

(19) 日本国特許庁(JP)

(12) 特 許 公 報(B2)

(11) 特許番号

特許第3920994号
(P3920994)

(45) 発行日 平成19年5月30日(2007.5.30)

(24) 登録日 平成19年2月23日(2007.2.23)

(51) Int. Cl.

G 0 6 F 15/16 (2006.01)

F I

G 0 6 F 15/16 6 2 0 G

請求項の数 11 (全 25 頁)

(21) 出願番号	特願平10-259364	(73) 特許権者	504199127
(22) 出願日	平成10年8月28日(1998.8.28)		フリースケール セミコンダクター イン
(65) 公開番号	特開平11-154144		コーポレイテッド
(43) 公開日	平成11年6月8日(1999.6.8)		アメリカ合衆国 7 8 7 3 5 テキサス州
審査請求日	平成17年8月26日(2005.8.26)		オースティン ウィリアム キャノン
(31) 優先権主張番号	08/924, 518		ドライブ ウェスト 6 5 0 1
(32) 優先日	平成9年9月5日(1997.9.5)	(74) 代理人	100083574
(33) 優先権主張国	米国 (US)		弁理士 池内 義明
		(74) 代理人	100116322
			弁理士 桑垣 衛
		(72) 発明者	ウィリアム・シー・モイヤー
			アメリカ合衆国テキサス州7 8 6 2 0、ド
			リッピング・スプリングス、パイアー・ブ
			ランチ・ロード 1 0 0 5

最終頁に続く

(54) 【発明の名称】 プロセッサをコプロセッサにインタフェースするための方法および装置

(57) 【特許請求の範囲】

【請求項 1】

コプロセッサバス(30)を介して結合されたコプロセッサ(14, 16)と協働するよう構成されたプロセッサ(12)を動作させる方法であって、前記コプロセッサバスはシステムバスから分離されておりかつ前記プロセッサは前記システムバスを使用することなく前記コプロセッサバスを介して前記コプロセッサと通信することが可能であり、前記プロセッサ(12)は前記コプロセッサ(14, 16)と、前記コプロセッサに転送されるべきアーギュメントの数を規定するカウントフィールドおよびコードフィールドを備えた少なくとも1つの命令(H_CALL)の実行において協働し、前記方法は、

前記命令を受ける段階、

前記コプロセッサバスにより第1のサイクルを介して、前記コプロセッサに前記カウントおよびコードフィールドを提供する段階、

もし前記カウントフィールドがゼロより大きなある値、n、を有していれば、前記コプロセッサに、前記コプロセッサバスによって第2のサイクルを介して、第1のオペランドを提供する段階、そして

前記命令を完了する段階、

を具備することを特徴とするプロセッサ(12)を動作させる方法。

【請求項 2】

コプロセッサバス(30)を介して結合されたコプロセッサ(14, 16)と協働するよう構成されたプロセッサ(12)を動作させる方法であって、前記コプロセッサバスは

10

20

システムバスから分離されておりかつ前記プロセッサは前記システムバスを使用することなく前記コプロセッサバスを介して前記コプロセッサと通信可能であり、前記プロセッサ(12)は前記コプロセッサ(14, 16)と、前記コプロセッサから受信すべきアーギュメントの数を規定するカウントフィールドおよびコードフィールドを備えた少なくとも1つの命令(H_R E T)の実行において協働し、前記方法は、

前記命令を受ける段階、

前記コプロセッサバスにより第1のサイクルを介して、前記コプロセッサに前記カウントおよびコードフィールドを提供する段階、

もし前記カウントフィールドがゼロより大きなある値、n、を有していれば、前記コプロセッサバスによって第2のサイクルを介して、前記コプロセッサから第1のオペランドを受ける段階、そして

前記命令を完了する段階、

を具備することを特徴とするプロセッサ(12)を動作させる方法。

【請求項3】

複数のレジスタを備えたレジスタファイルを有する、プロセッサによって、コプロセッサ通信バスを介して、前記レジスタファイルにおける全ての書き込み処理を放送する方法であって、

前記レジスタファイルに書き込まれるべきオペランドを受ける段階、

前記レジスタファイルにおける前記複数のレジスタの内の1つを選択する段階、そして前記レジスタファイルにかつ前記コプロセッサ通信バスを介して、

前記レジスタファイルに書き込まれるべき前記オペランド、

前記レジスタファイルにおける前記複数のレジスタの内の前記選択された1つを示す第1の制御信号、および

前記オペランドが前記複数のレジスタの内の前記選択された1つに書き込まれることを要求する第2の制御信号、を提供する段階、

を具備することを特徴とする方法。

【請求項4】

第1のプロセッサによって、コプロセッサ通信バスを介して、前記第1のプロセッサとは別個の第2のプロセッサにおける、複数のレジスタを備えた、レジスタファイルへの全ての書き込み処理を監視する方法であって、

前記第2のプロセッサから、前記コプロセッサ通信バスを介して、

前記レジスタファイルに書き込まれるべきオペランド、

前記レジスタファイルにおける前記複数のレジスタの内の選択された1つを示す第1の制御信号、および

前記オペランドが前記複数のレジスタの内の前記選択された1つに書き込まれることを要求する第2の制御信号、を受ける段階、そして

前記オペランド、前記第1の制御信号および前記第2の制御信号を受けたことに応じて所定の操作を行なう段階、

を具備することを特徴とする方法。

【請求項5】

プロセッサによって、コプロセッサバスを介して、実行のために前記プロセッサによって受けた命令のコプロセッサによる実行を開始する方法であって、

前記命令を受ける段階、

前記命令をデコードする段階、

少なくとも部分的に前記デコードと一致した時間に、前記コプロセッサに対し、

前記コプロセッサバスの第1の部分を通じて、前記命令の少なくとも所定の部分、および

前記命令が前記コプロセッサバスの第2の部分を通じて前記プロセッサによりデコ

10

20

30

40

50

ードされていることを示す第 1 の制御信号、
を提供する段階、

前記命令の実行が進行していることを示すために前記コプロセッサバスに第 2 の制御信号を提供する段階、そして

もし前記第 1 の制御信号が前記第 2 の制御信号が肯定される前に否定されれば、前記命令の処理を中断する段階、

を具備することを特徴とする方法。

【請求項 6】

プロセッサによって、実行のために前記プロセッサによって受けた命令のコプロセッサによる実行を開始する方法であって、

前記命令を受ける段階、

前記命令をデコードする段階、

少なくとも部分的に前記デコードと一致した時間に、前記コプロセッサに対し、

前記コプロセッサバスの第 1 の部分を介して、前記命令の少なくとも所定の部分、
および

前記命令が前記プロセッサによって前記コプロセッサバスの第 2 の部分を介してデコードされていることを示す第 1 の制御信号、

を提供する段階、そして

もし前記命令に先立ち、より前の命令の実行が例外を引き起こせば、前記第 1 の制御信号を否定しかつ前記命令の処理を中断する段階、

を具備することを特徴とする方法。

【請求項 7】

プロセッサによって、コプロセッサバスを介して、実行のために前記プロセッサによって受けた命令のコプロセッサによる実行を開始する方法であって、

前記命令を受ける段階、

前記命令をデコードする段階、

少なくとも部分的に前記デコードと一致する時間に、前記コプロセッサに対し、

前記コプロセッサバスの第 1 の部分を介して、前記命令の少なくとも所定の部分、
および

前記命令が前記コプロセッサバスの第 2 の部分を介して前記プロセッサによりデコードされていることを示す第 1 の制御信号、

を提供する段階、そして

もし前記命令が前記プロセッサ内の命令レジスタから放棄されれば、前記第 1 の制御信号を否定しかつ前記命令の処理を中断する段階、

を具備することを特徴とする方法。

【請求項 8】

プロセッサによって、コプロセッサバスを介して、実行のために前記プロセッサによって受けた命令のコプロセッサによる実行を開始する方法であって、

前記命令を受ける段階、

前記命令をデコードする段階、

少なくとも部分的に前記デコードと一致する時間に、前記コプロセッサに対し、

前記コプロセッサバスの第 1 の部分を介して、前記命令の少なくとも所定の部分、
および

前記命令が前記コプロセッサバスの第 2 の部分を介して前記プロセッサによりデコードされていることを示す第 1 の制御信号、

を提供する段階、そして

前記コプロセッサバスから第 2 の制御信号を受ける段階であって、該第 2 の制御信号は前記コプロセッサが前記コプロセッサ内の命令バッファを充填することを可能にする、段階、

を具備することを特徴とする方法。

10

20

30

40

50

【請求項 9】

プロセッサによって、コプロセッサバスを介して、実行のために前記プロセッサによって受けた命令のコプロセッサによる実行を開始する方法であって、

前記命令を受ける段階、

前記命令をデコードする段階、そして

少なくとも部分的に前記デコードと一致した時間に、前記コプロセッサに対し、

前記コプロセッサバスの第 1 の部分を介して、前記命令の少なくとも所定の部分、および

前記コプロセッサバスの第 2 の部分を介して前記プロセッサがスーパバイザモードで動作しているか否かを示す第 1 の制御信号、

を提供する段階、

を具備することを特徴とする方法。

10

【請求項 10】

プロセッサによって、コプロセッサバスを介して、実行のために前記プロセッサによって受けた命令のコプロセッサによる実行を開始する方法であって、

前記命令を受ける段階、

前記命令をデコードする段階、

少なくとも部分的に前記デコードと一致した時間に、前記コプロセッサに対し、

前記コプロセッサバスの第 1 の部分を介して、前記命令の少なくとも所定の部分、および

前記コプロセッサバスの第 2 の部分を介して前記命令が前記プロセッサによりデコードされていることを示す第 1 の制御信号、

を提供する段階、そして

前記コプロセッサから、前記命令の前記所定の部分が前記コプロセッサ内で例外を引き起こしたか否かを示す第 2 の制御信号を受ける段階、

を具備することを特徴とする方法。

20

【請求項 11】

コプロセッサによって、コプロセッサバスを介して前記コプロセッサに結合されたプロセッサにより受けた命令に応じて操作を行なう方法であって、

前記プロセッサから、前記コプロセッサバスの第 1 の部分を介して前記命令の少なくとも所定の部分および、前記コプロセッサバスの第 2 の部分を介して、前記命令が前記プロセッサによりデコードされていることを示す第 1 の制御信号を受ける段階、

前記命令の実行を開始する段階、そして

前記プロセッサに対し前記命令が例外を引き起こしたか否かを示す制御信号を提供する段階、

を具備することを特徴とする方法。

30

【発明の詳細な説明】**【0001】****【発明の属する技術分野】**

本発明は一般的にはプロセッサおよび少なくとも 1 つのコプロセッサ (coprocessor) を有するデータ処理システムに関し、かつより特定的には、該コプロセッサに対しプロセッサをインタフェースするための方法および装置に関する。

40

【0002】**【従来の技術】**

専用のかつ専門化されたハードウェア機能要素によって基礎 (ベースライン: baseline) アーキテクチャプロセッサの機能性を拡張する能力は規模変更可能な (scalable) および拡張性のあるアーキテクチャの重要な観点である。

【0003】

基礎アーキテクチャプロセッサの機能を拡張するための 1 つの好ましい方法はコプロセッサを使用することによるものである。これらはプロセッサの指令で動作する専用の通常は

50

単一目的のプロセッサである。コプロセッサの伝統的な用途の1つは浮動少数点能力をそのようなものを直接サポートしないアーキテクチャに対して選択的に提供するための数学的または数値演算コプロセッサとしてのものであった。そのような数学的コプロセッサのいくつかの例はインテル (Intel) 社の 8087 型および 80287 型のものである。コプロセッサのいくつかの他の可能性ある用途または形式は、乗算 - 累算器 (multiply - accumulators)、変調器 / 復調器 (モデム)、デジタル信号プロセッサ (DSP)、ピタゴラス計算機 (pythagorean calculators)、暗号プロセッサ、画像プロセッサ、およびベクトルプロセッサを含む。

【0004】

【発明が解決しようとする課題】

10

コプロセッサに対する2つの異なる手法がある。一方においては、デジタル・イクイップメント・コーポレーション (DEC) の PDP - 11 型ファミリのコンピュータのための浮動小数点ユニットはその主プロセッサに対して非常に密接に (tightly) 結合された。発生した1つの問題は、この密接な結合が主プロセッサがコプロセッサの動作についてかなりの程度まで知っていることを必要としたことである。これは新しいコプロセッサを集積または統合システムに付加することが主たる工学技術上の問題になるほど回路設計を複雑にする。

【0005】

別の構成はコプロセッサを主プロセッサに対してルーズに (loosely) 結合することであった。これは主プロセッサからコプロセッサの動作を引き離し (abstracting) かつ分離する利点を有し、かつ従って新しいコプロセッサを既存のプロセッサと共に集積または統合するのに必要な努力を実質的に軽減した。しかしながら、これは常にかかなりの犠牲をはらって行われた。性能の喪失がこの手法の1つの問題である。このルーズな結合から生じる形式の性能ヒットをとることに伴う1つの問題は、そのようなコプロセッサに頼ることに対する損益分岐点が相応じて増大することである。従って、コプロセッサに対するそうでなければ魅力的な数多くの応用はコスト効率がよくない。さらに、そのような手法はしばしば、すべての対応する付加的な回路およびチップ面積と共に、バスを使用することを必要とする。

20

【0006】

従って、たとえ非常に簡単な機能に頼ることでも有利になるようにインタフェースを使用することが十分高速となるよう十分密接に結合され、一方プロセッサのアーキテクチャがいずれかと与えられたコプロセッサのできるだけ多くの細部から分離できるようにインタフェースを引き離すようなコプロセッサインタフェースをもつことが重要である。この後者の一部はハードウェアに代えてソフトウェアで新しいコプロセッサのアプリケーションをあたえることができるようにするためインタフェースのプログラマが親しみ易くする (friendly) ことを含む。

30

【0007】

【課題を解決するための手段】

本発明の一態様では、通信バス (30) を介して結合されたコプロセッサ (14, 16) とカウントフィールドおよびコードフィールドを備えた少なくとも1つの命令 (H__CALL) の実行において協働するよう構成されたプロセッサ (12) における、前記命令を実行する方法が提供され、該方法は、前記命令を受ける段階、前記通信バスにより第1のサイクルを介して、前記コプロセッサに前記カウントおよびコードフィールドを提供する段階、もし前記カウントフィールドがゼロより大きなある値、n、を有していれば、前記コプロセッサに、前記通信バスによって第2のサイクルを介して、第1のオペランドを提供する段階、そして前記命令を完了する段階、を具備することを特徴とする。

40

【0008】

本発明の別の態様では、通信バス (30) を介して結合されたコプロセッサ (14, 16) と実効アドレス計算フィールドを備えた少なくとも1つの命令 (H__LD) の実行において協働するよう構成されたプロセッサ (12) における、前記命令を実行する方法が提

50

供され、該方法は、前記命令を受ける段階、前記通信バスによって第１のサイクルを介して、前記コプロセッサに前記実効アドレス計算フィールドを提供する段階、前記実効アドレス計算フィールドに従って実効アドレスを計算する段階、前記計算された実効アドレスに格納されたオペランドをフェッチする段階、前記通信バスによって第２のサイクルを介して、前記コプロセッサに前記フェッチされたオペランドを提供する段階、そして前記命令を完了する段階、を具備することを特徴とする。

【０００９】

本発明のさらに別の態様では、通信バス（３０）を介して結合されたコプロセッサ（１４，１６）と実効アドレス計算フィールドを備えた少なくとも１つの命令（Ｈ＿ＳＴ）の実行において協働するよう構成されたプロセッサ（１２）における、前記命令を実行する方法が提供され、該方法は、前記命令を受ける段階、前記通信バスによって第１のサイクルを介して、前記コプロセッサに前記実効アドレス計算フィールドを提供する段階、前記実効アドレス計算フィールドに従って実効アドレスを計算する段階、前記通信バスによって第２のサイクルを介して、前記コプロセッサからオペランドを受ける段階、前記計算された実効アドレスに前記受けたオペランドを格納する段階、そして前記命令を完了する段階、を具備することを特徴とする。

【００１０】

【発明の実施の形態】

本発明の特徴および利点は添付の図面と共に以下の詳細な説明を参照することによりさらに明瞭に理解され、図面においては同じまたは同様の参照数字は同じかつ対応する部分を示している。

【００１１】

以下の説明においては、数多くの特定の細部は特定のワードまたはバイト長などで説明され本発明の完全な理解を与えている。しかしながら、当業者には本発明はそのような特定の細部なしで実施できることが明らかであろう。他の場合には、本発明を不必要な細部によって不明確にすることのないよう回路がブロック図形式で示されている。たいていの場合、タイミングの考慮事項その他に関する詳細は本発明の完全な理解を得るのに必要でない限り省略されておりかつ関連する技術に習熟した当業者の技術の範囲内にある。

【００１２】

用語「バス（bus）」は、データ、アドレス、制御またはステータスのような、１つまたはそれ以上の種々の形式の情報を転送するために使用できる複数の信号または導体に言及している。用語「肯定する（assert）」および「否定する（negate）」は信号、ステータスピットまたは同様の装置をそれぞれその論理的に真のまたは論理的に偽の状態にすることに言及する場合に使用される。もし論理的に真の状態が論理レベル“１”であれば、論理的に偽の状態は論理レベル“０”である。また、もし論理的に真の状態が論理レベル“０”であれば、論理的に偽の状態は論理レベル“１”である。

【００１３】

図１は、データ処理システム１０の一実施形態を示すブロック図であり、該データ処理システム１０はプロセッサ１２、コプロセッサ１４、コプロセッサ１６、メモリ１８、他のモジュール２０および外部バスインタフェース２２を含みこれらはすべてバス２８によって双方向的に結合されている。本発明の別の実施形態は１個だけのコプロセッサ１４、２つのコプロセッサ１４および１６、またはさらに多くのコプロセッサ（図示せず）をもつものとして行うことができる。外部バスインタフェース２２は集積回路端子３５によって外部バス２６に双方向的に結合されている。メモリ２４は双方向的に外部バス２６に結合されている。プロセッサ１２は任意選択的に集積回路端子３１によってデータ処理システム１０の外部に結合することができる。コプロセッサ１４は任意選択的に集積回路端子３２によってデータ処理システム１０の外部に接続することができる。メモリ１８は任意選択的に集積回路端子３３を介してデータ処理システム１０の外部に接続することができる。他のモジュール２０も任意選択的に集積回路端子３４を介してデータ処理システム１０の外部に結合することができる。プロセッサ１２はコプロセッサインタフェース３０によって

コプロセッサ１４およびコプロセッサ１６の双方に双方向的に接続されている。

【００１４】

図２は、図１のプロセッサ１２の一部を示すブロック図である。一実施形態では、プロセッサ１２は制御回路４０、命令デコード回路４２、命令パイプ４４、レジスタ４６、演算論理ユニット（ＡＬＵ）４８、ラッチングマルチプレクサ（*latching multiplexer*: *MUX*）５０、ラッチングマルチプレクサ（*MUX*）５２、およびマルチプレクサ（*MUX*）５４を含む。本発明の一実施形態では、コプロセッサインタフェース３０は信号６０～７１を含む。クロック信号６０は制御回路４０によって発生される。コプロセッサオペレーション信号６１は制御回路４０によって発生されかつコプロセッサ１４および１６に与えられる。

10

【００１５】

スーパバイザモード信号６２は制御回路４０によって発生されかつコプロセッサ１４および１６に提供される。デコード信号６３は制御回路４０によって発生されかつコプロセッサ１４および１６に提供される。コプロセッサビジー信号（*busy signal*）６４はコプロセッサ１４またはコプロセッサ１６から制御回路４０によって受信される。実行信号（*Execute signal*）６５は制御回路４０によって発生されかつコプロセッサ１４および１６に提供される。例外信号（*Exception signal*）６６はコプロセッサ１４またはコプロセッサ１６から制御回路４０によって受信される。レジスタ書込み（*REGWR**）信号６７は制御回路４０によって発生されかつコプロセッサ１４および１６に提供される。レジスタ信号（*REG{4:0}*）６８は制御回路４０によって発生されかつコプロセッサ１４および１６に提供される。エラー信号（*H_ERR**）６９は制御回路４０によって発生されかつコプロセッサ１４および１６に提供される。データストロープ信号（*H_DS**）７０は制御回路４０によって発生されかつコプロセッサ１４および１６に提供される。データアクノレッジ（肯定応答）信号（*H_DA**）７１は制御回路４０によってコプロセッサ１４およびコプロセッサ１６から受信される。コプロセッサインタフェース７０の一部とも考えられる、ハードウェアデータポート信号（*HDP{31:0}*）７２はコプロセッサ１４および１６とプロセッサ１２内の内部回路との間で双方向的なものである。

20

【００１６】

本発明の一実施形態では、複数の信号がバス２８に対しまたはバス２８から提供されてメモリ１８および／またはメモリ２４に対しデータをロードしまたは格納（ストア）する。一実施形態では、これらの信号は制御回路４０によって発生されかつバス２８に提供される転送要求信号（*TREQ**）７３を含む。転送エラーアクノレッジ信号（*TEA**）７４はバス２８によって制御回路４０に提供される。転送アクノレッジ信号（*TA**）７５はバス２８によって制御回路４０に提供される。命令はバス２８から導体７６によって命令パイプ（*instruction pipe*）４４に提供される。データは導体７６によってMUX５４に提供される。

30

【００１７】

ドライブデータ信号７９はトライステートバッファ９５をイネーブルして導体８８および７６によってラッチングMUX５２からデータを提供できるようにする。アドレス選択信号７８はラッチングMUX５０が導体７７によってバス２８にアドレスを提供できるようにする。MUX５４への他の入力HDP信号（*HDP{31:0}*）７２によって提供される。MUX５４への他の入力ALU結果導体８６によって提供される。MUX５４の出力、結果信号（*result signal*）８３、はレジスタ４６にかつトライステートバッファ９６の入力に提供される。ドライブHDP信号８２はトライステートバッファ９６がHDP信号７２によって結果信号８３をドライブできるようにする。トライステートバッファ９６の出力はまたラッチングMUX５２の入力に結合されている。

40

【００１８】

本発明の別の実施形態はレジスタ４６に任意の数のレジスタを含めることができる。結果信号８３はラッチングMUX５０への入力として提供される。結果信号８３はMUX５４

50

によってレジスタ４６に提供される。結果選択信号（結果__選択：RESULT__SELECT）８１はMUX５４のどの入力の結果導体８３にドライブされるべきかを選択する。ソース選択信号（ソース__選択：SOURCE__SELECT）８０はラッチングMUX５２に提供されてどの信号が導体８８によってトライステートバッファ９５にドライブされるかを選択する。

【００１９】

制御回路４０は導体９１によって制御情報を提供しかつレジスタ４６からステータス情報を受信する。制御回路４０は導体９２によって制御信号を提供しかつ演算論理ユニット４８からステータス信号を受信する。制御回路４０は導体９３によって制御信号を提供しかつ命令パイプ４４および命令デコード回路４２からステータス信号を受信する。命令パイプ４４は導体８９によって命令デコード回路４２に命令を提供するよう結合されている。命令デコード回路４２は導体９０によって制御回路４０に対しデコードされた命令情報を提供する。レジスタ４６は導体８４によってソースオペランドを演算論理ユニット４８に提供する。レジスタ４６は導体８０、ラッチングMUX５２、トライステートバッファ９５および導体７６によってメモリ１８またはメモリ２４に格納されるべきデータを提供する。レジスタ４６は導体８４、ラッチングMUX５０およびアドレス導体７７によってメモリ１８またはメモリ２４にアドレス情報を提供する。レジスタ４６は導体８５によって演算論理ユニット４８に第２のソースオペランドを提供する。

【００２０】

図３は、コプロセッサ１４の一部の一実施形態を示すブロック図である。一実施形態では、コプロセッサ１４は制御回路１００、計算回路１０２および任意選択的な記憶回路１０４を含む。制御回路１００は信号６０～７２を含むコプロセッサインタフェース３０によってプロセッサ１２に双方向的に結合されている。本発明の一実施形態では、制御回路１００はデコード回路１０６を含み、該デコード回路１０６はプロセッサ１２からオペレーション信号６１およびデコード信号６３を受信する。制御回路１００は導体１０８によって制御情報を提供しかつ任意選択的な記憶回路１０４からステータス情報を受信する。制御回路１００は導体１０９によって制御情報を提供しかつ計算回路１０２からステータス情報を受信する。計算回路１０２および任意選択的な記憶回路１０４は導体１１０によって双方向的に結合されている。信号１１０の１つまたはそれ以上はバス２８または集積回路端子３２に対しまたはこれらから提供できる。制御回路１００は導体１１２によってバス２８または集積回路端子３２に対し情報を受信しあるいは情報を提供することができる。信号７２は計算回路１０２および任意選択的な記憶回路１０４に双方向的に結合されている。さらに、信号７２は双方向的にバス２８または集積回路端子３２に結合することができる。本発明の別の実施形態では、任意選択的な記憶回路１０４は実施しなくてもよい。任意選択的な記憶回路１０４が実施されない本発明の実施形態においては、それはレジスタ、任意の種類メモリ、ラッチまたはプログラマブル論理アレイ、その他を含む任意の種類記憶回路を使用して構成できる。本発明の別の実施形態では、計算回路１０２は任意の種類論理または計算機能を行うことができる。

【００２１】

本システムは特定の用途に関連する動作のために最適化された外部コプロセッサ１４（またはハードウェアアクセラレータ）によるタスク加速のためのサポートを提供する。これらの外部コプロセッサ１４、１６は人口または母集団（population）のカウントを行うためのコプロセッサ１４としての簡単なものでもよく、あるいはDSP加速コプロセッサ１４または高速乗算／累算操作が可能なコプロセッサ１４のようなより複雑な機能のものとすることもできる。

【００２２】

データは特定の構成に対して適切な１つまたはそれ以上のいくつかのメカニズムによってプロセッサ１２とコプロセッサ１４との間で転送される。これらはコプロセッサ１４への転送、およびコプロセッサ１４からの転送へと分割できる。

【００２３】

10

20

30

40

50

コプロセッサ 14 へのデータの転送のための 1 つのメカニズムはレジスタスヌーピングメカニズム (Register Snooping mechanism) であり、これは命令プリミティブ (instruction primitive) を含まないが、通常のプロセッサ 12 の動作の副産物である。これはコプロセッサ 14 が 1 つまたはそれ以上のプロセッサ 12 のレジスタへの更新を監視できるようにインタフェースにわたってプロセッサ 12 の汎用目的のレジスタ ("GPR") 46 に対する更新を反映することを含む。これはもしコプロセッサ 14 が内部レジスタまたは機能に対して GPR 46 を「オーバーレイ (overlays)」する場合に適切であろう。この場合、プロセッサ 12 からコプロセッサ 14 への明確な (explicit) 受け渡しは必要とされないであろう。

【 0024 】

10

命令プリミティブ (Instruction primitives) は外部コプロセッサ 14 , 16 およびプロセッサ 12 の間でオペランドおよび命令の明確な転送のためにベースプロセッサ 12 において提供される。命令およびデータ転送のレートに対する制御を可能にするためハンドシェイクメカニズムが提供される。

【 0025 】

コプロセッサ 14 の機能は構成に特有の (implementation specific) ユニットとなるよう設計され、従って与えられたユニットの正確な機能は、同じ命令マッピングが存在しても、異なる構成にわたり自由に変更できる。

【 0026 】

図 4 は、レジスタスヌーピング動作を示すタイミング図である。コプロセッサ 14 または外部モタへのパラメータの受け渡しの性能オーバーヘッドを避けるため、レジスタスヌーピングメカニズムが提供される。これはコプロセッサ 14 がプロセッサ 12 の汎用レジスタ 46 の 1 つまたはそれ以上のシャドウコピー (shadow copy) を実施できるようにする。この能力はプロセッサの GPR 46 の 1 つへ書き込まれる値を転送しかつどのレジスタ 46 が各々の GPR の更新に対して更新されるかを指示することによって実施される。ストロブ信号 REGWR * 67 が各々のレジスタの更新に対して肯定される。この値は 32 ビットの双方向データバス HDP [31 : 0] 72 にわたって転送され、かつ 5 ビットのリジスタナンババスが更新される (REG [4 : 0]) 68 の実際のプロセッサレジスタ 46 へのポインタを提供する。前記レジスタナンバは通常のファイル (normal file) におけるあるいは代替りのファイル (alternate file) におけるレジスタ 46 に言及または参照することができる。好ましい実施形態では、オルタネイトまたは代替りのファイルのレジスタは REG [4] = 1 によって示されかつノーマルまたは通常のファイルのレジスタは REG [4] = 0 によって示される。しかしながら、この発明はレジスタセットの実際の区分 (partitioning) には依存しないことに注意を要する。

20

30

【 0027 】

専用の 12 ビット命令バス (H__OP [11 : 0]) 61 は外部コプロセッサ 14 に対して発行されるコプロセッサインタフェース 30 のオペコード (opcode) を提供する。このバスはプロセッサのオペコードの下位 (low order) 12 ビットを反映する。上位 (high-order) 4 ビットはそれらは常に 0b0100 であるため反映されない。スーパーバイザモード指示子 (H-SUP) 62 もまた PSR (S) ビットの現在の状態を示すために提供され、プロセッサがスーパーバイザまたはユーザモードのいずれで動作しているかを示す。これはあるコプロセッサ機能をスーパーバイザモードに限定するのに有用なものとすることができる。プロセッサ 12 と外部コプロセッサ 14 , 16 の間の一組のハンドシェイク信号はコプロセッサインタフェース 30 の命令実行を調整する。

40

【 0028 】

プロセッサ 12 によって発生される制御信号はプロセッサ 12 の内部パイプライン構造の反映である。プロセッサパイプライン 44 は命令フェッチ、命令デコード 42、実行、および結果の書き戻し (write back) のステージから構成される。それは 1 つまたはそれ以上の命令レジスタ (IR) を含む。プロセッサ 12 はまた命令プリフェッチバッ

50

ファを含みデコードステージ 4 2 の前に命令のバッファリングを可能にする。命令は命令デコードレジスタ I R に入ることによりこのバッファから命令デコードステージ 4 2 へと進行する。

【 0 0 2 9 】

命令デコーダ 4 2 は I R から入力を受け、かつ I R に保持された値に基づき出力を発生する。これらのデコード 4 2 の出力は常に有効であるとは限らず、かつ例外条件または命令フローの変更により捨てられることがある。有効な場合でも、命令はそれらが命令パイプラインの実行ステージへ進行することができるまで I R に保持することができる。これは前の命令が実行を完了するまで（これは複数クロックを必要とするであろう）生じ得ないから、デコーダは I R が更新されるまで I R に含まれる値をデコードし続けることになる。

10

【 0 0 3 0 】

図 5 は、命令ハンドシェイクのための基本的な命令インタフェース動作を示すタイミング図である。命令デコードストロープ (H _ D E C *) 信号 6 3 はプロセッサ 1 2 によってコプロセッサインタフェース 3 0 のオペコードのデコードを示すために与えられる。この信号は、たとえ命令が実行されずに捨てられる場合でも、コプロセッサインタフェース 3 0 のオペコードが I R に存在する場合に肯定されることになる。前記 H _ D E C * 6 3 の出力は命令が実際に発行されまたは捨てられるまで同じ命令に対して複数クロックの間肯定された状態に留まることできる。

【 0 0 3 1 】

20

ビジー信号 (H _ B U S Y *) 6 4 は外部コプロセッサ 1 4 がコプロセッサインタフェース 3 0 の命令を受け入れることができるか否かを判定するためプロセッサ 1 2 によって監視され、かつ命令の発行が生じる場合に対し部分的に制御を行う。もし前記 H _ B U S Y * 6 4 信号が、 H _ D E C * 6 3 が肯定されている間に、否定されれば、命令実行はインタフェースによってストールされず、かつ H _ E X E C * 6 5 信号が命令実行が進行できるや否や肯定できるようになる。もし H _ B U S Y * 6 4 信号がプロセッサ 1 2 がコプロセッサインタフェース 3 0 のオペコードをデコードする場合に (H _ D E C * 6 3 の肯定によって示される) 肯定されれば、コプロセッサインタフェース 3 0 のオペコードの実行は強制的にストールされることになる。いったん H _ B U S Y * 6 4 信号が否定されると、プロセッサ 1 2 は H _ E X E C * 6 5 を肯定することにより命令を発行することができる。もしコプロセッサ 1 4 が命令をバッファリングできれば、 H _ B U S Y * 6 4 信号はバッファの充填を助けるために使用することができる。

30

【 0 0 3 2 】

図 6 は、 H _ B U S Y * 6 4 がコプロセッサインタフェース 3 0 の命令実行を制御するために使用される場合の命令インタフェース動作を示すタイミング図である。いったん何らかの内部インストール条件が解消されると、かつ H _ B U S Y * 6 4 信号が否定されると、プロセッサは H _ E X E C * 6 5 を肯定してコプロセッサインタフェース 3 0 の命令がパイプラインの実行ステージに入ったことを示す。外部コプロセッサ 1 4 は命令の実際の実行を制御するために H _ E X E C * 6 5 信号を監視すべきであり、その理由はプロセッサがある状況において実行する前にその命令を放棄できるからである。もし前の命令の実行が結果として例外 (e x c e p t i o n) が取られることになれば、 H _ E X E C * 6 5 信号は肯定されず、かつ H _ D E C * 6 3 出力は否定されることになる。同様のプロセスは I R における命令がプログラムフローの変化の結果として廃棄されれば生じ得る。

40

【 0 0 3 3 】

図 7 は、命令廃棄または命令放棄 (i n s t r u c t i o n d i s c a r d) を示すタイミング図である。もしある命令が廃棄されれば、 H _ D E C * 6 3 信号は他のコプロセッサインタフェース 3 0 のオペコードが H _ O P [1 1 : 0] 6 1 のバスに与えられる前に否定されることになる。

【 0 0 3 4 】

図 8 は、命令パイプラインストールの例を示すタイミング図である。 H _ D E C * 6 3 が

50

肯定されかつ $H_BUSY * 64$ が否定されてもプロセッサ 12 が $H_EXEC * 65$ の肯定を遅らせることができる状況がある。これは前の命令が完了するのを待機している間に生じる得る。

【0035】

図9は、ストールのないバック-バック (back-to-back) 実行の例を示すタイミング図である。バック-バックコプロセッサインタフェース 30 の命令に対しては、 $H_OP[11:0]61$ バスが新しい命令が IR に入る際に更新されても、否定されることなく肯定された状態に留まることができる。一般に、 $H_EXEC * 65$ の肯定は前のクロックに際してデコードされた命令の実行に対応する。

【0036】

図10は、内部パイプラインストールを有するバック-バック動作を示すタイミング図である。この場合、 $H_BUSY * 64$ は否定されるが、プロセッサは内部ストール条件が消えるまで第2のコプロセッサインタフェース 30 の命令に対して $H_EXEC * 65$ を肯定しない。

【0037】

図11は、 $H_BUSY * 64$ ストールを有するバック-バックコプロセッサインタフェース 30 の命令を示すタイミング図である。この例では、外部コプロセッサ 14 はビジーであり、かつ直ちに第2の命令を受け入れることはできない。 $H_BUSY * 64$ は第2の命令がプロセッサ 12 によって発行されることを防止するため肯定する。いったんコプロセッサ 14 が自由になると、 $H_BUSY * 64$ が否定され、かつ次のコプロセッサインタフェース 30 の命令が実行ステージへと進む。

【0038】

コプロセッサインタフェース 30 のオペコードのデコードに関連する例外は外部コプロセッサ 14 によって $H_EXCP * 66$ 信号によって通知することができる。プロセッサ 12 へのこの入力 $H_DEC * 63$ が肯定されかつ $H_BUSY * 64$ が否定されるクロックサイクルの間にサンプルされ、かつもしコプロセッサインタフェース 30 のオペコードが前に述べたように廃棄されなければハードウェアコプロセッサ 14 の例外に対して例外処理を生じる結果となる。この例外処理の詳細は以下に説明する。

【0039】

図12は、コプロセッサインタフェース 30 のオペコードのデコードおよび試みられた実行に応じてコプロセッサ 14 により肯定される $H_EXCP * 66$ 信号の例を示すタイミング図である。 $H_EXCP * 66$ 信号は $H_DEC * 63$ が肯定されかつ $H_BUSY * 64$ が否定されるクロックの間にプロセッサ 12 によってサンプルされる。 $H_EXEC * 65$ 信号は例外がインタフェースによって通知されるか否かにかかわらず肯定され、この肯定は例外が取られる場合を命令廃棄の場合と区別する。

【0040】

前記例外は前のクロックサイクルにデコードされた命令に対応し、かつ実際の実行は行われるべきでないことに注意を要する。コプロセッサ 14 は違背 (offending) 命令を受け入れかつそれに対するプロセッサパイプラインの実行ステージが認識される前に例外を通知しなければならない。前記 $H_EXCP * 66$ 信号は $H_DEC * 63$ が否定されあるいは $H_BUSY * 64$ が肯定されるすべてのクロックサイクルに対して無視される。

【0041】

図13は、コプロセッサインタフェース 30 のオペコードのデコードおよび試みられた実行に応じてコプロセッサ 14 によって肯定される前記 $H_EXCP * 66$ 信号の例を示すタイミング図である。これを図14のタイミング図と比較すると、この例では、コプロセッサインタフェース 30 の命令は廃棄され、従って $H_EXEC * 65$ 信号は肯定されず、かつ $H_DEC * 63$ は否定される。

【0042】

図14は、 $H_BUSY * 64$ が肯定されて例外を生じる結果となるコプロセッサインタ

10

20

30

40

50

フェース 30 のオペコードの実行を遅らせる例を示すタイミング図である。

【 0 0 4 3 】

H__BUSY * 6 4 および H__EXCP * 6 6 信号はすべてのコプロセッサ 1 4 , 1 6 によって共有され、従ってそれらは調整された方法でドライブされなければならない。これらの信号は H__DEC * 6 3 が肯定されるクロックサイクルに際して H__OP [1 1 : 1 0] 6 1 に対応してコプロセッサ 1 4 , 1 6 により (ハイまたはローに、いずれか適切なものに) ドライブされるべきである。出力をクロックのローの部分の間にのみドライブすることにより、これらの信号は競合なしに複数のコプロセッサ 1 4 , 1 6 によって共有することができる。ラッチをプロセッサ 1 2 の内部に保持することはこの入力に対してそれを何らのユニットもそれをドライブしていない間にクロックのハイのフェーズに対して有効な状態に保持するために提供される。

10

【 0 0 4 4 】

コプロセッサインタフェース 30 の命令プリミティブのいくつかはまたプロセッサ 1 2 および外部コプロセッサ 1 4 の間でデータ項目またはデータアイテムの転送を行うことを含む。オペランドはコプロセッサインタフェース 30 にわたって実行されている特定のプリミティブの関数として転送することができる。プロセッサ 1 2 の G P R の 1 つまたはそれ以上を 3 2 ビットの双方向のデータバスにわたってコプロセッサ 1 4 からあるいはコプロセッサ 1 4 へ転送するための備えが設けられている。さらに、データシンク / ソース (s i n k / s o u r c e) がコプロセッサインタフェース 30 として単一データ項目をメモリ 1 8 にロードしあるいはメモリ 1 8 からストアするための備えも設けられている。プロセッサ 1 2 はパラメータを C L K 6 0 のハイ部分の間に H D P [3 1 : 0] 7 2 を介して外部コプロセッサ 1 4 に受け渡し、オペランドはクロックのローフェーズの間にプロセッサ 1 2 によってコプロセッサインタフェース 30 から受信されかつラッチされる。小さな期間のバスハンドオフを可能にするためドライブが行われる前にクロックがハイに遷移する際に遅延が与えられる。コプロセッサ 1 4 のインタフェースは立下りクロックエッジにおいて同じ小さな遅延を提供しなければならない。データ項目のハンドシェイクはデータストローブ (H__DS * 7 0) 出力、データアクノレッジ (H__DA * 7 1) 入力、およびデータエラー (H__ERR * 6 9) 出力信号によってサポートされる。

20

【 0 0 4 5 】

プロセッサ 1 2 はソフトウェアサブルーチンが呼び出され (c a l l e d) または戻る (r e t u r n e d) のとほぼ同じ方法でコプロセッサインタフェース 30 に対し呼出したまたはコール (c a l l) または戻り (r e t u r n) パラメータのリストを転送する能力を提供する。アーギュメント (a r g u m e n t s) のカウントは渡されるパラメータの数を制御するため H__CALL または H__RET プリミティブに示される。プロセッサ 1 2 のレジスタ R 4 の内容で始まるレジスタ値は H__CALL (H__RET) プリミティブの実行の一部として外部コプロセッサ 1 4 へ (から) 転送される。7 つまでのレジスタパラメータが受け渡しできる。この規約はソフトウェアサブルーチンコールの規約と同様のものである。

30

【 0 0 4 6 】

オペランド転送のハンドシェイクはデータストローブ (H__DS * 7 0) 出力およびデータアクノレッジ (H__DA * 7 1) 入力信号によって制御される。データストローブは転送の期間の間プロセッサ 1 2 によって肯定され、かつ転送はプロセッサ 1 2 のインタフェース動作とほぼ同様に、重複した方法で行われることになる。データアクノレッジ (H__DA * 7 1) はデータエレメントがコプロセッサ 1 4 によって受け入れられまたはドライブされたことを示すために使用される。

40

【 0 0 4 7 】

図 1 5 は、前記 H__CALL プリミティブに関連するレジスタ 4 6 の転送の例を示すタイミング図である。命令プリミティブは複数のプロセッサレジスタを転送するために提供されかつ該転送は理想的にはクロックごとに行うことができる。外部コプロセッサ 1 4 への転送のために、プロセッサは現在のアイテムのアクノレッジの前に (または同時に) 次の

50

オペランドを自動的にドライブし始める（もし必要であれば）。外部論理はデータの喪失がないことを保証するため1つのレベルのバッファリングが可能でなくてはならない。この図はコプロセッサインタフェース30へのH__CALL転送のシーケンシングまたは順序付けを示しており、この場合2つのレジスタが転送されるべきものである。第2の転送は否定されたデータアクノレッジ（H__DA*）71により反復される。

【0048】

外部コプロセッサ14からプロセッサレジスタ46への転送のために、プロセッサ12はH__DS*70が肯定された後のクロックサイクルごとに外部コプロセッサ14から値を受け入れることができ、かつこれらの値はそれらが受信される際にレジスタファイル46へと書き込まれ、従ってバッファリングは必要とされない。

10

【0049】

図16は、前記H__RETプリミティブに関連するレジスタ46の転送の例を示すタイミング図である。この例では、2つのレジスタ46の値が転送される。コプロセッサ14はH__EXEC*65信号の肯定に続くクロックで始まりデータを駆動することができ、それはこれがH__DS*70が始めに肯定されるクロックであるためである。H__DS*70の出力はCLK60の立上がりエッジで遷移し、一方H__DA*71の入力はCLK60のローフェーズの間にサンプルされる。

【0050】

プロセッサ12はH__LDまたはH__ST命令プリミティブによってコプロセッサインタフェース30へあるいはから単一命令オペランドを転送する能力を提供する。

20

【0051】

H__LDプリミティブはメモリ18からコプロセッサ14へデータを転送するために使用される。コプロセッサ14へのオペランド転送のハンドシェイクはデータストローブ（H__DS*）70信号によって制御される。データストローブはプロセッサ12によって肯定されて有効なオペランドがHDP[31:0]72バス上に与えられたことを示す。データアクノレッジ（H__DA*）71入力はこの転送に対しては無視される。

【0052】

図17は、コプロセッサインタフェース30へのH__LD転送のシーケンシングを示すタイミング図である。この場合、1つのノーウエイト状態（no-wait state）のメモリ18のアクセスがある。nのウエイト状態を有するメモリ18のアクセスに対しては、オペランドおよびH__DS*70はnクロック後にドライブされる。もしベースレジスタ46をロードの実効アドレスによって更新するためのオプションが選択されれば、更新値はそれが計算された後の最初のクロック（H__EXEC*65の肯定に続くクロック）でHDP[31:0]72においてドライブされる。

30

【0053】

図18は、メモリ18のアクセスが結果としてアクセス例外になる場合のプロトコルを示すタイミング図である。そのような場合、H__ERR*69信号は外部コプロセッサ14へと肯定し戻される。

【0054】

H__STプリミティブはデータをコプロセッサ14からメモリ18に転送するために使用できる。もし前記ストアの実効アドレスによってベースレジスタ46を更新するオプションが選択されれば、該更新値はそれが計算された後の最初のクロック（H__EXEC*65の肯定に続くクロック）でHDP[31:0]72においてドライブされる。

40

【0055】

図19は、H__STプリミティブに関連する転送の例を示すタイミング図である。H__STプリミティブに関連するハンドシェイクは2つの部分からなり、すなわちストアのためのデータを提供しなければならない、コプロセッサ14からの初期ハンドシェイク、およびいったんメモリ18へのストアが完了した場合のプロセッサ12からの完了ハンドシェイクである。

【0056】

50

前記初期ハンドシェイクはコプロセッサ 14 がストアデータをプロセッサ 12 へとドライブしたことを通知するためプロセッサ 12 への $H_DA * 71$ 入力を使用する。 $H_DA * 71$ 信号はコプロセッサ 14 によってデータが $HDP[31:0]72$ へとドライブされるのと同じクロックで肯定される。ストアデータはハーフワードサイズのストアに対してはバスの下半分 (lower half) から取られ、上位 16 ビットはメモリ 18 に書き込まれない。前記 $H_DA * 71$ 信号は $H_EXEC * 65$ 信号が肯定されたクロックと共に始まりサンプルされることになる。メモリサイクルは $H_DA * 71$ が認識されるクロックの間要求され、かつストアデータは引き続くクロックに際してメモリ 18 にドライブされる。いったんストアが完了すると、プロセッサ 12 は $H_DS * 70$ 信号を肯定する。

10

【0057】

図 20 は遅延されたストアデータを備えた転送の例を示すタイミング図である。

【0058】

図 21 は、ストアがアクセスエラーの結果となる場合のプロトコル信号を示すタイミング図である。ここでは、 $H_ERR * 69$ 信号が肯定されることに注意を要する。もしハードウェアユニットが $H_EXEC * 65$ が肯定されるクロックで $H_EXCP * 66$ を肯定することにより命令を中止すれば、 $H_DA * 71$ 信号は肯定されるべきではない。

【0059】

図 22 ~ 26 はハードウェアアクセラレータ (またはコプロセッサ) 14 へのインタフェースのために命令セットの一部として提供される命令を示す。プロセッサ 12 が前記プリミティブのフィールドのいくつかを解釈し、他のものはコプロセッサ 14 のみによって解釈される。

20

【0060】

図 22 は、 H_CALL プリミティブに対する命令フォーマットを示す。この命令はコプロセッサ 14 によって実施される機能を「コール」するまたは「呼び出す」(call) ために使用される。パラダイム (paradigm) は、ハードウェアコンテキストにおけるものを除き、標準的なソフトウェアコール規約と同様である。 H_CALL プリミティブはプロセッサ 12 およびコプロセッサ 14 の双方によって解釈されて「コールパラメータ (call parameters)」またはアргументのリストをプロセッサ 12 から転送し、かつコプロセッサ 14 において特定の機能を開始する。

30

【0061】

命令ワードの UU および $CODE$ (コード) フィールドはプロセッサ 12 によって解釈されず、これらはコプロセッサ 14 の特定の機能を示すために使用される。 UU フィールドは特定のコプロセッサ 14, 16 を特定することができ、かつ $CODE$ フィールドは特定の動作または操作を特定することができる。 CNT フィールドはプロセッサ 12 およびコプロセッサ 14 の双方によって解釈され、かつコプロセッサ 14 に受け渡すためにレジスタアргументの数を特定する。

【0062】

アргументは汎用レジスタ 46 から $R4$ で始まり、かつ $R(4 + CNT - 1)$ へと続いて受け渡される。7 個までのパラメータまたはレジスタ 46 が単一の H_CALL の実施または発動において受け渡すことができる。

40

【0063】

H_CALL 命令はモジュール方式のモジュール発動 (modular module invocation) を実施するために使用できる。この種のインタフェースを使用することは長い間より高い信頼性およびより少ないバグを備えたソフトウェアシステムを生み出すために知られてきている。通常、機能パラメータは値 (value) によって最もよく受け渡される。これは副作用を大幅に低減する。多くの場合、C および C++ のようなブロック構造言語に対する近代のコンパイラは短いシーケンスのパラメータまたはアргументをレジスタ 46 における発動された機能またはサブルーチンに渡す。この技術は H_CALL 命令と共に実施することができる。コンパイラは 7 個までのパラメータま

50

たはアーギュメントをR 4で始まり引き続くレジスタ4 6にロードするよう構成でき、次にH__CALL命令を発生し、これは標準的なコンパイラ発生サブルーチンリンケージ命令を置き換える。

【0064】

図23は、H__RETプリミティブに対する命令フォーマットを示す。この命令はコプロセッサ14によって実施される機能「からの戻り(return from)」のために使用される。パラダイムは、ハードウェアコンテキストにおけるものを除き、プロセッサ12によって使用されるソフトウェアコール規約と同様である。H__RETプリミティブはプロセッサ12およびコプロセッサ14の双方によって解釈されて「戻りパラメータ(return parameters)」または値のリストをコプロセッサ14からプロセッサ12へと転送する。

10

【0065】

前記命令ワードのUUおよびCODE(コード)フィールドはプロセッサ12によって解釈されず、これらはコプロセッサ14の特定の機能を示すために使用される。UUフィールドはハードウェアユニットを特定することができ、かつCODEフィールドは戻りのためにコプロセッサ14における特定の動作または一組のレジスタ46を特定することができる。CNTフィールドはプロセッサ12およびコプロセッサ14の双方によって解釈され、かつコプロセッサ14からプロセッサ12へと渡すレジスタ46のアーギュメントの数を特定する。

【0066】

アーギュメントはR 4で始まりかつ $R(4 + CNT - 1)$ へとプロセッサ12の一般または汎用レジスタ46へ渡される。7個までのパラメータ(またはレジスタ内容)を戻すことができる。

20

【0067】

H__CALL命令と同様に、H__RET命令はまたモジュール方式のプログラミングを実施するために使用できる。構造プログラミング(structured programming)は機能戻り値が値によってコーリングルーチンへと最もよく渡し戻されることを要求する。これはしばしばコンパイラによって1つまたはそれ以上の戻り値をサブルーチンまたは機能戻りのためのレジスタに与えることによって効率的に行われる。しかしながら、伝統的な構造プログラミングはサブルーチンまたは機能が該サブルーチンまたは機能の発動の直後に戻ることを期待することに注目すべきである。コプロセッサ14の場合には、実行はしばしば発動プロセッサ12のものと非同期である。H__RET命令はプロセッサ12およびコプロセッサ14を再同期するために使用できる。従って、プロセッサ12は1つまたはそれ以上のレジスタ46をロードし、コプロセッサ14を1つまたはそれ以上のH__CALL命令によってアクティブまたは活性化し、関連のない命令を実行し、かつ次にコプロセッサ14と再同期し一方でH__RET命令を発行することによりコプロセッサ14から結果としての値(単数または複数)を受けることができる。

30

【0068】

図24は、H__EXECプリミティブに対する命令フォーマットを示す。この命令は機能を開始しあるいはアクセラレータによって実施される動作モードに入るために使用できる。H__EXEC命令はUUフィールドによって特定される特定のコプロセッサ14, 16における機能を制御するために使用できる。コードフィールドはプロセッサ12によって解釈されずむしろ指定されたコプロセッサ14, 16のために確保される。該命令ワードのUUおよびCODEフィールドはプロセッサ12によって解釈または変換処理されず、これらはコプロセッサ14の特定の機能を特定するために使用される。UUフィールドは特定のコプロセッサ14, 16を指定することができ、かつCODEフィールドは特定の動作を示すことができる。

40

【0069】

図25は、H__LD命令のための命令フォーマットを示す。この命令はメモリオペランドを汎用目的のレジスタ(GPR)46に一時的に格納することなくある値をメモリ18か

50

らコプロセッサ 14 に受け渡すために使用される。メモリオペランドはベースポインタおよびオフセットを使用してアドレスされる。

【0070】

H__LD命令はメモリ 18 へのある値のロードを行い、かつメモリオペランドをそれをレジスタ 46 に格納することなくコプロセッサ 14 に渡す。H__LD動作は3つのオプション、w - ワード、h - ハーフワードおよびu - 更新 (update) を有する。DISPはIMM2フィールドをロードのサイズでスケーリングし、かつゼロ拡張 (zero-extend) を行うことによって得られる。この値はレジスタRXの値に加えられかつ特定されたサイズのロードはこのアドレスから行われ、ロードの結果はハードウェアインタフェース 28 に渡される。ハーフワードのロードに対しては、フェッチされたデータは32ビットへとゼロ拡張される。もしuオプションが特定されれば、ロードの実効アドレスはそれが計算された後にレジスタRX 46 に入れられる。

10

【0071】

前記命令ワードのUUフィールドはプロセッサ 12 によって解釈されず、このフィールドは特定のコプロセッサ 14 , 16 を特定することができる。Szフィールドはオペランドのサイズ (ハーフワードまたはワードのみ) を特定する。DispフィールドはRbaseフィールドによって特定されるレジスタの内容に加えられるべき符号なしの (unsigned) オフセット値を特定してロードのための実効アドレスを形成する。Dispフィールドの値は転送されるべきオペランドのサイズによってスケーリングされる。Upフィールドは前記Rbaseレジスタ 46 がロードの実効アドレスによってそれが計算された後に更新されるべきか否かを特定する。このオプションまたは選択肢は「自動更新 (auto-update)」アドレッシングモードを可能にする。

20

【0072】

図 26 は、H__ST命令に対する命令フォーマットを示す。この命令はプロセッサ 12 のレジスタ 46 にメモリオペランドを一時的に記憶することなくある値をコプロセッサ 14 からメモリ 18 に渡すために使用される。前記メモリオペランドはベースポインタおよびオフセットを使用してアドレスされる。

【0073】

前記命令ワードのUUフィールドはプロセッサ 12 によって解釈されない。むしろ、このフィールドは特定のコプロセッサ 14 , 16 を特定することができる。Szフィールドはオペランドのサイズ (ハーフワードまたはワードのみ) を特定する。DispフィールドはRbaseフィールドによって特定されるレジスタ 46 の内容に加えられるべき符号なしのオフセット値を特定しストアのための実効アドレスを形成する。Dispフィールドの値は転送されるべきオペランドのサイズによってスケーリングまたは尺度変更される。Upフィールドは前記Rbaseレジスタが前記ストアの実効アドレスによってそれが計算された後に更新されるべきか否かを特定する。このオプションは「自動更新」アドレッシングモードを可能にする。

30

【0074】

H__ST命令はメモリ 18 へのコプロセッサ 14 からのオペランドのストアを、それをレジスタ 46 に格納することなく、達成する。H__ST動作は3つのオプション、w - ワード、h - ハーフワード、およびu - 更新を有する。DispはIMM2フィールドをストアのサイズによってスケーリングすることおよびゼロ拡張により得られる。この値はレジスタRXの値に加えられかつ特定されたサイズのストアはこのアドレスに対して行なわれ、該ストアのためのデータはハードウェアインタフェースから得られる。もし前記uオプションが特定されれば、ロードの実効アドレスがそれが計算された後にレジスタRXに入れられる。

40

【0075】

前記H__LD命令およびH__ST命令はデータがレジスタ 46 を通って導き移動されることなくオペランドをメモリ 18 からコプロセッサ 14 へかつコプロセッサ 14 からメモリ 18 へ移動するための効率的なメカニズムを提供する。オフセットおよびインデックスの備

50

えはアレイを効率的に歩進するためのメカニズムを提供する。したがって、これらの命令は特にループ内で有用である。両方の命令はプロセッサ 12 をコプロセッサ 14 とロードまたはストアされるそれぞれのオペランドに対して同期させることに注目すべきである。もしこれが必要ではないかあるいは好ましい場合には、あるいは、指定されたレジスタ（単数または複数）46 をメモリ 18 からのデータによって反復的にロードすることによりデータをコプロセッサ 14 に流し、かつコプロセッサインタフェースバス 30 もまたレジスタスヌーピングのために使用されるからコプロセッサ 14 がこれらのロードを検出するようにすることができる。

【0076】

当業者は本発明の精神から離れることなく修正および変更を行なうことができることを理解するであろう。したがって、この発明は添付の特許請求の範囲に含まれるすべてのそのような変更および修正を含むものとする。

【0077】

【発明の効果】

以上のように、本発明によれば、プロセッサおよび少なくとも 1 つのコプロセッサを有するデータ処理システムにおいて、プロセッサとコプロセッサとの間のインタフェースにおける従来技術の欠点を克服し高性能のデータ処理システムが実現できる。

【図面の簡単な説明】

【図 1】本発明に係わるデータ処理システムの一実施形態を示すブロック図である。

【図 2】図 1 のプロセッサの一部を示すブロック図である。

【図 3】図 1 のコプロセッサの一部の一実施形態を示すブロック図である。

【図 4】本発明に係わるレジスタスヌーピング動作を示すタイミング図である。

【図 5】命令ハンドシェイクのための基本的な命令インタフェース動作を示すタイミング図である。

【図 6】H__BUSY* 信号がコプロセッサインタフェース命令の実行を制御するために使用される場合の命令インタフェース動作を示すタイミング図である。

【図 7】命令放棄を示すタイミング図である

【図 8】命令パイプラインストールの例を示すタイミング図である。

【図 9】ストールのないバック - バック実行の例を示すタイミング図である。

【図 10】内部パイプラインストールを有するバック - バック動作を示すタイミング図である。

【図 11】H__BUSY* ストールを有するバック - バックコプロセッサインタフェース 30 の命令を示すタイミング図である。

【図 12】コプロセッサインタフェースオペコードのデコードおよび試みられた実行に応じてコプロセッサにより肯定される H__EXCP* 信号の例を示すタイミング図である。

【図 13】コプロセッサインタフェース命令が放棄された場合のコプロセッサインタフェースオペコードのデコードおよび試みられた実行に応じてコプロセッサにより肯定される H__EXCP* 信号の例を示すタイミング図である。

【図 14】コプロセッサインタフェースオペコードの実行を遅延するために H__BUSY* が肯定された例を示すタイミング図である。

【図 15】H__CALL プリミティブに関連するレジスタ転送の例を示すタイミング図である。

【図 16】H__RET プリミティブに関連するレジスタ転送の例を示すタイミング図である。

【図 17】コプロセッサインタフェースへの H__LD 転送のシーケンシングを示すタイミング図である。

【図 18】メモリアクセスが結果としてアクセス例外となるプロトコルを示すタイミング図である。

【図 19】H__ST プリミティブに関連する転送の例を示すタイミング図である。

【図 20】遅延されたストアデータを有する転送の例を示すタイミング図である。

10

20

30

40

50

【図 2 1】ストアが結果としてアクセスエラーとなる場合のプロトコル信号を示すタイミング図である。

【図 22】本発明に係る H_C A L L プリミティブのための命令フォーマットを示す説明図である。

【図 23】本発明に係る H _ R E T プリミティブのための命令フォーマットを示す説明図である。

【図 2 4】本発明に係る H _ E X E C プリミティブのための命令フォーマットを示す説明図である。

【図 25】本発明に係わる H __ L D 命令に対する命令フォーマットを示す説明図である。

【図 26】本発明に係わる H__S T 命令のための命令フォーマットを示す説明図である。

【符号の説明】

10 データ処理システム

1 2 プロセッサ

1 4 , 1 6 コプロセッサ

18 メモリ

20 他モジュール

2.2 外部バスインタフェース

24 メモリ

26 外部バス

28 バス

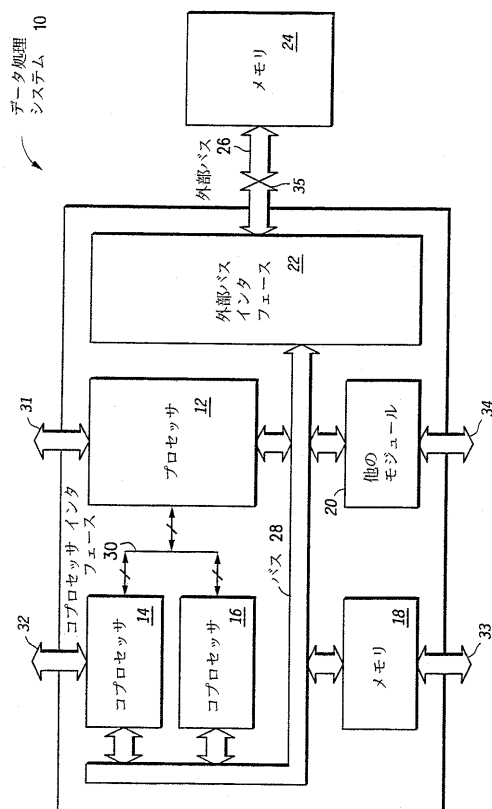
30 コプロセッサインタフェース

3 1 , 3 2 , 3 3 , 3 4 , 3 5 集積回路端子

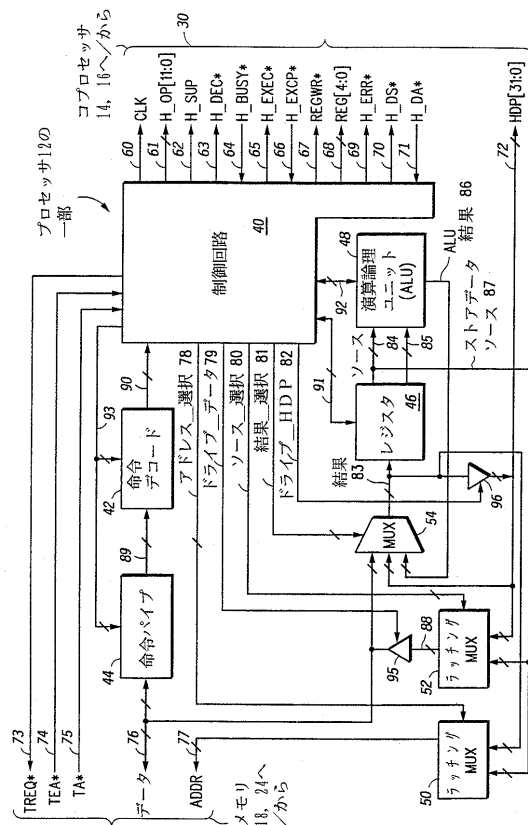
10

20

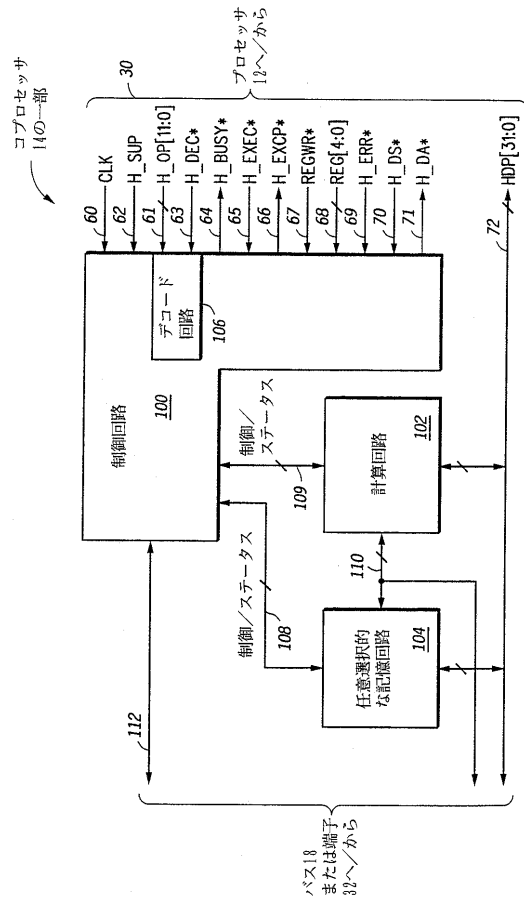
【圖 1】



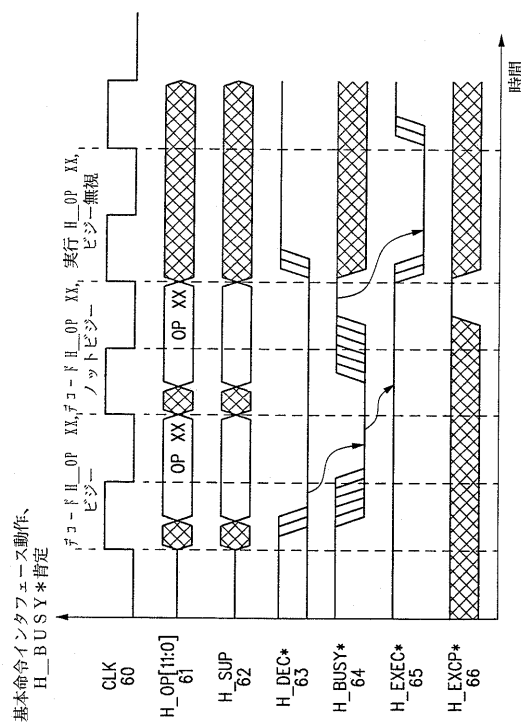
【 圖 2 】



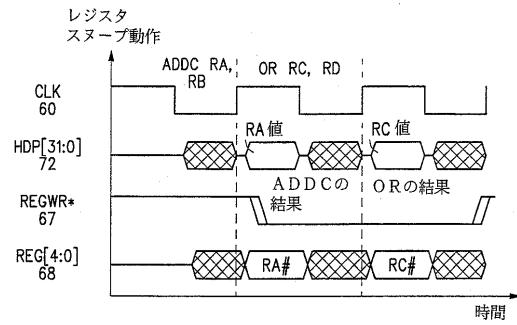
【図 3】



【図 6】

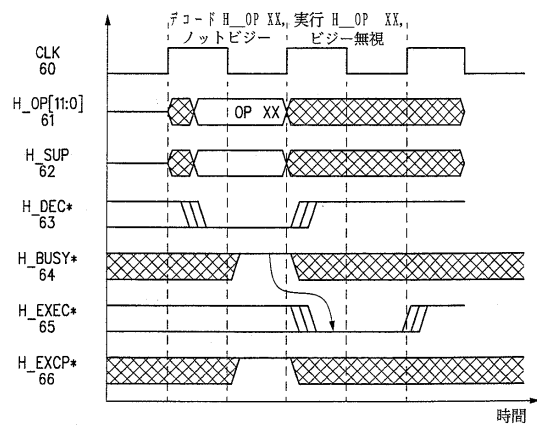


【図 4】

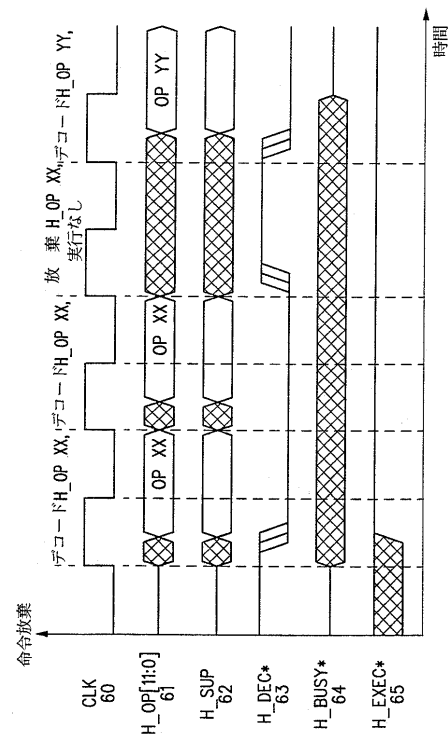


【図 5】

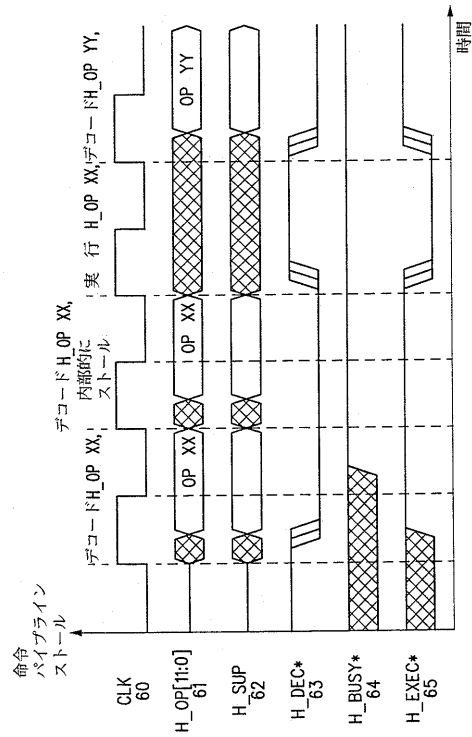
基本命令インタフェース動作、
H_BUSY*否定



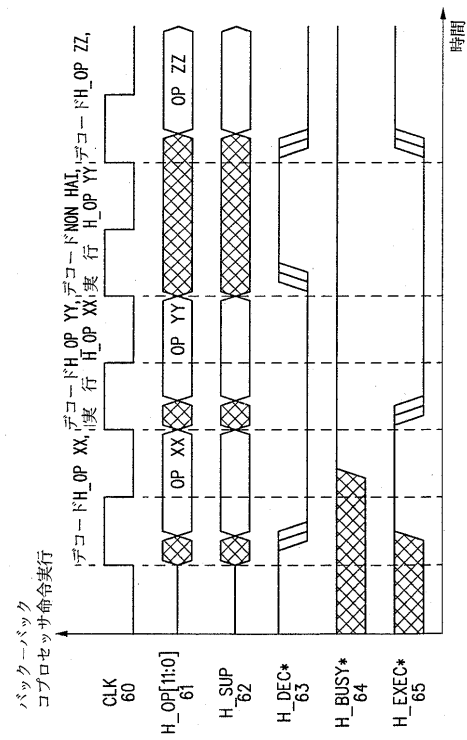
【図 7】



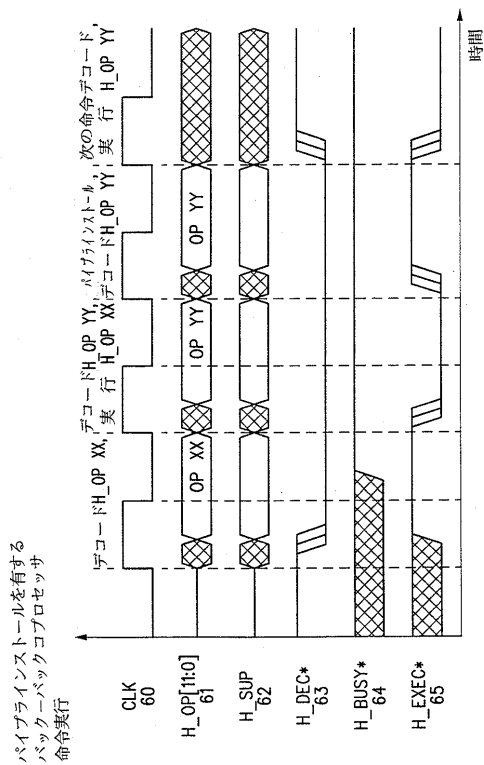
【図 8】



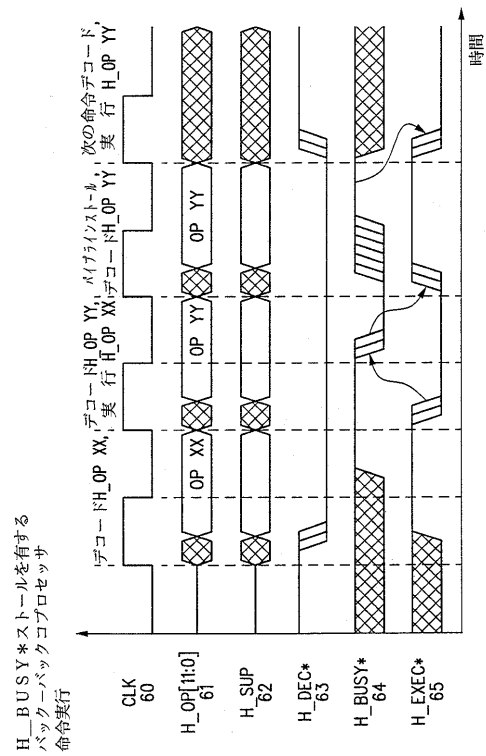
【図 9】



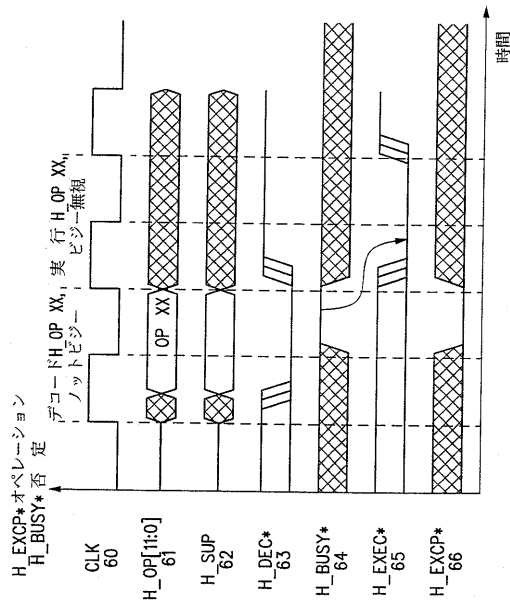
【図 10】



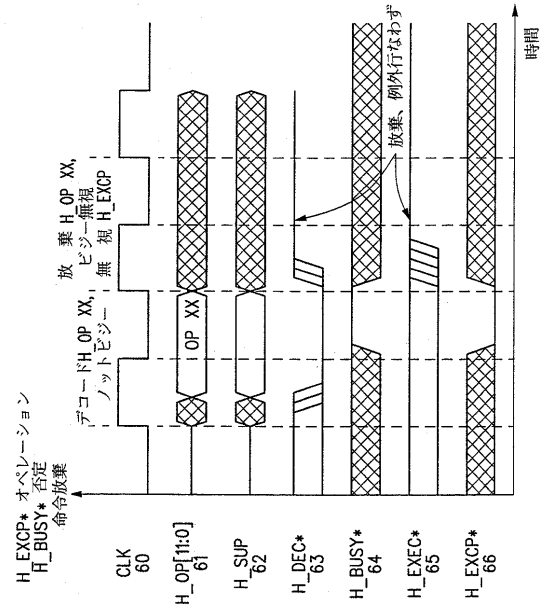
【図 11】



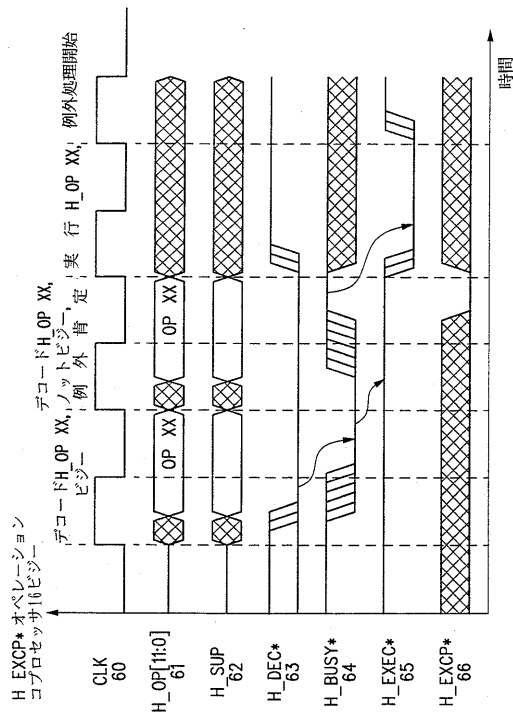
【図 12】



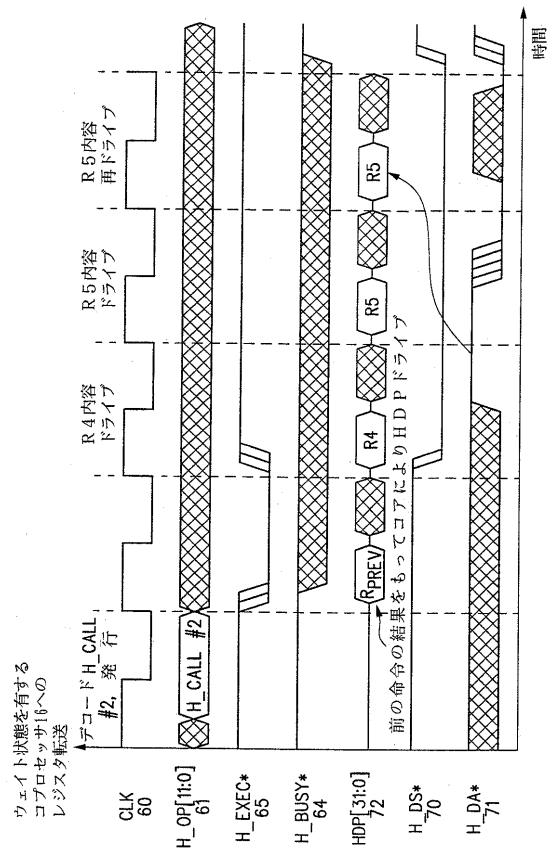
【図 13】



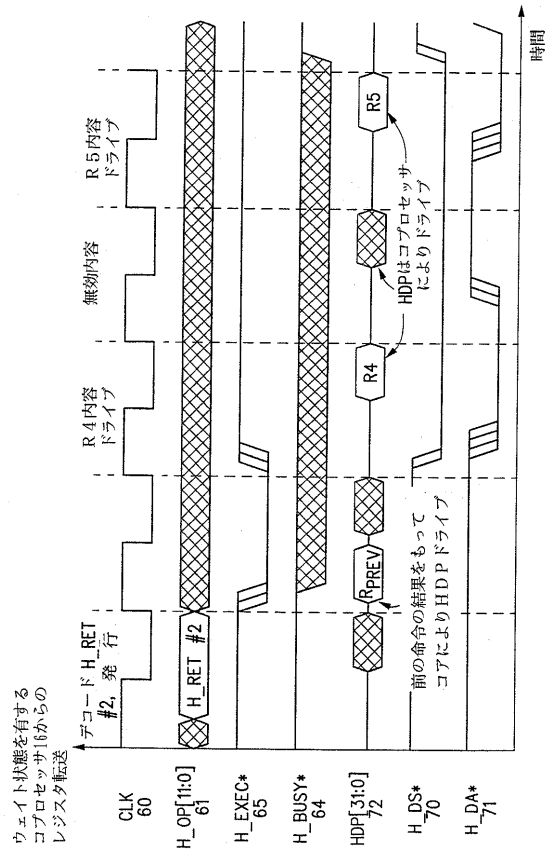
【図 14】



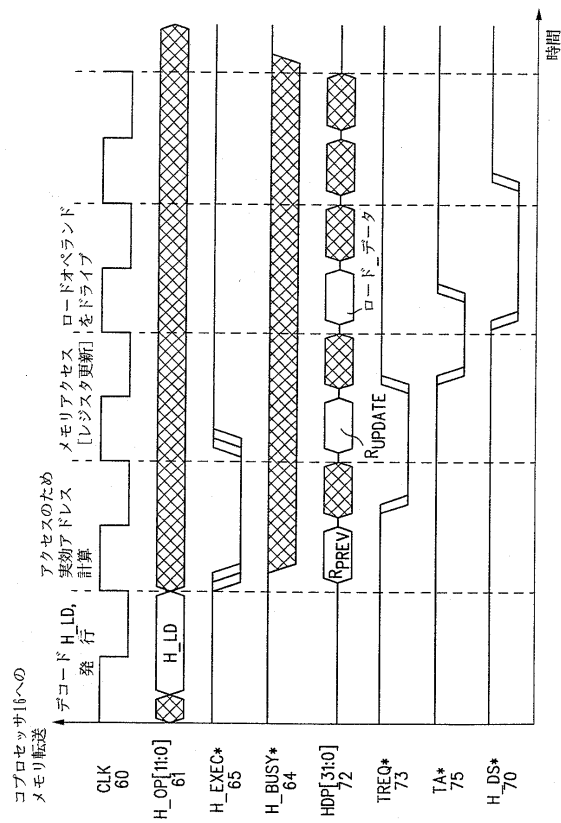
【図 15】



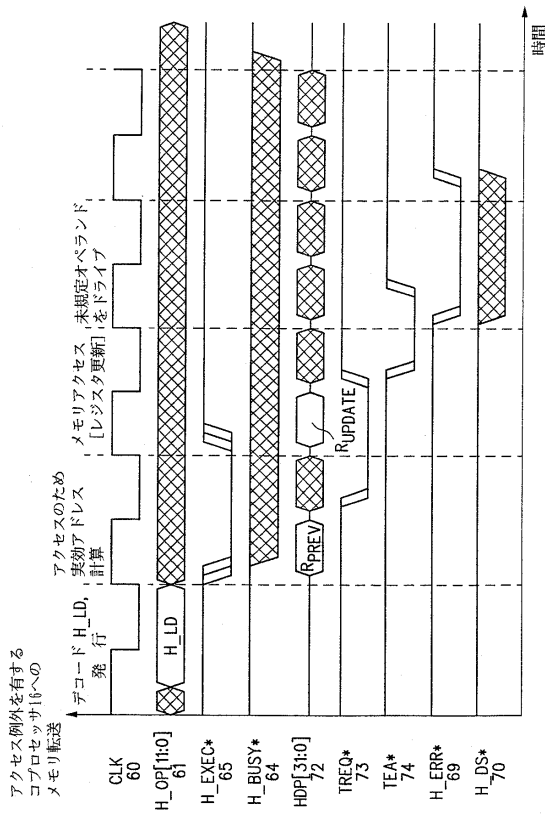
【図 16】



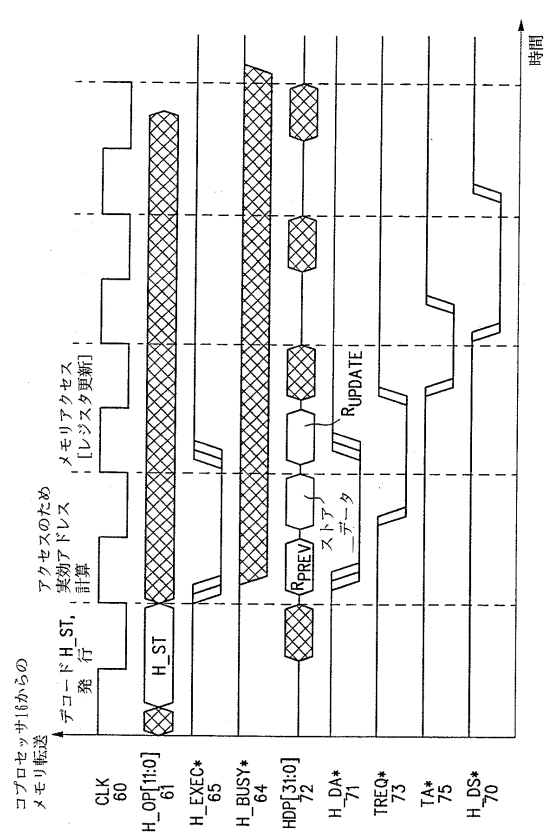
【図 17】



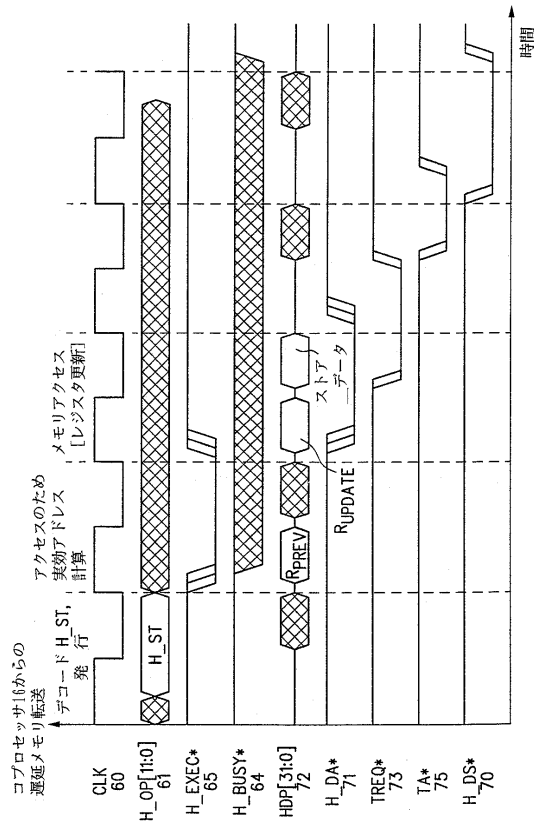
【図 18】



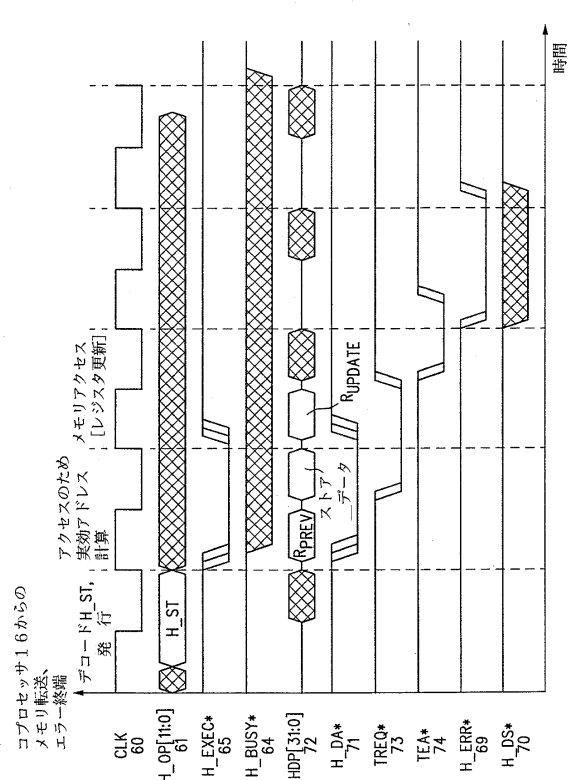
【図 19】



【 ㄨ 2 0 】



【 図 2 1 】



【 ㊦ 2 2 】

H_CALL ハードウェアアクセラレータ(コプロセッサ)コールプリミティブ															
動作: ハードウェアアクセラレータへパラメータを渡す															
アセンブラ シンタックス: H_CALL #UU, R4-RLAST, #CODE															
記述: H_CALLは1組のレジスタをベースとしたパラメータおよびコードをハードウェアブロック(コプロセッサ) #UUに渡す															
条件コード: 影響なし															
命令フォーマット:															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	UU		0	1	1	CNT			コード			

【 図 2 3 】

H_RET ハードウェアアクセラレータ（コプロセッサ）リターンプリミティブ

動作： ハードウェアアクセラレータからパラメータを渡す

アセンブラ

シンタクス： H_RET #UU, R4-RLAST, #CODE

記述： H_RET はコードをコプロセッサ #UU に渡しかつ CPU レジスタにロードされるべき 1 組の戻り (RETURN) パラメータを受ける

条件コード： 影響なし

命令フォーマット：

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	UU		0	1	0	CNT			コード			

命令フィールド：

U フィールド—ハードウェアブロック（コプロセッサ）を特定

00 — ブロック 0

01 — ブロック 1

10 — ブロック 2

11 — ブロック 3

CNT フィールド—R 4 で始まり、渡すべきレジスタの数を特定

000 — リザーブ、使用しない

001 — パス R4

010 — パス R4-R5

⋮

111 — パス R4-R10

【 図 2 4 】

H EXEC ハードウェアアクセラレータ（コプロセッサ）実行プリミティブ

動作： 実行コードをハードウェアアクセラレータへ渡す

アセンブラ

シNTAXス： H_EXEC #UU, #CODE

記述： H_EXECはコプロセッサ#UUにおける機能を制御するために使用される。コードフィールドはCPUによって解釈されない

条件コード： 影響なし

命令フォーマット：

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	UU	0	0	コード								

命令フィールド：

UUフィールド—ハードウェアブロック（コプロセッサ）を特定

00 — ブロック 0

01 — ブロック 1

10 — ブロック 2

11 — ブロック 3

コードフィールド—ハードウェアブロックに対するオペレーションコードを指定

【 図 2 5 】

H_LD	ハードウェアアクセラレータ（コプロセッサ）ロードプリミティブ
動作：	メモリからオペランドをロードしかつハードウェアアクセラレータへ渡す
アセンブラ シンタクス：	H_LD.[HW][U] #UU, (RX, DISP) H_LD.[U] #UU, (RX, DISP)
記述：	H_LDはメモリにある値のロードを行ない、かつメモリア ペラントをそれをGPRに格納することなくコプロセッサに渡す。H_ LD動作は3つのオプション、W-ワード、H-ハーフワードおよびU -更新、を有する。DISPはIMM2フィールドをロードのサイズお よびゼロ拡張によってスケリングすることにより得られる。この値は レジスタRXの値に加えられかつ指定されたサイズのロードはこのアド レスから行なわれ、該ロードの結果はハードウェアインフェースに渡さ れる。ハーフワードに対しては、フェッチされたデータは32ビ ットにゼロ拡張される。もしUオプションが指定されれば、ロードの実 効アドレスはそれが計算された後にレジスタRXに入れられる。
条件コード：影響なし	
命令フォーマット：	
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0	
0 1 0 0 UU 1 SZ 0 UP IMM2 RX	
命令フィールド：	
UUフィールド—ハードウェアブロック（コプロセッサ）を特定	
00 — ブロック 0	
01 — ブロック 1	
10 — ブロック 2	
11 — ブロック 3	
サイズ—ロードサイズを指定	
0 — ワード	
1 — ハーフワード	
UP — ベースレジスタが更新されるべきか否かを指定	
0 — 更新無し	
1 — 実効アドレスによってベースレジスタ更新	
IMM2 フィールド—2ビットスケリングされたイミディエイト値を指定	
レジスタX—ベースアドレスがスケリングされたイミディエイト フィールドに加えられるべきことを指定	

【圖 26】

H_ST ハードウェアアクセラレータ（コプロセッサ）ストアプリミティブ

動作：オペランドをハードウェアアクセラレータからメモリへストア

アセンブラ

シンタックス：H_ST.[HW][U] #UU, (RX, DISP)

記述： H_STはコプロセッサからオペランドをそれをGPRに格納することなくメモリへストアする。H_ST動作はW-ワード、H-ハーフワードおよびU-更新を有する。DISPは前記ストアのサイズおよびゼロ拡張によりIMM2フィールドをスケールリングすることにより得られる。この値はレジスタRXの値に加えられるがこのアドレスへの指定されたサイズのストアが行なわれ、ストアのためのデータはハードウェアインタフェースから得られる。もし前記Uオプションが指定されれば、ロードの実効アドレスはそれが計算された後にレジスタRXに入れられる。

条件コード：影響なし

命令フォーマット：

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	UU	1	SZ	1	UP	IMM2					RX	

命令フィールド：

UU フィールド—ハードウェアブロック（コプロセッサ）を特定

00 — ブロック 0

01 — ブロック 1

10 — ブロック 2

11 — ブロック 3

サイズ—ストアサイズを指定

0 — ワード

1 — ハーフワード

UP—ベースレジスタが更新されるべきか否かを指定

0 — 更新無し

1 — 実効アドレスによってベースレジスタ更新

IMM2 フィールド—2ビットスケールリングされたイミディエイト値を指定

レジスタX—ベースアドレスがスケールリングされたイミディエイトフィールドに加えらるべきことを指定

フロントページの続き

- (72)発明者 ジョン・アレンズ
アメリカ合衆国テキサス州78748、オースチン、シャッケルフォード・ドライブ 10709
- (72)発明者 ジェフリー・ダブリュ・スコット
アメリカ合衆国テキサス州78759、オースチン、ストーンレイク・ブルバード 9801
#1334

審査官 石川 正二

- (56)参考文献 特開平02-294830(JP,A)
特開平08-069377(JP,A)
特開平04-032954(JP,A)
特開平01-118954(JP,A)
特開平01-240932(JP,A)

- (58)調査した分野(Int.Cl., DB名)
G06F 15/16 620