体が 1 つのキャッシュセットだけで構成されているキャッシュ。キャッシュエントリの数は、キャッシュウェイの数と同じです。

直接マップキャッシュも*参照*。

キャッシュ
プロセッサとメインメモリの間に配置され、使用頻度の高い命令やデータのコピーを格納および取得するために使用される、オンチップまたはオフチップの高速アクセスメモリ位置のブロック。これによって、メモリアクセスの平均速度が大幅に向上するため、プロセッサのパフォーマンスも向上します。

この用語集の最後のページにあるキャッシュ用語の図も*参照*。

キャッシュウェイ
キャッシュラインまたはブロックのグループ。キャッシュウェイのサイズは、2 の（インデクスのビット数）乗です。

この用語集の最後のページにあるキャッシュ用語の図も*参照*。

キャッシュセット
キャッシュセットは、キャッシュラインまたはブロックのグループです。キャッシュセットには、同じインデクスでアドレス指定が可能なすべてのウェイが含まれます。キャッシュセットの数は、必ず 2 のべき乗です。

この用語集の最後のページにあるキャッシュ用語の図も*参照*。

キャッシュの競合
特定のキャッシュセットを使用する、使用頻度の高いメモリ キャッシュラインの数が、キャッシュのセットアソシエティビティを超えたとき。この場合、メインメモリの動作が増大して、パフォーマンスが低下します。

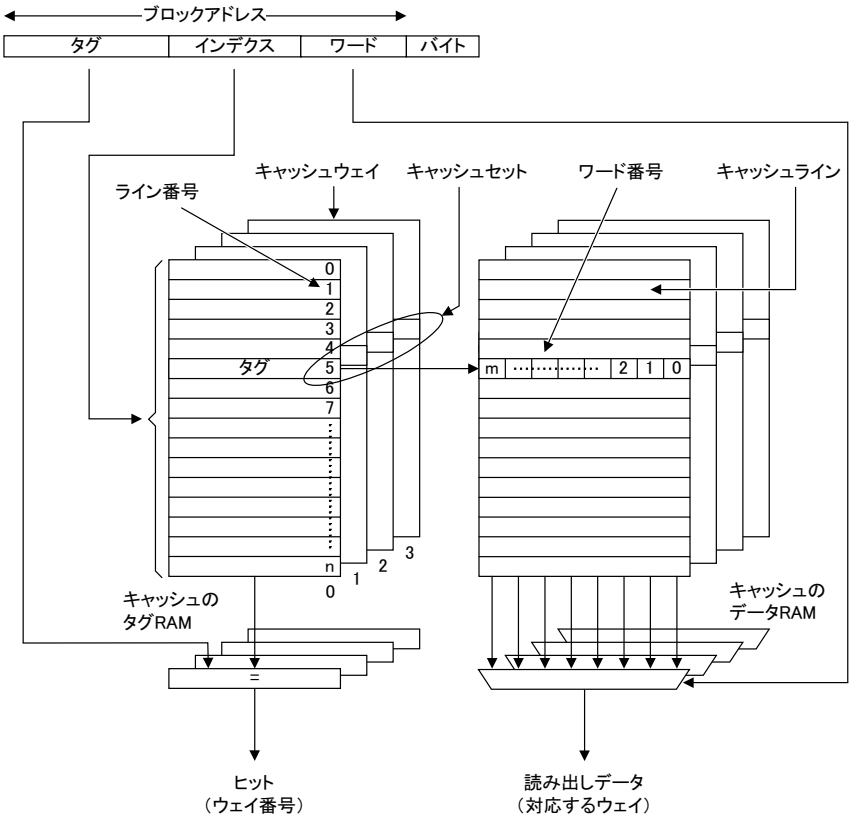
キャッシュヒット
命令または命令がアクセスするデータがすでにキャッシュに保持されているため、高速で処理可能なメモリアクセス。

キャッシュミス
命令または命令がアクセスするデータがキャッシュに存在しないために、メインメモリへのアクセスが必要になり、高速で処理できないメモリアクセス。

キャッシュ用語の図
この図は、次のキャッシュ用語を示したものです。

- ブロックアドレス
- キャッシュライン

- キャッシュセット
- キャッシュウェイ
- インデクス
- タグ



キャッシュライン

キャッシュ内の記憶域の基本単位。キャッシュラインは、サイズが常に 2 のべき乗（通常は 4 または 8 ワード）で、適切なメモリ境界にアラインしている必要があります。

この用語集の最後のページにあるキャッシュ用語の図も*参照*。

キャッシュライン インデクス

キャッシュウェイ内の各キャッシュラインに関連付けられた番号。各キャッシュウェイ内のキャッシュラインには、0 から（セットアソシエティビティ - 1）までの番号が付けられます。

この用語集の最後のページにあるキャッシュ用語の図も*参照*。

キャッシュロックダウン

上書きされないように、キャッシュメモリ内のラインを固定すること。重要な命令やデータをキャッシュにロードし、それらを含むキャッシュラインが後から再割り当てされないようにすることができます。これによって、対象の命令やデータに対する以降のアクセスが、必ずキャッシュヒットになるため、可能な限り高速に実行されることが保証されます。

キューの先頭ポインタ ライトバッファ内で、次に書き込まれるエントリへのポインタ。

クリーニング キャッシュ内で変更されていないキャッシュラインを、クリーンなキャッシュラインと呼びます。キャッシュをクリーニングするとは、ダーティなキャッシュエントリをメインメモリに書き込むことです。キャッシュラインがクリーンな場合は、次のレベルのメモリにキャッシュと同じデータが保持されているため、キャッシュミスが発生したときにキャッシュラインが書き込まれることはありません。

ダーティも参照。

クロックゲート 制御信号でマクロセルのクロック信号をゲートし、そのクロックを使用することによって、マクロセルの動作状態が制御されます。

コアリセット ウォームリセット*参照。*

コールドリセット パワーオン リセットと呼ばれることもあります。電力オンによりプロセッサが起動することを意味します。電力をオフにしてから再度オンにすると、メインメモリと多くの内部設定がクリアされます。プログラムの障害によってはプロセッサがロックし、システムを再度使用可能にするためにコールドリセットが必要な場合があります。それ以外の場合は、ウォームリセットのみが必要です。

ウォームリセット*も参照。*

コヒーレンシ メモリコヒーレンシ*参照。*

コプロセッサ メインプロセッサを補完するプロセッサ。メインプロセッサが実行できない付加機能を実行します。通常は、浮動小数点算術演算、信号処理、メモリ管理などに使用されます。

コンテキスト マルチタスクのオペレーティングシステムで、各プロセスが動作する環境。ARM プロセッサでは、メモリ内でアクセス可能な物理アドレス範囲と、それに関連付けられたメモリアクセス許可の意味に限定されます。

再マッピング アプリケーションの実行開始後に、物理メモリまたはデバイスのアドレスを変更すること。通常、この動作は、初期化完了後に ROM を RAM に置き換えるために行われます。

指数 浮動小数点数の構成要素で、表現される数値の値を決定するために使用され、通常は 2 の整数乗を示します。

実装固有 動作がアーキテクチャで定義されていないが、実装ごとに文書化する必要がないことを意味します。使用可能な実装オプションが多数あり、選択したオプションによってソフトウェアの互換性に影響がない場合に使用されます。

実装定義 動作がアーキテクチャで定義されていないため、実装ごとに定義して文書化する必要があることを意味します。

自動テストパターン生成 (ATPG)

特殊なソフトウェアツールを使用して、ASIC 設計用の製造テストベクタを自動生成する処理。

ジョイントテスト アクショングループ (JTAG)

IEEE 1149.1 規格を策定した団体の名前。この規格では、集積回路デバイスのインサーキット テストに使用される、バウンダリスキャン アーキテクチャが定義されています。頭文字の JTAG で広く知られています。

条件付き実行 条件コードフラグが、命令の実行開始時に該当する条件が TRUE であることを示している場合は、命令が正常に実行されます。それ以外の場合、命令は何も実行しません。

条件フィールド 命令が実行可能な条件を指定する、命令内の 4 ビットのフィールド。

スキャンチェイン スキャンチェインはシリアル接続されたデバイスで構成され、標準の JTAG TAP インタフェースを使用してバウンダリスキャン技術を実装しています。各デバイスには少なくとも 1 つの TAP コントローラがあり、**TDI** と **TDO** との間の接続チェインを形成するシフトレジスタを搭載しています。このチェインを通して、テストデータがシフトされます。プロセッサには数個のシフトレジスタを搭載できるため、デバイスの選択した部分にアクセスできます。

制御ビット プログラムステータス レジスタ (PSR) の最下位 8 ビット。制御ビットは、例外が発生したときに変化します。プロセッサが特権モードの場合にのみ、ソフトウェアから変更できます。

セット キャッシュセット *参照*。

セットアソシエイティブ キャッシュ
セットアソシエイティブ キャッシュでは、メモリアドレスをセット数で割った剰余に対応するキャッシュ内の位置にのみ、ラインを配置できます。キャッシュ内に *n* 個のウェイがある場合、そのキャッシュは *n* ウェイ セットアソシエイティブと呼ばれます。セットアソシエティビティには、1 以上の任意の値を使用することができ、必ずしも 2 のべき乗にする必要はありません。

ダーティ ライトバック キャッシュ内で変更されたキャッシュラインを、ダーティなキャッシュラインと呼びます。キャッシュラインは、ダーティビットをセットすることによって、ダーティとしてマークされます。キャッシュラインがダーティな場合は、次のレベルのメモリに更新されていないデータが保持されているため、キャッシュミスが発生したときにキャッシュラインをメモリに書き込む必要があります。ダーティデータをメインメモリに書き込む処理を、キャッシュのクリーニングと呼びます。

クリーニング も *参照*。

タグ キャッシュ内のキャッシュラインの識別に使用されるブロックアドレスの上位部分。CPU からのブロックアドレスは、セット内の各タグと並列に比較され、対応するラインがキャッシュに存在するかどうかが判断されます。存在する場合はキャッシュヒットとなり、そのラインをキャッシュからフェッチできます。ブロックアドレスがどのタグにも対応しない場合はキャッシュミスとなり、そのラインは次のレベルのメモリからフェッチされる必要があります。

この用語集の最後のページにあるキャッシュ用語の図 も *参照*。

ダブルワード 64 ビットのデータ項目。特に指定のない限り、その内容は符号なし整数とみなされます。

ダブルワードアラインド
メモリアドレスが 8 で割り切れるデータ項目。

直接マップキャッシュ 1 ウェイのセットアソシエイティブキャッシュ。各キャッシュセットは、単一のキャッシュラインで構成されているため、キャッシュのルックアップでは、単一のキャッシュラインが選択されてチェックが行なわれます。

通信チャネル デバッグインタフェースを使用して、プロセッサ上で実行中のソフトウェアと外部ホストとの間で通信を行うために使用されるハードウェア。この通信がデバッグ目的の場合は、デバッグ通信チャネルと呼ばれます。ARMv6 準拠のコアの通信チャネルには、データ転送レジスタ、データステータスおよび制御レジスタの一部のビット、および JTAG インタフェースにおける DBGTAP コントローラのような外部デバッグインタフェース コントローラが含まれます。

常に 0 (SBZ) ソフトウェアで 0（ビットフィールドの場合はすべてのビットに 0）を書き込む必要があります。1 を書き込んだ場合、結果は予測不能です。

常に 0 または保持 (SBZP)
ソフトウェアで 0（ビットフィールドの場合はすべてのビットに 0）を書き込むか、同じプロセッサの同じフィールドから以前に読み出した値をそのまま書き戻して保持する必要があります。

常に 1 (SBO)
ソフトウェアで 1（ビットフィールドの場合はすべてのビットに 1）を書き込む必要があります。0 を書き込んだ場合、結果は予測不能です。

データアボート
不正なメモリアクセスの実行を停止しなければならないことを、メモリシステムからコアに通知する手段。データアボートは、無効なデータメモリにアクセスしようとしたときに発生します。

アボート、外部アボート、プリフェッチアボートも*参照*。

データキャッシュ
プロセッサとメインメモリとの間に配置され、使用頻度の高いデータのコピーを格納および取得するために使用される、オンチップの高速アクセスメモリ位置のブロック。これによって、メモリアクセスの平均速度が大幅に向上するため、プロセッサのパフォーマンスも向上します。

テスト アクセスポート (TAP)
JTAG バウンダリスキャン アーキテクチャの入出力インタフェースと制御インタフェースを形成する、4 つの必須端子と 1 つのオプション端子の集合。必須端子は、**TDI**、**TDO**、**TMS**、**TCK** です。オプション端子は、**TRST** です。この信号は、デバッグロジックのリセットに使用されるため、ARM コアには不可欠です。

デバッガ
ソフトウェアの障害を検出し、場所を特定し、修正するために使用されるプログラムと、ソフトウェアのデバッグをサポートするカスタムハードウェアとを組み合わせたデバッグシステム。

デバッグアクセス ポート (DAP)
システムバスへのアクセスで、AMBA（AHB または AHB-Lite）マスタとして動作する TAP ブロック。DAP は、システム規模のデバッグをサポートする、モジュラーブロックのセットを総称するために使用される用語です。DAP はモジュラーコンポーネントで、単一のデバッグインタフェースを通して、メモリマップされた AHB や CoreSight APB など複数のシステムに対して、任意のアクセスをサポートするよう拡張可能なように意図されています。

同期化基本命令
メモリ同期化基本命令は、メモリの同期を保証するために使用される命令です。これには、LDREX、STREX、SWP、SWPB 命令が含まれます。

トラップ
VFP コプロセッサに例外状態が発生し、FPSCR レジスタの対応する例外イネーブルビットがセットされている場合。ユーザトラップ ハンドラが実行されます。

内部スキャンチェイン
一連のレジスタが互いに接続され、デバイスを経由するパスを形成したもので、デバイスの内部ノードへテストパターンをインポートし、結果の値をエクスポートする運用テストで使用されます。

バースト
連続アドレスに対する一連の転送。アドレスが連続しているため、2 回目以降の転送ではアドレスを指定する必要がありません。この方法によって、一連の転送の実行速度が向上します。AHB バス上のバーストは、**HBURST** 信号を使用して制御されます。この信号によって、転送が 1 ビート、4 ビート、8 ビート、16 ビートのどれであるか、アドレスがどのようにインクリメントされるかが指定されます。

ビートも*参照*。

ハーフワード
16 ビットのデータ項目。

バイト
8 ビットのデータ項目。

バイト不変
バイト不変のシステムでは、リトルエンディアンとビッグエンディアンの動作が切り替えられても、メモリの各バイトのアドレスは変更されません。1 バイトを超えるデータ項目をメモリからロード、またはメモリにストアするとき、そのデータ項目を構成するバイトが、メモリアクセスのエンディアン形式に応じて正しい順序に配列されます。ARM アーキテクチャでは、ARMv6 およびそれ以降のバージョンでバ

イト不変システムがサポートされています。バイト不変のサポートが選択されている場合は、アンアラインドのハーフワードとワードでのメモリアクセスもサポートされます。複数ワードアクセスは、ワードアラインしている必要があります。

ワード不変も*参照*。

バイトレーン ストローブ

AHB 信号 **HBSTRB** のことで、アンアラインドまたはエンディアン混在のデータアクセスを行うとき、転送に含まれるアクティブなバイトレーンを決定するために使用されます。**HBSTRB** の 1 ビットは、データバスの 8 ビットに相当します。

ハイベクタ

例外ベクタの代替位置。ハイベクタのアドレス範囲は、アドレス空間の最下位ではなく、最上位付近にあります。

バウンダリスキャン チェイン

バウンダリスキャン チェインは、標準の JTAG TAP インタフェースを使用してバウンダリスキャン技術を実装しているデバイスのシリアル接続で構成されます。各デバイスには少なくとも 1 つの TAP コントローラがあり、**TDI** と **TDO** との間の接続チェインを形成するシフトレジスタを搭載しています。このチェインを通して、テストデータがシフトされます。プロセッサには数個のシフトレジスタを搭載できるため、デバイスの選択した部分にアクセスできます。

パワーオンリセット

コールドリセット*参照*。

バンクレジスタ

現在のプロセッサモードによって用途が定義される物理レジスタ。バンクレジスタは、r8 ～ r14 です。

ビート

バースト内の個別の転送を意味する別の用語。例えば、INCR4 バーストは 4 ビートで構成されます。

バーストも*参照*。

ビッグエンディアン

データワードの最上位バイトから最下位バイトまでが、メモリアドレスの昇順に保存されるバイト配列方式。

リトルエンディアンとエンディアン形式も*参照*。

ビッグエンディアン メモリ

次のような条件が満たされるメモリを指します。

- ワードアラインしたアドレスのバイトまたはハーフワードが、そのアドレスのワード内で最上位のバイトまたはハーフワードである。
- ハーフワードアラインされたアドレスのバイトが、そのアドレスのハーフワード内で最上位バイトである。

リトルエンディアン メモリも*参照*。

標準遅延フォーマット (SDF)

バスの各ビットレベルまでのタイミング情報が含まれ、SDF バックアノテーションで使用されるファイル形式。SDF ファイルは複数の方法で生成できますが、遅延計算機から生成されるのが最も一般的です。

不正確トレース

命令またはデータのトレースが、予想よりも早くまたは遅く、開始または終了する可能性があるフィルタリング構成。ほとんどの場合、トレースは予想よりも遅く開始または終了します。

例えば、メモリ上の特定の位置への 4 回目の書き込み後にトレースを開始し、4 回目の書き込みを実行した命令はトレースせず、その後の命令をトレースするように、カウンタを使用して **TraceEnable** を構成した場合です。これは、**TraceEnable** の構成でカウンタを使用した場合は、必ず不正確トレースが実行されるためです。

不正命令

アーキテクチャで未定義の命令。

物理アドレス (PA) MMU は、*修飾仮想アドレス* (VA) に対して変換を実行し、AXI に渡して外部アクセスを実行するための *物理アドレス* (PA) を生成します。また、PA は、データがキャッシュからキャストアウトされたときにアドレス変換の必要がないように、データキャッシュにも格納されます。

フラット アドレスマッピング
メモリ空間内の物理アドレスと、対応する仮想アドレスが等しくなるようにメモリを編成する方式。

プリフェッチ パイプライン処理のプロセッサで、先行する命令の実行が完了する前に、その後の命令をメモリからフェッチしてパイプラインに送り込む処理。プリフェッチされた命令は、必ず実行されるとは限りません。

プリフェッチアボート 不正なメモリアクセスの実行を停止しなければならないことを、メモリシステムからコアに通知する手段。プリフェッチアボートは、無効な命令メモリへのアクセスを試みた結果として、外部または内部のメモリシステムにより引き起こされる可能性があります。

データアボート、外部アボート、アボートも*参照*。

ブレークポイント プログラムの実行を停止させる位置にある命令を特定するために、デバッガによって提供される機構。プログラマは、ブレークポイントを挿入することによって、プログラムの実行中の決まった位置で、レジスタの内容、メモリの位置、変数の値を検査して、プログラムが正常に動作しているかどうかをテストできます。プログラムのテストが完了した後で、ブレークポイントは削除されます。

ウォッチポイントも*参照*。

プロセッサ コンピュータ命令を使用してデータを処理するために必要な、コンピュータシステムの回路。プロセッサは、マイクロプロセッサの略称です。完全に機能する最小のコンピュータシステムを作成するには、クロックソース、電源、メインメモリも必要です。

ブロックアドレス タグ、インデクス、ワードフィールドで構成されるアドレス。タグビットによって、キャッシュヒット時に照合するキャッシュエントリを保持するウェイが識別されます。インデクスビットによって、アドレス指定されるセットが識別されます。ワードフィールドには、キャッシュエントリ内の特定のワード、ハーフワード、バイトの識別に使用可能なワードアドレスが格納されます。

この用語集の最後のページにあるキャッシュ用語の図も*参照*。

分岐予測 パイプライン化されたプロセッサで、条件付き分岐が行なわれるかどうかを予測する処理。分岐の発生を正確に予測することによって、プロセッサは、完全に条件が解決される前に、分岐以後の命令をプリフェッチできます。分岐予測は、ソフトウェアで、またはカスタムハードウェアを使用して実行できます。分岐予測手法は、ランタイム前に予測が決定される場合は静的として、プログラムの実行中に予測決定を変更できる場合は動的として分類されます。

ベースレジスタ 命令のアドレス計算の基準値を保持するために、ロード / ストア命令で指定されるレジスタ。命令とそのアドレシングモードによっては、メモリに送られる仮想アドレスを形成するために、ベースレジスタの値にオフセットを加算または減算できます。

ベースレジスタ ライトバック
命令のターゲットアドレス計算に使用されるベースレジスタの内容を、アドレスがメモリ順序で次の上位または下位アドレスを指し示すように更新すること。これによって、命令で連続した転送を行うとき、そのたびにターゲットアドレスをフェッチする必要がなく、連続するメモリに対してより高速なバーストアクセスが可能になります。

ペナルティ 命令フローが仮定または予想と異なるため、実行ステージの有効なパイプライン動作が発生しないサイクル数。

変換テーブル メモリ内に保持されるテーブルで、さまざまな固定サイズのメモリ領域のプロパティを定義したデータが含まれます。

変換テーブルウォーク 完全な変換テーブルルックアップを実行する処理。ハードウェアによって自動的に実行されます。

変換ルックアサイドバッファ (TLB)
最近使用されたページテーブル エントリのキャッシュで、メモリアクセスのたびにページテーブル ウォークを行うオーバヘッドを回避するために使用されます。メモリ管理ユニットの一部分です。

変更不可 (DNM) 変更不可フィールドの値は、ソフトウェアで変更しないようにする必要があります。DNM フィールドは、予測不能な値として読み出され、同じプロセッサの同じフィールドから読み出された同じ値のみを書き込む必要があります。DNM フィールドについての記載では、括弧付きの RAZ または RAO が続いて、将来の互換性のためにビットを読み出す方法を示している場合がありますが、プログラマはこの動作を前提とすべきではありません。

ホールト デバッグモード
互いに排他な 2 つのデバッグモードのうちの 1 つ。ホールト デバッグモードでは、ブレークポイントやウォッチポイントなどの*デバッグイベント*が発生すると、プロセッサが特別なデバッグ状態に入ります。デバッグ状態では、プロセッサが外部デバッグインタフェースを通して制御されます。このインタフェースは、プロセッサ状態、コプロセッサ状態、メモリ、I/O 位置のすべてにアクセスできます。

モニタ デバッグモード*も参照。*

ホスト データや他のサービスを別のコンピュータに提供するコンピュータ。特に、デバッグ対象のターゲットにデバッグサービスを提供するコンピュータ。

保存プログラムステータス レジスタ (SPSR)
現在のモードへの切り替えを引き起こした例外が発生する直前の、タスクの CPSR を保持しているレジスタ。

マイクロプロセッサ プロセッサ*参照。*

マクロセル インタフェースと動作が定義された複合論理ブロック。一般的な VLSI システムは、複数のマクロセル（プロセッサ、ETM、メモリブロックなど）と、特定用途の論理回路で構成されます。

未サポート値 VFP コプロセッサのハードウェアでは処理されず、サポートコードにバウンスされて完了される特定のデータ値。このようなデータには、無限大、NaN、通常と異なる値、0 が含まれます。これらの値をハードウェアで完全にまたは部分的にサポートするか、サポートコードにその処理の完了をゆだねるかは、実装で選択できます。未サポートデータの処理から発生したすべての例外は、対応する例外イネーブルビットがセットされている場合に、ユーザコードにトラップされます。

ミス キャッシュミス*参照。*

未定義 未定義命令トラップを生成する命令を指します。ARM 例外の詳細については、*『ARM アーキテクチャ リファレンスマニュアル』*を参照して下さい。

無効化 有効ビットをクリアし、キャッシュラインを無効としてマークすること。この処理は、キャッシュラインに有効なキャッシュエントリが含まれていない場合に必ず実行する必要があります。例えば、キャッシュのフラッシュ後は、すべてのラインが無効になります。

無視 (IGN) メモリ書き込みを無視する必要があります。

命令あたりのクロック数 (CPI)
命令あたりのサイクル数 (CPI) *参照。*

命令あたりのサイクル数 (CPI)
命令あたりのサイクル数（または命令あたりのクロック数）は、1 クロックサイクルで実行可能なコンピュータ命令の数の指標です。この性能指標は、同じ命令セットを実装した異なる CPU のパフォーマンスを比較するために使用できます。値が低いほど、パフォーマンスが高いことを意味します。

命令キャッシュ
プロセッサとメインメモリとの間に配置され、使用頻度の高い命令のコピーを格納および取得するために使用される、オンチップの高速アクセスメモリ位置のブロック。これによって、メモリアクセスの平均速度が大幅に向上するため、プロセッサのパフォーマンスも向上します。

命令サイクル数
命令がパイプラインの実行ステージを占有するサイクル数。

メモリ管理ユニット (MMU)
キャッシュと、メモリブロックに対するアクセス許可を制御し、仮想アドレスを物理アドレスに変換するハードウェア。

メモリコヒーレンシ
メモリは、データ読み出しまたは命令フェッチによって読み出された値が、最後にその位置に書き込まれた値と一致していれば、コヒーレントです。メモリコヒーレンシは、メインメモリ、ライトバッファ、キャッシュを搭載したシステムのように、対応する物理位置が複数存在する場合には、実現が難しくなります。

メモリバンク
インターリーブされているメモリにおいて、並列にいくつかに分割されているメモリのうちの 1 つで、通常は 1 ワード幅です。これによって、一度に単一ワードではなく、複数ワードを読み書きできます。すべてのメモリバンクは同時にアドレス指定され、バンクイネーブル信号またはチップセレクト信号によって、転送ごとにアクセスされるバンクが決定されます。連続したワードアドレスへアクセスすると、連続したバンクへのアクセスが発生します。これによって、バンクアクセスに関連する遅延は隣接バンクへのアクセス中に発生するため、メモリ転送が高速化されます。

メモリ保護ユニット (MPU)
メモリブロックに対するアクセス許可を制御するハードウェア。MMU とは異なり、MPU は仮想アドレスから物理アドレスへの変換は行ないません。

モニタ デバッグモード
互いに排他な 2 つのデバッグモードのうちの 1 つ。モニタ デバッグモードでは、プロセッサは、デバッグモニタまたはオペレーティングシステムのデバッグタスクで提供されるソフトウェア アボートハンドラを稼働します。これによって、ブレークポイントまたはウォッチポイントに遭遇して、通常のプログラム実行が中断している間であっても、重要なシステム割り込み処理を継続することができます。

ホールトモード *も参照*。

予測不能
読み出しの場合は、この位置から読み出しによって返されるデータが予測不能なことを意味します。データはどのような値にもなり得ます。書き込みの場合は、この位置への書き込みによって予測不能な動作が発生するか、デバイスの構成に予測不能な変化が発生することを意味します。予測不能な命令によって、プロセッサまたはシステムのいずれかの部分に停止やハングが発生しないようにする必要があります。

読み出し
読み出しは、ロードの意味を持つメモリ操作として定義されます。ARM 命令の LDM、LDRD、LDC、LDR、LDRT、LDRSH、LDRH、LDRSB、LDRB、LDRBT、LDREX、RFE、STREX、SWP、SWPB と、Thumb 命令の LDM、LDR、LDRSH、LDRH、LDRSB、LDRB、POP が該当します。ハードウェアで高速化される Java バイトコードの場合は、Java スタックの状態と Java ハードウェアアクセラレーションの実装によっては、大量の読み出しが発生する可能性があります。

予約
制御レジスタまたは命令の形式に含まれているフィールドが実装で定義される、または 0 ではない場合に予測不能な結果が引き起こされる場合、そのフィールドは予約と記載されています。これらのフィールドは、アーキテクチャの将来の拡張に備えて予約される場合と、実装固有の場合があります。実装で使用されないすべての予約ビットは、0 として読み書きする必要があります。

ライトスルー (WT)
ライトスルー キャッシュでは、キャッシュが更新されると同時にデータがメインメモリに書き込まれます。

ライトバック (WB)
ライトバックキャッシュでは、キャッシュミスに続くラインの置き換えでキャッシュから削除されたデータのみが、メインメモリに書き込まれます。それ以外の場合は、プロセッサの書き込みによってのみ、キャッシュが更新されます。コピーバックと呼ばれることもあります。

ライトバッファ データキャッシュとメインメモリとの間に、FIFO バッファとして配置されている高速メモリブロック。メインメモリへのストアを最適化するために使用されます。

ライン キャッシュライン*参照*。

リトルエンディアン データワードの最下位バイトから最上位バイトまでが、メモリアドレスの昇順に保存されるバイト配列方式。

ビッグエンディアンとエンディアン形式も*参照*。

リトルエンディアン メモリ

次のような条件が満たされるメモリを指します。

- ワードアラインされたアドレスのバイトまたはハーフワードが、そのアドレスにあるワード内の最下位のバイトまたはハーフワードである。
- ハーフワードアラインされたアドレスのバイトが、そのアドレスにあるハーフワード内の最下位バイトである。

ビッグエンディアン メモリも*参照*。

領域 命令またはデータメモリ空間の一部分。

例外 プログラムの実行に割り込む必要があるほど重大であると判断された、フォールトまたはエラーイベント。例として、無効なメモリアクセス、外部割り込み、未定義命令の実行などが挙げられます。例外が発生すると、通常のプログラムフローが中断され、対応する例外ベクタで実行が再開されます。例外ベクタには、例外を処理する割り込みハンドラの最初の命令が含まれています。

例外処理ルーチン 割り込みハンドラ*参照*。

例外ベクタ 割り込みベクタ*参照*。

ロード / ストアアーキテクチャ

データ処理操作が、メモリの内容に対して直接ではなく、レジスタの内容に対してのみ行われるプロセッサアーキテクチャ。

ロードストア ユニット (LSU)

プロセッサで、ロード / ストア転送を処理する部分。

ワード 32 ビットのデータ項目。

ワード不変 ワード不変システムでは、リトルエンディアン動作とビッグエンディアン動作との切り替え時に、各メモリバイトのアドレスが変更されます。これによって、一方のエンディアン形式でアドレス A が割り当てられたバイトは、他方のエンディアン形式ではアドレス A EOR 3 が割り当てられます。このため、メモリのアラインされたワードは、エンディアン形式に関係なく、常にメモリ上の同じ 4 バイトに同じ順序で構成されます。エンディアン形式の切り替えは、バイト配列が変わるためではなく、バイトアドレスが変更されるために発生します。ARM アーキテクチャでは、ARMv3 以降のバージョンでワード不変システムがサポートされています。ワード不変のサポートが選択されている場合、アンアラインドアドレスが指定されたロード / ストア命令の動作は命令によって異なり、通常は、アンアラインドアクセスに対して予測される動作にはなりません。ワード不変システムでは、エンディアン形式が設定される前のリセットハンドラの冒頭部分を除いては、常に期待通りのバイトアドレスが生成されるエンディアン形式を使用し、リセットハンドラの冒頭部分では、アラインしたワードメモリ アクセスのみを使用することをお勧めします。

バイト不変も*参照*。

割り込みハンドラ 割り込みが発生したときに、プロセッサの制御が渡されるプログラム。

割り込みベクタ 下位メモリ、またはハイベクタが構成されている場合は上位メモリの複数の固定アドレスの 1 つで、対応する割り込みハンドラの最初の命令が格納されています。