

(19) 日本国特許庁 (JP)

(12) 特 許 公 報 (B2)

(11) 特許番号

特許第6340887号
(P6340887)

(45) 発行日 平成30年6月13日 (2018. 6. 13)

(24) 登録日 平成30年5月25日 (2018. 5. 25)

(51) Int. Cl.		F I			
G 0 6 F	9/38	(2006.01)	G 0 6 F	9/38	3 7 0 A
G 0 6 F	9/35	(2006.01)	G 0 6 F	9/34	3 1 0

請求項の数 10 (全 35 頁)

(21) 出願番号	特願2014-82660 (P2014-82660)	(73) 特許権者	000005223
(22) 出願日	平成26年4月14日 (2014. 4. 14)		富士通株式会社
(65) 公開番号	特開2015-203950 (P2015-203950A)		神奈川県川崎市中原区上小田中4丁目1番1号
(43) 公開日	平成27年11月16日 (2015. 11. 16)	(74) 代理人	100094525
審査請求日	平成29年1月10日 (2017. 1. 10)		弁理士 土井 健二
		(74) 代理人	100094514
			弁理士 林 恒徳
		(72) 発明者	田端 猛一
			神奈川県川崎市中原区上小田中4丁目1番1号 富士通株式会社内
		(72) 発明者	吉田 利雄
			神奈川県川崎市中原区上小田中4丁目1番1号 富士通株式会社内

最終頁に続く

(54) 【発明の名称】 演算処理装置及び演算処理装置の制御方法

(57) 【特許請求の範囲】

【請求項 1】

命令をデコードする命令デコーダと、
前記命令デコーダによりメモリアクセス命令のエントリを生成されるメモリアクセスエントリ部と、

前記メモリアクセスエントリ部から出力された前記メモリアクセス命令のエントリをメモリに対して実行するメモリアクセスパイプラインと、

前記命令デコーダにより複数のデータを1つの命令で処理するマルチデータ命令のエントリを生成されるマルチデータ命令エントリ部と、

複数の演算器と複数のマルチデータ命令用レジスタとを有し、前記マルチデータ命令エントリ部から出力された前記マルチデータ命令のエントリの処理を前記複数の演算器により並列に処理し、前記複数のマルチデータ命令用レジスタに演算結果を格納する演算パイプラインとを有し、

前記演算パイプラインは、前記複数のマルチデータ命令用レジスタに格納されている複数のメモリアドレスについて前記メモリにメモリアクセスするインダイレクトメモリアクセスを実行する前記マルチデータ命令のエントリの前記マルチデータ命令エントリ部からの出力にตอบสนองして、前記メモリアクセスパイプラインに前記インダイレクトメモリアクセスを実行する前記マルチデータ命令に対応する複数のメモリアクセス命令のメモリアクセス要求を生成し、前記複数の演算器が前記複数のマルチデータ命令用レジスタから取得した前記複数のメモリアドレスを前記メモリアクセスパイプラインに供給する演算処理装置

10

20

。

【請求項 2】

前記演算パイプラインは、前記メモリアクセスパイプラインの第 1 サイクルのステージに前記複数のメモリアクセス命令のメモリアクセス要求を生成し、前記メモリアクセスパイプラインの前記第 1 サイクルより後の第 2 サイクルのステージに前記複数のメモリアドレスを供給する

請求項 1 に記載された演算処理装置。

【請求項 3】

前記演算パイプラインは、前記メモリアクセスパイプラインに生成した複数のメモリアクセス命令のメモリアクセス要求のパイプライン転送タイミングにあわせて前記複数のメモリアドレスを供給する

10

請求項 2 に記載された演算処理装置。

【請求項 4】

さらに、前記メモリアクセスパイプラインに接続されたキャッシュユニットを有し、
前記演算パイプラインは、前記メモリアクセスパイプラインに生成する前記複数のメモリアクセス命令のメモリアクセス要求に前記キャッシュユニット内のアクセス先メモリアドレスを格納する複数のフェッチポートの識別情報を含める

請求項 1 に記載された演算処理装置。

【請求項 5】

前記演算パイプラインは、前記メモリアクセスパイプラインに、前記複数のメモリアクセス命令のメモリアクセス要求をシリアルに生成し、前記複数のメモリアドレスをシリアルに供給する

20

請求項 1、2、3 のいずれかに記載された演算処理装置。

【請求項 6】

前記メモリアクセスパイプラインが複数設けられ、
前記演算パイプラインは、前記複数のメモリアクセスパイプラインに、前記複数のメモリアクセス命令のメモリアクセス要求の少なくとも一部のメモリアクセス要求を並列に生成し、前記複数のメモリアドレスの少なくとも一部のアドレスを並列に供給する

請求項 1、2、3 のいずれかに記載された演算処理装置。

【請求項 7】

30

さらに、前記メモリアクセスパイプラインに接続されたキャッシュユニットを有し、
前記キャッシュユニットは、前記複数のメモリアクセス命令のメモリアクセス要求に
応答して、前記複数のマルチデータ命令用レジスタとの間でデータ転送を行う

請求項 1 に記載された演算処理装置。

【請求項 8】

前記演算パイプラインは、前記メモリアクセスエントリ部に抑止信号を出力して、前記メモリアクセスエントリ部に、前記メモリアクセスパイプラインに生成する前記複数のメモリアクセス命令のメモリアクセス要求と衝突するメモリアクセス命令のエントリの出力を抑止させる

請求項 1 に記載された演算処理装置。

40

【請求項 9】

前記演算パイプラインは、前記マルチデータ命令エントリ部に抑止信号を出力して、前記マルチデータ命令エントリ部に、前記メモリアクセスパイプラインにシリアルに生成する前記複数のメモリアクセス命令のメモリアクセス要求と衝突する前記インダイレクトメモリアクセスを実行する前記マルチデータ命令のエントリの出力を抑止させる

請求項 5 に記載された演算処理装置。

【請求項 10】

命令をデコードする命令デコーダと、
前記命令デコーダによりメモリアクセス命令のエントリを生成されるメモリアクセスエントリ部と、

50

前記メモリアクセスエントリ部から出力された前記メモリアクセス命令のエントリをメモリに対して実行するメモリアクセスパイプラインと、

前記命令デコーダにより複数のデータを1つの命令で処理するマルチデータ命令のエントリを生成されるマルチデータ命令エントリ部と、

複数の演算器と複数のマルチデータ命令用レジスタとを有し、前記マルチデータ命令エントリ部から出力された前記マルチデータ命令のエントリの処理を前記複数の演算器により並列に処理し、前記複数のマルチデータ命令用レジスタに演算結果を格納する演算パイプラインとを有する演算処理装置の制御方法において、

前記演算パイプラインが、前記複数のマルチデータ命令用レジスタに格納されている複数のメモリアドレスについて前記メモリにメモリアクセスするインダイレクトメモリアクセスを実行する前記マルチデータ命令のエントリの前記マルチデータ命令エントリ部からの出力に
 応答して、前記メモリアクセスパイプラインに前記インダイレクトメモリアクセスを実行する前記マルチデータ命令に対応する複数のメモリアクセス命令のメモリアクセス要求を生成し、

前記演算パイプラインが、前記複数の演算器が前記複数のマルチデータ命令用レジスタから取得した前記複数のメモリアドレスを前記メモリアクセスパイプラインに供給する演算処理装置の制御方法。

【発明の詳細な説明】

【技術分野】

【 0 0 0 1 】

本発明は、演算処理装置及び演算処理装置の制御方法に関する。

【背景技術】

【 0 0 0 2 】

演算処理装置であるCPUまたは演算処理部であるCPUコアの高速化手法として、スーパースカラやアウト・オブ・オーダー、SIMD (Single Instruction Multiple Data) などの技術が知られている。例えば、スーパースカラでは複数の命令を同時に処理し、アウト・オブ・オーダーではCPUコア内部の資源について処理が可能になったものを順不同で処理しプログラムの順序通り完了させる。

【 0 0 0 3 】

一方、SIMDは、1つの命令で複数のデータを並列に処理する。SIMD処理ではSIMDレジスタと呼ばれるレジスタを用いる。SIMDレジスタには、SIMD命令で並列に処理可能な個数と同数もしくはそれ以上の複数個のデータが1つの固まりとして格納される。SIMD命令で指定されるオペランドによりこの複数個のデータを1つの固まりとして格納するSIMDレジスタが選択され、SIMDレジスタ内の複数個のデータごとに命令の処理が実行される。この複数個のデータの各々を要素と呼び並列に処理するデータの要素数をSIMD幅と呼ぶ。SIMD命令においては、例えば命令で指定される第一ソースオペランド、第二ソースオペランド、第三ソースオペランドをSIMDレジスタから読み出し、SIMD演算を行いデスティネーションオペランドにデータを書き込む。このSIMD処理では、複数個のデータについて同じ命令の処理が並列に実行される。

【 0 0 0 4 】

また、SIMD命令以外の命令の処理においては、汎用レジスタと呼ばれるレジスタを用いる。汎用レジスタには、例えばメモリアクセスやSIMDで並列に処理を行わないデータが格納される。

【 0 0 0 5 】

従来は、メモリとSIMDレジスタ間のデータ転送は、汎用レジスタに格納されたアドレスを用いて行われる。メモリアクセス命令においては、オペランドアドレス生成器が、命令で指定される第一ソースオペランド、第二ソースオペランドを汎用レジスタから読み出し、メモリアクセスのためのアドレスの生成を行う。この読み出したアドレスを先頭アドレスとして用いて、メモリの連続するアドレス領域に存在するSIMD幅のデータを読み出し、デスティネーションオペランドに対するSIMDレジスタに読み出したデータを

ロードする，もしくはメモリの連続するアドレス領域にS I M Dレジスタから読み出したS I M D幅のデータをストアする。

【先行技術文献】

【特許文献】

【0006】

【特許文献1】特開2009-163442号公報

【特許文献2】特公平4-79026号公報

【特許文献3】特開2004-38750号公報

【特許文献4】特開2011-34450号公報

【発明の概要】

【発明が解決しようとする課題】

【0007】

しかしながら，並列に処理が可能なデータがメモリの連続するアドレス領域に存在しない場合は，上記のS I M Dロード命令，S I M Dストア命令を適用することはできない。

【0008】

さらに，従来のC P Uコアには，独立した複数のアドレスが格納されたS I M Dレジスタをソースオペランドに指定してメモリアクセスするインダイレクトメモリアクセス命令をS I M D命令で実行するための構成が設けられていない。したがって，従来のC P Uコアは，例えば，S M I Dレジスタの各々の要素を個別にアクセス可能な構成を利用して，S I M D幅に対する複数のメモリアクセス命令によりデータ転送を行う。

【0009】

または，ブロックロード命令のようにインダイレクトアクセス命令を複数の命令に分解して，S I M D幅に対する回数の命令を順次実行することも考えられる。しかし，上記のように複数の命令を実行する方法では，命令デコーダをS I M D幅と同じ回数使用しなければならず，また命令実行のためのリザーベーションステーションやコミットスタックエントリをS I M D幅と同数使用しなければならず，C P Uコアの多くの内部資源を使用する。さらに，命令デコーダを複数サイクル占有するため，依存関係のない後続命令をアウト・オブ・オーダーで実行することができず，アウト・オブ・オーダーで処理できるC P Uコアの構成を生かすことができない。

【0010】

そこで，本実施の形態の一つの目的は，レジスタに格納された複数個の独立したデータをアドレスとして使用し，メモリ領域の複数箇所にアクセスする命令を1つの命令で実行する演算処理装置及び演算処理装置の制御方法を提供することにある。

【課題を解決するための手段】

【0011】

本実施の形態の第1の側面は，

命令をデコードする命令デコーダと，

前記命令デコーダによりメモリアクセス命令のエントリを生成されるメモリアクセスエントリ部（R S A）と，

前記メモリアクセスエントリ部から出力された前記メモリアクセス命令のエントリをメモリに対して実行するメモリアクセスパイプライン（E A G A）と，

前記命令デコーダにより複数のデータを1つの命令で処理するマルチデータ命令のエントリを生成されるマルチデータ命令エントリ部（R S F）と，

複数の演算器と複数のマルチデータ命令用レジスタとを有し，前記マルチデータ命令エントリ部から出力された前記マルチデータ命令のエントリの処理を前記複数の演算器により並列に処理し，前記複数のマルチデータ命令用レジスタに演算結果を格納する演算パイプライン（F L A）とを有し，

前記演算パイプラインは，前記複数のマルチデータ命令用レジスタに格納されている複数のメモリアドレスについて前記メモリにメモリアクセスするマルチデータインダイレクトメモリアクセス命令のエントリの出力に応答して，前記メモリアクセスパイプラインに

10

20

30

40

50

前記マルチデータインダイレクトメモリアクセス命令に対応する複数のメモリアクセス要求を生成し、前記複数の演算器が前記複数のマルチデータ命令用レジスタから取得した前記複数のメモリアドレスを前記メモリアクセスパイプラインに供給する演算処理装置である。

【発明の効果】

【0012】

第1の側面によれば、マルチデータインダイレクトメモリアクセス命令を少ない資源で効率的に実行する。

【図面の簡単な説明】

【0013】

10

【図1】本実施の形態における演算処理装置が実現可能なインダイレクトメモリアクセス方式を説明する図である。

【図2】ブロックロード命令によるパイプライン処理の例を示す図である。

【図3】本実施の形態におけるSIMDインダイレクトメモリアクセス命令によるパイプライン処理を示す図である。

【図4】本実施の形態における演算処理装置を搭載した情報処理装置を示す図である。

【図5】CPUコア30の全体構成を示す図である。

【図6】本実施の形態のSIMDインダイレクトメモリアクセス（ロードまたはストア）命令を実行するCPUコアの構成を示す図である。

【図7】浮動小数点演算リザベーションステーションRSFにエントリとして格納されるフラグ構成を示す図である。

20

【図8】SIMDインダイレクトロード命令と通常のロード命令の処理を示すフローチャートである。

【図9】SIMDインダイレクトロード命令と通常のロード命令の処理を示すフローチャートである。

【図10】SIMDインダイレクトメモリアクセスの一つであるSIMDインダイレクトロード命令のパイプライン及びタイムチャートを示す図である。

【図11】本実施の形態のSIMDインダイレクトメモリアクセス（ロードまたはストア）命令を実行するCPUコアの構成を示す図である。

【図12】演算用インタフェース331とアドレス用インタフェース310の構成を示す図である。

30

【図13】SIMD幅が2のSIMDインダイレクトメモリアクセス命令の場合のパイプラインとアドレス用インタフェース310の入出力信号変化を示す図である。

【図14】SIMD幅が4のSIMDインダイレクトメモリアクセス命令の場合のパイプラインとアドレス用インタフェース回路310の入出力信号変化を示す図である。

【図15】図6の1つのメモリアクセスパイプラインEAGAを有する場合の演算用インタフェース331とアドレス用インタフェース310の構成を示す図である。

【図16】SIMD幅2の場合での後続するRSAから投入されるメモリアクセスとの衝突を示す図である。

【図17】SIMD幅4の場合での後続するRSAから投入されるメモリアクセスとの衝突を示す図である。

40

【図18】SIMD幅4の場合での後続するSIMDインダイレクトメモリアクセス命令のエントリの投入により生成されるメモリアクセス要求との衝突を示す図である。

【図19】インダイレクトメモリアクセス要求の衝突を回避する抑止信号を生成する演算用インタフェース333の構成を示す図である。

【図20】RSFとそのSIMDインダイレクトメモリアクセス命令のエントリの出力抑止回路を示す図である。

【図21】RSAとその通常のメモリアクセス命令のエントリの出力抑止回路を示す図である。

【図22】CSE内の完了待ち合わせ回路を示す図である。

50

【発明を実施するための形態】

【0014】

本実施の形態において1つの命令で複数のデータについて処理を行う命令をSIMD命令（またはマルチデータ命令）と称する。SIMD命令は、例えばSIMD幅の数のデータについてSIMD幅の数の演算器が並列に処理を行い、SIMD幅の数のレジスタを1つのレジスタ単位とするSIMDレジスタに処理結果を格納する。

【0015】

図1は、本実施の形態における演算処理装置が実現可能なインダイレクトメモリアクセス方式を説明する図である。図1は、SIMDレジスタ332__1に格納された複数の独立したデータをアドレスとして使用し、1つの命令でメモリ領域14の複数箇所にアクセスするインダイレクトメモリアクセス方式である。図1の例では、SIMD幅が4の例であり、このようなインダイレクトメモリアクセスを行う命令を、SIMDインダイレクトメモリアクセス命令と称する。

10

【0016】

図1(A)は、SIMDインダイレクトロード命令（またはSIMD間接ロード命令）の例であり、SIMDレジスタ332__1に格納された4つの独立したデータをアドレスとして利用し、メモリ14の4つのアドレスADD_0-ADD_3のデータDATA_0-DATA_3を読み出し、別のSIMDレジスタ332__2に書込む。このSIMDインダイレクトロード命令は、例えば次のように記述される。

```
load %f100 %f200
```

20

ここで、%f100はアドレスが格納されたSIMDレジスタ332__1のレジスタ番号であり、%f200はデータを書込むSIMDレジスタ332__2のレジスタ番号である。

【0017】

図1(B)は、SIMDインダイレクトストア命令（またはSIMD間接ストア命令）の例であり、SIMDレジスタ332__1に格納された4つの独立したデータをアドレスとして利用し、別のSIMDレジスタ332__3内のデータDATA_0-DATA_3をメモリ14の4つのアドレスADD_0-ADD_3の領域に書き込む。このSIMDインダイレクトストア命令は、例えば次のように記述される。

```
store %f100 %f300
```

ここで、%f100はアドレスが格納されたSIMDレジスタ332__1のレジスタ番号であり、%f300は書き込みデータが格納されたSIMDレジスタ332__3のレジスタ番号である。

30

【0018】

上記の場合、SIMDレジスタ332__1に4つの独立したアドレスを書き込む処理は、例えば4回のロード命令を実行することで行われる。または、メモリの連続するアドレスに4つの独立したアドレスを書き込んでおき、メモリの先頭アドレスをソースアドレスとするSIMDロード命令を実行することで行われる。

【0019】

図2は、ブロックロード命令によるパイプライン処理の例を示す図である。ここでのブロックロード命令は、例えばメモリの連続領域のデータを複数の汎用レジスタに書き込む命令である。ブロックロード命令は、命令デコーダでデコードされると、命令デコーダが複数のメモリアクセス命令を生成し、その複数のメモリアクセス命令が、順次命令デコーダでデコードされ、メモリアクセス用リザベーションステーションにエントリされ、メモリアクセスされる。つまり、マルチフロー方式である。

40

【0020】

したがって、4つのデータをメモリからロードする場合は、ブロックロード命令は4つのメモリアクセス命令に分割され、それぞれ命令デコードとリザベーションステーションへのエントリとメモリアクセスとが4回繰り返される。そのため、後続の演算命令は、4サイクルにわたりデコード待ち状態となる。

【0021】

50

このようなブロックロード命令の手法を利用して上記のS I M Dインダイレクトロード命令を実現しようとすると、同様に、命令デコードとリザベーションステーションへのエントリとインダイレクトロードの処理とを4回繰り返す必要があり、C P Uコア内の資源を4サイクルにわたり占有し、後続の演算命令の命令デコードが可能になるのはマルチフローの最後の命令のデコードが完了した後になる。これでは、後続命令が依存関係のない命令の場合に利用可能なアウト・オブ・オーダの利点を生かすことができない。

【0022】

[本実施の形態]

図3は、本実施の形態におけるS I M Dインダイレクトメモリアクセス命令によるパイプライン処理を示す図である。本実施の形態のS I M Dインダイレクトメモリアクセス命令では、1つのS I M Dインダイレクトメモリアクセス命令を命令デコーダがデコードし、命令デコーダが1つの命令をS I M Dリザベーションステーションにエントリし、4回のメモリアクセスを繰り返し実行する。したがって、命令デコーダは1サイクルで開放されるので、S I M Dインダイレクトメモリアクセス命令と依存関係のない後続の演算命令を、次のサイクルで命令デコードすることができる。したがって、アウト・オブ・オーダのメリットを生かすことができる。さらに、図3には示されていないが、1つのS I M Dインダイレクトメモリアクセス命令をS I M Dリザベーションステーションにエントリするので、リザベーションステーションに複数のエントリを使用する必要はなく、コミットスタックエントリのエントリも1つしか使用しないので、C P Uコア内の資源を効率的に使用する。

【0023】

図4は、本実施の形態における演算処理装置を搭載した情報処理装置を示す図である。コンピュータなどの情報処理装置10は、C P U/メモリボード12と、大容量の記憶装置であるハードディスク11とを有する。C P U/メモリボード12は、C P Uチップである演算処理装置20と、演算処理装置20と外部のハードディスク11などを接続するインタコネクト13と、DRAM等のメモリ14とを有する。

【0024】

演算処理装置20は、例えば、4つのC P Uコア(演算処理部)30A - 30Dと、4つのC P Uコアで共有される二次キャッシュ24と、入出力インタフェース26と、メインメモリ14へのアクセスを制御するメモリアクセスコントローラ28とを有する。

【0025】

図5は、C P Uコア30の全体構成を示す図である。C P Uコア30は、分岐命令の予測を行う分岐予測部302と、プログラムカウンタPCと分岐予測部302の予測に基づいて命令フェッチアドレスを生成する命令フェッチアドレス生成器301と、一次命令キャッシュ303と、フェッチされた命令をデコードする命令デコーダ305と、レジスタリネーミング部306と、メモリアクセス用リザベーションステーションR S A (Reservation Station for Address generate)と、整数演算用リザベーションステーションR S E (Reservation Station for Execute)と、浮動小数点S I M DリザベーションステーションR S F (Reservation Station for Floating)と、分岐用リザベーションステーションR S B R (Reservation Station for Branch)と、コミットスタックエントリC S E (Commit Stack Entry)とを有する。

【0026】

メモリアクセス用リザベーションステーションR S AのメモリアクセスパイプラインE A G Aは、アドレス用インタフェース310と、オペランドアドレス生成器311と、アドレス選択回路313と、一次データキャッシュ312とを有する。整数演算用リザベーションステーションR S Eの整数演算パイプラインE X Aは、演算用インタフェース333と、固定小数点演算器320と、固定小数点リネーミングレジスタ321と、固定小数点レジスタ322とを有する。

【0027】

また、浮動小数点S I M DリザベーションステーションR S Fの浮動小数点S I M D演

10

20

30

40

50

算パイプラインFLAは、演算用インタフェース333と、最大SIMD幅の数のSIMD演算器330と、浮動小数点SIMDリネーミングレジスタ331と、浮動小数点SIMDレジスタ332とを有する。さらに、CPUコア30は、2つのプログラムカウンタPC、NEXTPCを有する。また、CPUコア30は、演算器320、330が生成したデータを一時的に格納するストアバッファSTBを有する。

【0028】

SIMD演算器330のSIMD幅は、例えば2もしくは4を命令で指定可能である。浮動小数点SIMDレジスタは最大SIMD幅の4つ要素で構成されている。これらのレジスタの要素をそれぞれ、要素0、要素1、要素2、要素3と呼ぶ。浮動小数点SIMD幅2の演算を行う場合、SIMDレジスタの要素0と要素1を使用する。浮動小数点SIMD幅4の演算を行う場合、SIMDレジスタのすべての要素を使用する。

10

【0029】

メモリアクセスパイプラインEAGA、整数演算パイプラインEXA、浮動小数点SIMD演算パイプラインFLAは、それぞれ1つのパイプラインまたは2つ以上のパイプラインを有してもよく、それぞれ独立に命令を実行可能である。また、一次データキャッシュ312は、メモリアクセスパイプラインEAGAのパイプライン数が2の場合、それに合わせて、2つのポートを設け同時に最大2つのアドレスによりアクセスを行うことができるようにしてもよい。さらに、メモリアクセスパイプラインEAGAのパイプライン数を、最大SIMD幅と同じ4組にしてもよい。その場合は、一次データキャッシュ312も4つのポートを有して同時の最大4つのアドレスによりアクセスを行うことができるようにするのが望ましい。

20

【0030】

命令フェッチアドレス生成器301は、分岐予測部302またはプログラムカウンタPCからの命令アドレスを選択し、一次命令キャッシュ303に対して命令フェッチリクエストを発行する。一次命令キャッシュ303は、命令フェッチリクエストに応じた命令を命令バッファ304に格納する。命令バッファ304から命令デコーダ305に対しては、プログラムにより指定された順番通りに、すなわちインオーダで命令が供給され、命令デコーダ305は、命令バッファから供給された命令をインオーダでデコードする。

【0031】

命令デコーダ305は、デコードした命令の種類に応じて、各リザベーションステーションRSA、RSE、RSF及びRSBRのいずれかに、命令に対応する必要なエントリを作成する。これとともに命令デコーダ305はデコードされたすべての命令に対応するエントリをCSEに作成する。

30

【0032】

レジスタリネーミング部306は、リザベーションステーションRSA、RSE、RSFのいずれかにエントリが作成された場合に、命令に応じた処理で使用するレジスタのアドレスに、リネーミングレジスタ321、331のアドレスを割り当てる。

【0033】

リザベーションステーションRSA、RSE、RSFは、保持されたエントリのうち、処理に必要な資源（データ、演算器、レジスタ等）が準備されたものから順次パイプラインに出力し、後段のパイプラインEAGA、EXA、FLAに出力したエントリに対応する処理を実行させる。これにより、命令がアウト・オブ・オーダで実行される。

40

【0034】

浮動小数点演算用リザベーションステーションRSFには、例えば、SIMD演算命令に対応するエントリが格納される。1つのパイプラインFLAはSIMD幅の数のSIMD演算器330を有する。SIMD演算器330はRSFからのエントリに基づいて演算対象とするデータを選択し、SIMD幅の数のSIMD演算器で演算を並列に実行する。演算結果は浮動小数点・SIMDリネーミングレジスタ331に一時的に格納される。

【0035】

メモリアクセス用リザベーションステーションRSAには、命令デコーダ305により

50

SIMDインダイレクトメモリアクセス命令以外のメモリアクセス命令に対応するエンタリが生成され、格納される。そして、RSAは格納されている複数のエンタリのいずれかを選択してパイプラインに出力する。メモリアクセス命令のエンタリがパイプラインに出力されると、そのエンタリに対応するメモリアクセス要求がパイプラインの各ステージを順番に転送する。オペランドアドレス生成回路311は、RSAのエンタリのメモリアクセス要求に基づいて演算対象とするデータを選択し、アドレスを生成し、生成されたアドレスを用いてメモリアクセス要求を一次データキャッシュ312に入力する。一次データキャッシュ312は、メモリアクセス要求に対するメモリアクセスを実行する。

【0036】

コミットスタックエンタリCSEは、命令デコーダ305によりデコードされたすべての命令に対応するエンタリを保持し、各エンタリに対応する処理の実行状況を管理し、これらの命令をインオーダで完了させる。例えば、CSEは、次に完了させるべきエンタリに対応する処理の結果が、固定小数点リネーミングレジスタ321および浮動小数点SIMDリネーミングレジスタ331に格納されたと判定すると、格納されたデータを固定小数点レジスタ322または浮動小数点SIMDレジスタ332に出力させる。これにより、各リザベーションステーションでアウト・オブ・オーダに実行された命令が、インオーダで完了する。

【0037】

図5のCPUコア30では、固定小数点演算パイプラインEXAはSIMD構成になっていない。一方、浮動小数点演算パイプラインFLAはSIMD構成になっていて、最大SIMD幅の数のSIMD演算器330を有する。しかし、固定小数点演算パイプラインEXAもSIMD構成になっていてもよい。

【0038】

本実施の形態のCPUコア30は、浮動小数点SIMD演算パイプラインFLAの演算用インタフェース333の出力信号をメモリアクセスパイプラインEAGAのアドレス用インタフェース310に供給してメモリアクセス命令を生成させるためのバス334と、SIMD演算器330が取得したアドレスをアドレス選択回路313に供給するためのバス335とを有する。アドレス用インタフェース310は、演算用インタフェース333の出力信号に基づいて生成したSIMD幅の数のメモリアクセス命令をメモリアクセスパイプラインEAGAに出力する。また、アドレス選択回路313は、オペランドアドレス生成器331からのバスに代えて浮動小数点SIMD演算器330からのバス335を選択し、SIMD演算器330が浮動小数点SIMDレジスタ332や浮動小数点SIMDリネーミングレジスタ331から取得したアドレスを、前述のSIMD幅のメモリアクセス命令と共に一次データキャッシュ312へ供給する。

【0039】

[実施の形態のSIMDインダイレクトメモリアクセス命令を処理する構成と処理の概略]

図6は、本実施の形態のSIMDインダイレクトメモリアクセス(ロードまたはストア)命令を実行するCPUコアの構成を示す図である。図6には、後述するパイプラインの各サイクルが括弧付きで示されている。

【0040】

図6のCPUコア30は、メモリアクセス用リザベーションステーションRSA(またはメモリアクセスエンタリ部)がメモリアクセス命令のエンタリを出力する1つのメモリアクセスパイプラインEAGAを有する。また、浮動小数点SIMDリザベーションステーションRSF(またはマルチデータ命令エンタリ部)がSIMD命令のエンタリを出力するSIMD演算パイプラインFLAも1つ有する。そして、SIMD演算パイプラインFLAは、最大SIMD幅4と同じ数の浮動小数点SIMD演算器330を有する。

【0041】

本実施の形態のSIMDインダイレクトメモリアクセス命令の処理の概略は次のとおりである。命令デコーダ305は、SIMDインダイレクトメモリアクセス命令をデコード

10

20

30

40

50

して、そのエントリを浮動小数点S I M DリザベーションステーションR S Fに生成する。R S Fは、S I M Dインダイレクトメモリアクセス命令のエントリをS I M D演算パイプラインF L Aに出力すると、それに応答してS I M D演算パイプラインF L Aがバス334を介してS I M D幅に対応した数のメモリアクセス要求をメモリアクセスパイプラインE A G Aに生成する。具体的には、演算用インタフェース333が投入されたエントリのフラグ信号群を、バス334を介してアドレス用インタフェース310に供給し、アドレス用インタフェース310がそのフラグ信号群に基づいてメモリアクセスパイプラインE A G Aに複数のメモリアクセス命令のアクセス要求を順次生成する。または、演算用インタフェース333がそのフラグ信号に基づいて複数のメモリアクセス命令のアクセス要求を順次生成し、バス334を介してアドレス用インタフェース310に供給してパイプラインE A G Aに生成してもよい。

10

【0042】

また、S I M Dインダイレクトメモリアクセス命令のエントリの投入または出力に応答して、S I M D幅の数のS I M D演算器330は、S I M D幅の数のアドレスを浮動小数点S I M Dレジスタ332から並列に取得し、バス335を介してメモリアクセスパイプラインE A G Aに供給する。具体的には、S I M D演算器330は、取得した複数のアドレスをバス335を経由して順次アドレス選択回路313に供給する。アドレス選択回路313は、バス335から供給される複数のアドレスを、先に生成された複数のメモリアクセス命令のアクセス要求のタイミングに合わせて選択し、一次データキャッシュ312に出力する。

20

【0043】

具体的には、図6中の右上に示したとおり、アドレス選択回路313は、オペランドアドレス生成器311のアドレスフラグA__E A G A__A D Dとバス335のいずれかを選択するセレクト5を有する。そして、後述するようにS I M Dインダイレクトメモリ命令のエントリがS I M D演算パイプラインF L Aに出力されたことに応答してアドレス用インタフェース310が生成するフラグ信号B1__E A G A__I N D I R E C Tの「1」により、セレクト5はバス335側を選択し、バス335を経由して供給されるアドレスを選択し、アドレスフラグA__E A G A__A D Dとして一次データキャッシュ312に転送する。これにより、アドレス用インタフェース310がB1サイクルのステージで生成したメモリアクセス要求の転送タイミングに整合して、B1サイクルのステージより後のAサイクルのステージでバス335を介してアドレスが供給され、S I M Dインダイレクトメモリアクセス命令のアドレスを加えたメモリアクセス要求が一次データキャッシュ312に転送される。

30

【0044】

一次データキャッシュ312は、ロード命令の場合は一次キャッシュ312からまたはメモリ14から読み出した複数のデータを、浮動小数点S I M Dリネーミングレジスタ331に格納する。そして、コミットスタックエントリからの指令に応じて、読み出した複数のデータを浮動小数点S I M Dリネーミングレジスタ331から浮動小数点S I M Dレジスタ332に転送する。これらのレジスタ331、332は、S I M D幅の数のレジスタが一括してレジスタ番号で特定される。また、ストア命令の場合は、浮動小数点S I M Dレジスタ332に格納されている複数のデータを一次キャッシュ312またはメモリ14に順次書き込む。

40

【0045】

S I M Dインダイレクトメモリアクセス命令の処理の概略をより具体的に説明すると次の通りである。

【0046】

まず、命令デコーダ305はS I M Dインダイレクトメモリアクセス命令をデコードし、R S F及びC S Eにエントリを作成する。C S Eのエントリ番号(エントリされた命令識別情報)をI I Dと呼ぶ。演算やメモリアクセスの完了の際にC S EにこのI I Dと完了信号を通知することにより、C S Eは命令完了の判定を行う。エントリの作成と同時に

50

、命令デコーダ305は、一次データキャッシュ312が管理する資源であるフェッチポートFPをSIMD幅と同数の連続した個数確保する。フェッチポートFPは、一次データキャッシュがメモリアクセスを行う際に必要なメモリアドレスを記憶しておく資源であり、通常のメモリアクセス命令では1つのFPが確保される。SIMDインダイレクトメモリアクセス命令ではSIMD幅と同数のアドレスによりアクセスを行うため、複数のプリフェッチポートFPを使用する。

【0047】

図7は、浮動小数点演算リザベーションステーションRSFにエントリとして格納されるフラグ構成を示す図である。SIMDインダイレクトメモリアクセス命令を実行するため、インダイレクトフラグINDIRECTとフェッチポートフラグFPが追加されている。INDIRECTフラグはデコードした命令がSIMDインダイレクトメモリアクセス命令の場合に「1」となる。FPフラグはデコード時に確保した先頭のFP番号を示す。また、RSFには、これら以外にもSIMD演算器330に演算種の指示を行うOPCODE、命令のSIMD幅を識別する4SIMDフラグ（幅が2なら「0」、4なら「1」）、演算に使用するオペランドを示すR1__ADDRESS、CSEのエントリ番号を示すIIDなどを格納する。

【0048】

RSFは、SIMDインダイレクトメモリアクセス命令のエントリに必要な資源が準備され実行可能となると、浮動小数点SIMDパイプラインFLAの演算用インタフェース333にその命令のエントリを出力または投入する。

【0049】

演算用インタフェース333は、RSFから出力された命令がSIMDインダイレクトメモリアクセス命令のエントリである場合、そのエントリのインダイレクト命令、SIMD幅、IID、FP、FLAの命令が有効か否かを示すバリッドのフラグ信号群を、バス334を介して、アドレス用インタフェース310に転送する。アドレス用インタフェース310はこのフラグ信号群に基づいて、メモリアクセスパイプラインEAGAに複数のメモリアクセス命令のメモリアクセス要求をシリアルに生成する。

【0050】

上記の複数のメモリアクセス命令の生成と同時に、SIMD演算器330は、演算用インタフェース333からのフラグ信号に基づいて、SIMD幅の数のアドレスをSIMDレジスタ332から並列に読み出す。SIMDレジスタ332のレジスタ番号は、SIMDインダイレクトメモリアクセス命令のソースオペランドに示されている。SIMD演算器330は、SIMDレジスタ332からのアドレスの読み出しが完了すると、その複数のアドレスをアドレス選択回路313にバス335を介して転送する。そして、アドレス用インタフェース310が、アドレス用インタフェース310がパイプラインEAGAに順次生成した複数のメモリアクセス要求と、バス335を介して転送されてきた複数のアドレスとを、タイミングを整合させて、アドレス選択回路313に転送する。すなわち、アドレス選択回路313は、オペランドアドレス生成器331からのアドレスに代えて、SIMD演算器330から供給されてきたアドレスを選択し、複数のアドレスを一次データキャッシュ312にシリアルに転送する。一次データキャッシュ312は、複数のアドレスそれぞれについてデータの読み出しまたはSIMDレジスタ内のデータの書き込みを行う。

【0051】

一次データキャッシュ312は、データの読み出しを完了すると読み出したデータを浮動小数点SIMDリネーミングレジスタに格納するとともに、CSEに読み出しが完了したエントリ識別情報IIDと完了通知を送る。データの書き込みの場合は、一次データキャッシュ312は、単にCSEに書き込みが完了したエントリ識別情報IIDと完了通知を送る。CSEはエントリ識別情報IIDと完了通知によりSIMD幅すべての要素の読み出しまたは書き込みが完了するのを待ち、SIMDインダイレクトメモリアクセス命令を完了させる。

【 0 0 5 2 】

次に、演算用インタフェース 3 3 3 は、S I M D インダイレクト命令のエントリに基づいて生成されるメモリアクセス要求が、後続のメモリアクセス命令のアクセス要求と衝突することを防止するために、R S A と R S F に命令のエントリの出力を抑止する抑止信号を生成する。すなわち、第 1 に、演算用インタフェース 3 3 3 は、S I M D インダイレクトメモリアクセス命令のエントリにตอบสนองして、抑止信号 3 3 6 を R S A に出力し、R S A に、その S I M D インダイレクトメモリアクセス命令に基づいてアドレス用インタフェース 3 1 0 で生成されるメモリアクセス要求と衝突する後続のメモリアクセス命令のエントリの出力を抑止させる。第 2 に、演算用インタフェース 3 3 3 は、S I M D インダイレクトメモリアクセス命令のエントリにตอบสนองして、抑止信号 3 3 7 を R S F に出力し、R S F に、後続の S I M D インダイレクトメモリアクセス命令のエントリの出力を抑止させる。これにより、先行する S I M D インダイレクトメモリアクセス命令によりアドレス用インタフェース 3 1 0 に生成された複数サイクルにわたるメモリアクセス要求と、後続の S I M D インダイレクトメモリアクセス命令に基づいて生成されるメモリアクセス要求とが衝突することを防止する。

10

【 0 0 5 3 】

図 8、図 9 は、S I M D インダイレクトロード命令と通常のロード命令の処理を示すフローチャートである。まず、命令フェッチ (S1)、命令バッファに格納 (S2)、命令デコード (S3) が行われ、命令デコードの結果、通常のロード命令の場合 (S4のNO)、工程 S5 以下の処理が行われ、S I M D インダイレクトメモリアクセス命令の場合 (S4のYES)、工程 S21 以下の処理が行われる。

20

【 0 0 5 4 】

通常のロード命令の場合 (S4のNO)、命令デコード 3 0 5 は、フェッチポート F P を 1 つ確保し、R S A へロード命令のエントリを作成する (S5)。R S A は、ロード命令のエントリを投入する準備が完了したことを確認し (S6のYES)、先行する S I M D インダイレクトメモリアクセス命令に基づいて生成されるメモリアクセス命令と衝突していない場合 (S7のNO)、ロード命令のエントリをメモリアクセスパイプライン E A G A に出力または投入する。

【 0 0 5 5 】

メモリアクセスパイプライン E A G A では、オペランドアドレス生成器 3 1 1 が固定小数点レジスタ 3 2 2 などからデータを読み出し (S8)、オペランドアドレス生成器 3 1 1 がアドレスを生成し (S9)、アドレス選択回路 3 1 3 がオペランドアドレス生成器からのアドレスを選択する (S10)。そして、一次データキャッシュ 3 1 2 がそのアドレスを使用してデータを読み出す処理を実行する (S11)。一次データキャッシュ 3 1 2 が読み出したデータを浮動小数点 S I M D リネーミングレジスタに格納してその処理を完了すると (S12)、通常のロード命令の場合は、C S E がフェッチポート F P を 1 個開放し、S I M D リネーミングレジスタ 3 3 1 から S I M D レジスタ 3 3 2 に読み出しデータを転送する (S19)。

30

【 0 0 5 6 】

以上のように、通常のロード命令は、R S A にロード命令のエントリが生成され、メモリアクセスパイプライン E A G A のオペランドアドレス生成器 3 3 1 がアドレスの取得と生成を行い、一次データキャッシュへ 3 1 2 にロード要求を行う。

40

【 0 0 5 7 】

なお、連続するメモリアドレスに対する S I M D ロード命令のエントリが R S A に生成された場合は、オペランドアドレス生成器 3 1 1 がその先頭のアドレスを固定小数点レジスタ 3 2 2 などから読み出し、一次データキャッシュ 3 1 2 が連続する例えば 2 つのアドレスのデータを 2 つの S I M D リネーミングレジスタに格納する。ただし、この連続するメモリアドレスに対する S I M D ロード命令は、本実施の形態における複数の S I M D レジスタ内の独立した複数のアドレスに対する S I M D インダイレクトメモリアクセス命令とは異なる命令である。

50

【 0 0 5 8 】

次に，S I M D インダイレクトメモリアクセス命令の場合（S4のYES），命令デコーダはフェッチポートF PをS I M D 幅の個数確保し，R S FにS I M D 命令のエントリを生成する（S21）。R S Fは，エントリを投入する準備が完了したことを確認し（S22のYES），先行するS I M D インダイレクトメモリアクセス命令に基づいて生成されるメモリアクセス命令と衝突していない場合（S23のNO），S I M D インダイレクトメモリアクセス命令のエントリをS I M D 演算パイプラインF L Aに出力または投入する。

【 0 0 5 9 】

そして，S I M D 演算パイプラインF L AのS I M D 演算器330が，S I M D 幅の数のアドレスをS I M D レジスタ332から並列に読み出す（S23）。この読み出しには後述するとおり2サイクルを要する。そして，この読み出しとともに，S I M D 演算パイプラインF L Aの演算用インタフェース333がバス334を介してアドレス用インタフェース310にフラグ信号群を転送し，メモリアクセスパイプラインE A G Aにメモリアクセスのリクエスト0を生成させる（S24）。生成されたリクエストはメモリアクセスパイプラインE A G Aを転送する。さらに，S I M D 演算器330はS I M D レジスタ332から取得したS I M D 幅の数のアドレスを，バス335を介してアドレス選択回路313に供給し（S35），アドレス選択回路313は，リクエスト0に基づきS I M D 演算器からのアドレスを選択し（S26），リクエスト0にフェッチポートF Pを割り当てる（S27）。

10

【 0 0 6 0 】

上記の工程S23-S27を2回繰り返す。2回目はメモリアクセスのリクエスト1が生成される。さらに，S I M D 幅が4の場合（S28のYES），工程S23-S27と同じ処理工程S29-S32を2回繰り返す。これによりメモリアクセスのリクエスト2，3が生成される。

20

【 0 0 6 1 】

そして，一次データキャッシュ312は，フェッチポートF Pのアドレスを利用してデータを読み出す処理を実行開始する（S11）。C S Eは，S I M D 幅が2の場合は2回の一次データキャッシュの処理完了通知（S12,S14）に応答して，S I M D 幅が4の場合は4回の一次データキャッシュの処理完了通知（S12,S14,S16,S17）に応答して，S I M D 処理の完了を検出し，フェッチポートF PをS I M D 幅の個数開放し，S I M D リネーミングレジスタ331からS I M D レジスタ332に読み出しデータを転送する（S20）。

30

【 0 0 6 2 】

以上のように，S I M D インダイレクトロード命令の場合は，命令デコーダがS I M D インダイレクトロード命令を1回デコードし，命令デコーダがR S FにS I M D インダイレクトロード命令のエントリを1つ生成し，S I M D 演算パイプラインF L AがメモリアクセスパイプラインE A G AにS I M D 幅の数のメモリアクセスのリクエストを生成し，複数のS I M D 演算器に複数のアドレスをS I M D レジスタから並列に取得させ，S I M D 演算器が取得した複数のアドレスをメモリアクセスパイプラインE A G Aに転送して複数のメモリアクセスのリクエストに合体させ，メモリアクセスパイプラインE A G Aが一次データキャッシュへのロードリクエストを行う。S I M D インダイレクトストア命令の場合も，一次データキャッシュがメモリにストアすることを除いて上記のロード命令と同じ動作である。

40

【 0 0 6 3 】

次に，本実施の形態のS I M D インダイレクトメモリアクセス命令のパイプライン処理を説明する。まず，S I M D インダイレクトロード命令のパイプラインステージを以下に示す。図6に括弧付きで示したステージを参照することで以下のパイプラインステージが明らかになる。

D (D e c o d e) : 命令デコーダが命令をデコードする。

D T (D e c o d e T r a n s f e r) : Dサイクルの命令を転送し，R S Fに格納する。

P (P r i o r i t y) : R S FがS I M D 演算器へ投入する命令のエントリを決定し出

50

力（投入）する。

P T (P r i o r i t y T r a n s f e r) : P サイクルのエントリのフラグ信号群を、演算用インタフェースを介して転送し、S I M D 演算器 3 3 0 に投入する。

B 1 , B 2 (B u f f e r) : S I M D 演算器が演算に必要なデータをレジスタから入力する。例えば、データは浮動小数点 S I M D レジスタ 3 3 2 やリネーミングレジスタ 3 3 1 から取得される。この例では取得に 2 サイクルを要する。

X (e X e c u t i o n) : S I M D 演算器がメモリアクセスに必要なデータを読み出す。

A (A d d r e s s) : S I M D 演算器がメモリにアクセスするアドレスをアドレス選択回路 3 1 3 に転送する。

10

T (T a g) : 一次データキャッシュがアドレスに基づいてタグにアクセスする。

M (M a t c h) : 一次データキャッシュが読み出したキャッシュタグを比較する。

B (B u f f e r) : 一次データキャッシュから読み出したデータをバッファする。

R (R e s u l t) : 一次データキャッシュアクセスを完了する。

R T (R e s u l t) : R サイクルのデータを転送し、リネーミングレジスタへの書き込みを行い、C S E へ完了通知を行う。

C (C o m m i t) : すべての要素が完了したかどうかの命令完了の判定を行う。

W (W r i t e) : 完了した命令による各種レジスタの更新やリソースの解放を行う。このとき、浮動小数点 S I M D リネーミングレジスタ 3 3 1 から S I M D レジスタ 3 3 2 に読み出したデータを転送する。

20

【 0 0 6 4 】

S I M D インダイレクトロード命令以外の通常ロード命令のパイプラインステージを以下に示す。

D (D e c o d e) : 命令をデコードする。

D T (D e c o d e T r a n s f e r) : D サイクルの命令を転送し、R S A に命令のエントリを格納する。

P (P r i o r i t y) : リザベーションステーション R S A から実行ユニットへ投入する命令のエントリを決定し出力（投入）する。

B 1 , B 2 (B u f f e r) : オペランドアドレス生成器がロードアドレス生成に必要なデータを決定しレジスタから入力する。

30

A (A d d r e s s) : オペランドアドレス生成器がメモリにアクセスするアドレスを計算する。

T (T a g) : 一次データキャッシュが計算したアドレスに基づいてタグにアクセスする。

M (M a t c h) : 一次データキャッシュが読み出したキャッシュタグを比較する。

B (B u f f e r) : 一次データキャッシュから読み出したデータをバッファする。

R (R e s u l t) : 一次データキャッシュアクセスを完了する。

R T (R e s u l t) : R サイクルのデータを転送し、リネーミングレジスタへの書き込みを行い、C S E へ完了通知を行う。

C (C o m m i t) : 命令完了の判定を行う。

40

W (W r i t e) : 完了した命令による、各種レジスタの更新やリソースの解放を行う。このとき、リネーミングレジスタからレジスタに転送する。

【 0 0 6 5 】

図 1 0 は、S I M D インダイレクトメモリアクセスの一つである S I M D インダイレクトロード命令のパイプライン及びタイムチャートを示す図である。

【 0 0 6 6 】

R S F は、タイミング 3 の P サイクルで S I M D インダイレクトロード命令のエントリを S I M D 演算パイプライン F L A に投入する。そして、タイミング 5 の B 1 サイクルで、演算用インタフェース 3 3 3 が S I M D インダイレクトロード命令のフラグ信号を出力する。

50

【 0 0 6 7 】

S I M D 幅 2 の S I M D インダイレクトロード命令の場合は、S I M D 演算パイプライン F L A が、タイミング 6 において、アドレス用インタフェース 3 1 0 内のパイプライン E A G A にメモリアクセス用のリクエスト 0 を生成し、次のタイミング 7 において、リクエスト 1 を生成する。生成されたメモリアクセスのリクエストは、メモリアクセスパイプライン E A G A にて、S I M D インダイレクトロード命令以外のロード命令における B 1 サイクルとなる。生成したメモリアクセスのエントリ識別情報 I I D は S I M D 演算パイプライン F L A から送られたものを使用し、フェッチポート F P は S I M D 演算パイプライン F L A の F P の値とそれに 1 を加算した値を使用する。S I M D 演算パイプライン F L A の S I M D 演算器 3 3 0 が、タイミング 7 の X 1 サイクルで S I M D レジスタ（または S I M D リネーミングレジスタ）から読み出した複数のデータのうち要素 0 を、タイミング 8 の X 2 サイクルで S I M D レジスタ（または S I M D リネーミングレジスタ）から読み出した要素 1 をそれぞれシリアルにメモリアクセスパイプライン E A G A のアドレス選択回路 3 1 3 に転送する。タイミング 8 , 9 (A サイクル) で、アドレス選択回路 3 1 3 は、S I M D 演算パイプライン F L A から転送されてきた要素 0 と要素 1 のアドレスを選択し、一次データキャッシュ 3 1 2 に転送する。一次データキャッシュにアクセスしたデータすべてが存在した場合、一次データキャッシュ 3 1 2 は、タイミング 1 3 , 1 4 で読み出したデータを S I M D リネーミングレジスタ 3 3 1 に転送し、タイミング 1 4 ですべてのメモリアクセスの完了報告を行う。その結果、C S E は、命令完了の判定を行い、S I M D リネーミングレジスタ 3 3 1 のデータを S I M D レジスタ 3 3 2 に転送する。

10

20

【 0 0 6 8 】

また、S I M D 幅 4 の S I M D インダイレクトロード命令の場合は、S I M D 演算パイプライン F L A が、タイミング 6 , 7 , 8 , 9 において、アドレス用インタフェース内のパイプライン E A G A にメモリアクセス用のリクエスト 0 , 1 , 2 , 3 をシリアルに生成する。生成された 4 つのメモリアクセスのリクエストは、メモリアクセスパイプライン E A G A にて、S I M D インダイレクトロード命令以外のロード命令における B 1 サイクルとなる。生成したメモリアクセスのリクエストのエントリ識別情報 I I D は S I M D 演算パイプライン F L A から送られたものを使用し、フェッチポート F P は S I M D 演算パイプライン F L A の F P 値とそれに 1 , 2 , 3 を加算した値を使用する。S I M D 演算パイプライン F L A の S I M D 演算器 3 3 0 が、タイミング 7 , 8 , 9 , 1 0 の X 1 , X 2 , X 3 , X 4 サイクルで S I M D レジスタ（または S I M D リネーミングレジスタ）から読み出した S I M D のデータのうち要素 0 と要素 1 と要素 2 と要素 3 をそれぞれ、シリアルにメモリアクセスパイプライン E A G A のアドレス選択回路 3 1 3 に転送する。タイミング 8 , 9 , 1 0 , 1 1 (A サイクル) で、アドレス選択回路 3 1 3 は、S I M D 演算パイプライン F L A から転送されてきたアドレスをそれぞれ選択し、一次データキャッシュ 3 1 2 に転送する。一次データキャッシュにアクセスしたデータすべてが存在した場合、一次データキャッシュ 3 1 2 は、タイミング 1 3 , 1 4 , 1 5 , 1 6 で読み出したデータを S I M D リネーミングレジスタ 3 3 1 に転送し、タイミング 1 6 ですべてのメモリアクセスの完了報告を行う。その結果、C S E は、命令完了の判定を行い、S I M D リネーミングレジスタ 3 3 1 のデータを S I M D レジスタ 3 3 2 に転送する。

30

40

【 0 0 6 9 】

以上、S I M D インダイレクトロード命令について説明したが、S I M D インダイレクトストア命令でも、S I M D 演算パイプライン F L A がメモリアクセスパイプライン E A G A に S I M D 幅の数のメモリアクセスのリクエストを生成することと、S I M D 演算器が S I M D 幅の数のアドレスを S I M D レジスタから並列に取得してメモリアクセスパイプライン E A G A にシリアルに転送することと、一次データキャッシュに S I M D 幅の数のメモリストアのリクエストを投入することは同じである。S I M D インダイレクトストア命令の場合は、一次データキャッシュは S I M D レジスタに格納されている S I M D 幅の数のデータを一次キャッシュメモリまたはメモリに書き込む。

【 0 0 7 0 】

50

〔本実施の形態におけるSIMDインダイレクトメモリアクセスの詳細説明〕

図11は、本実施の形態のSIMDインダイレクトメモリアクセス（ロードまたはストア）命令を実行するCPUコアの構成を示す図である。図11のCPUコアの構成におけるSIMDダイレクトメモリアクセスの詳細な説明を行う。

【0071】

図11のCPUコア30は、図6と異なり、メモリアクセス用リザベーションステーションRSA（またはメモリアクセスエントリ部）が、メモリアクセス命令のエントリを出力するメモリアクセスパイプラインとして、2つのパイプラインEAGA, EAGBを有する。それに対応して、一次データキャッシュ312は、2つのメモリアクセス要求を並列に処理する構成を有する。また、浮動小数点SIMDリザベーションステーションRSF（またはマルチデータ命令エントリ部）が、SIMD命令のエントリを出力するSIMD演算パイプラインとして、2つのパイプラインFLA, FLBを有する。そして、SIMD演算パイプラインFLA, FLBは、最大SIMD幅4と同じ数の浮動小数点SIMD演算器330を、それぞれ有する。浮動小数点SIMDレジスタ332と浮動小数点SIMDリネーミングレジスタ331は、最大SIMD幅4と同じ数のレジスタが一括してレジスタ番号で指定可能である。それ以外の構成は、図6と同じである。

【0072】

したがって、SIMD演算パイプラインFLAは、SIMDインダイレクトアクセスメモリ命令のエントリにตอบสนองして、2つのメモリアクセス要求を2つのメモリアクセスパイプラインEAGA, EAGBに同時に生成することができ、SIMD演算器330は、SIMDレジスタ332から取得した2つのアドレスを2つのメモリアクセスパイプラインEAGA, EAGBのアドレス選択回路313に並列に転送することができる。後述する図13に示す通りである。

【0073】

SIMD幅が2の場合は、SIMD演算パイプラインFLAは、バス334を介して、1サイクルで2つのメモリアクセス要求を2つのパイプラインEAGA, EAGBに生成する。すなわち、演算用インタフェース333がフラグ信号群をバス334を介してアドレス用インタフェース310に転送し、アドレス用インタフェース310は、その転送されたフラグ信号群に基づいて、1サイクルで2つのメモリアクセス要求を2つのパイプラインEAGA, EAGBに生成する。そして、SIMD演算器330は、バス335を介して、1サイクルで2つのアドレスを2つのパイプラインEAGA, EAGBのアドレス選択回路313に転送する。アドレス選択回路313内のセレクトL5（図6参照）は、前述のとおり、インダイレクトフラグ信号B1__EAGA__INDIRECT, B1__WAGB__INDIRECTの「1」により、バス335側を選択し、SIMD演算器330から供給される2つのアドレスを2つのパイプラインEAGA, EAGBに出力する。これにより、アドレス用インタフェース310が生成した2つのメモリアクセス要求に、アドレス選択回路313のサイクルAのステージで、バス335から供給された2つのアドレスが加えられる。

【0074】

また、SIMD幅が4の場合は、SIMD演算パイプラインFLAは、2サイクルで4つのメモリアクセス要求を2つのパイプラインEAGA, EAGBに生成し、2サイクルで4つのアドレスを転送する。図14に示すとおりである。

【0075】

図8, 図9のフローチャート図は、図11のCPUコアの構成にも適用できる。ただし、図1のCPUコアは、2つのメモリアクセスパイプラインEAGA, EAGBを有するので、図8の工程S24-S27, 工程S29-S32をそれぞれ1回行えば良い。

【0076】

〔演算用インタフェース331とアドレス用インタフェース310によるインダイレクトメモリアクセス要求の生成〕

図12は、演算用インタフェース331とアドレス用インタフェース310の構成を示

10

20

30

40

50

す図である。演算用インタフェース 331 は、RSF から投入される演算命令のエントリから後段の SIMD 演算器 330 などに対して制御信号を適切なタイミングで出力する。同様に、アドレス用インタフェース 310 は、RSA から投入されるメモリアクセス命令のエントリから後段のオペランドアドレス生成器 311 などに対して制御信号を適切なタイミングで出力する。

【0077】

アドレス用インタフェース 310 は、RSA から 2 つのパイプライン EAGA, EAGB に投入された、SIMD インダイレクトメモリアクセス命令以外の通常メモリアクセス命令のエントリのフラグ信号を、ラッチ回路群 F1_A, F1_B でラッチし、後段のオペランドアドレス生成器 311 に転送する。一方、演算用インタフェース 331 は、RSF から SIMD 演算パイプライン FLA に投入された SIMD インダイレクトメモリアクセス命令のエントリのフラグ信号を、アドレス用インタフェース 310 にバス 334 を介して転送する。そして、アドレス用インタフェース 310 内のアンドゲート A1, A2, ラッチ回路群 F2, F3, セレクタ L1, L2, L3, L4, オアゲート A_R1, R2, 加算器 ADD1, ADD2 らの回路が、その転送されてきたフラグ信号に基づいて、2 つのメモリアクセスパイプライン EAGA, EAGB に、それぞれメモリアクセスのリクエストを生成する。

【0078】

図 12 では、アドレス用インタフェース 310 が破線で囲まれた回路を有するように示されている。しかし、破線で囲まれた回路の一部を演算用インタフェース 333 が有するようにしてもよい。したがって、演算用インタフェース 333 とアドレス用インタフェース 310 とバス 334 の構成により、SIMD 演算パイプライン FLA が、2 つのメモリアクセスパイプライン EAGA, EAGB に、それぞれメモリアクセスのリクエストを生成する。

【0079】

図 12 の各信号について説明する。

【0080】

パイプライン FLA 側のエントリのフラグ信号については、次の通りである。入力信号（バリッド信号）B1__FLA__VALID__EAITF は、浮動小数点・SIMD パイプラインの B1 サイクルでパイプライン FLA の SIMD 演算器 330 に演算要求が出たときに 1 となる。

【0081】

入力信号（インダイレクト信号）B1__FLA__INDIRECT__EAITF は、浮動小数点・SIMD パイプラインの B1 サイクルで演算要求が SIMD インダイレクトメモリアクセス命令であった場合 1 となる。

【0082】

入力信号（4 SIMD 信号）B1__FLA__4SIMD__EAITF は、浮動小数点・SIMD パイプラインの B1 サイクルで SIMD 幅が 4 であるときに 1 となる。

【0083】

入力信号（IID 信号）B1__FLA__IID__EAITF には、パイプライン FLA で実行される命令のエントリの識別情報 IID が転送される。

【0084】

入力信号（FP 信号）B1__FLA__FP__EAITF は、SIMD インダイレクトメモリアクセス命令において命令デコーダ 305 で確保したフェッチポート FP の先頭の番号を転送する。

【0085】

パイプライン EAGA, EAGB 側のエントリのフラグ信号については、次の通りである。入力信号（バリッド信号）P__EAGA__VALID, P__EAGB__VALID は、RSA からオペランドアドレス生成器 331 及び一次データキャッシュ 312 へメモリアクセス要求が出力された時に 1 となる。

【0086】

10

20

30

40

50

入力信号（FP信号）P__EAG A__FP，P__EAG B__FPには，RSAからオペランドアドレス生成器331にメモリアクセス要求が出たとき，一次キャッシュメモリ312で使用するフェッチポート番号FP番号が転送される。

【0087】

入力信号（IID信号）P__EAG A__IID，P__EAG B__IIDには，RSAからオペランドアドレス生成器331にメモリアクセス要求が出たとき，それぞれの要求に対応するエントリ識別情報IIDが転送される。

【0088】

アドレス用インタフェース回路310は，SIMDインダイレクトメモリアクセス命令のエントリがSIMD演算パイプラインFLAに投入された場合，演算用インタフェース333が出力するフラグ信号を用いて，メモリアクセスパイプラインEAG A，EAG Bに2つもしくは4つのメモリアクセス要求を生成する。このメモリアクセス要求は，以下に説明する4つの出力信号B1__EAG A__***と，4つの出力信号B1__EAG B__***に対応する。また，アドレス用インタフェース回路310は，通常のメモリアクセス命令のエントリがメモリアクセスパイプラインEAG A，EAG Bに投入された場合は，そのエントリのフラグ信号をそのままオペランドアドレス生成器311に転送する。

【0089】

出力信号（バリッド信号）B1__EAG A__VALID__ORは，オアゲートR1により出力され，RSFが投入したSIMDインダイレクトメモリアクセス命令により生成されたメモリアクセス要求と，RSAからの通常のメモリアクセス命令に対するメモリアクセス要求の論理和である。このバリッド信号が1であるとき，メモリアクセスパイプラインEAG Aのオペランドアドレス生成器311及び一次データキャッシュ312へのメモリアクセス要求が有効となる。

【0090】

出力信号（バリッド信号）B1__EAG B__VALID__ORは，メモリアクセスパイプラインEAG B側のバリッド信号であり，上記と同様である。

【0091】

出力信号（インダイレクト信号）B1__EAG A__INDIRECT，出力信号B1__EAG B__INDIRECTは，対応するバリッド信号B1__EAG A__VALID__OR，B1__EAG B__VALID__OR信号が1であるときに有効になる信号であり，メモリアクセス要求がSIMDインダイレクトメモリアクセス命令により生成されたことを示す。オアゲートR2が出力する。この信号は，後続のオペランドアドレス生成器311を経由してアドレス選択回路313に転送され，アドレス選択回路313においてSIMD演算器330からバス335を介して転送されるアドレスを選択するために使用される。

【0092】

出力信号（IID信号）B1__EAG A__IID，出力信号（IID信号）B1__EAG B__IIDは，対応するバリッド信号B1__EAG A__VALID__OR，B1__EAG B__VALID__OR信号が1であるときに有効になる信号である。SIMDインダイレクトメモリアクセス命令である場合，セクタL4が演算用インタフェース333から転送されてきた入力信号B1__FLA__IID__EAITFのエントリ識別情報IIDを選択する。もしそうでない場合，セクタL4は，RSAからのIID信号P__EAG A__IID，P__EAG B__IIDを選択する。

【0093】

出力信号（FP信号）B1__EAG A__FP，出力信号（FP信号）B1__EAG B__FPは，対応するバリッド信号B1__EAG A__VALID__OR，B1__EAG B__VALID__OR信号が1であるときに有効になる信号である。SIMDインダイレクトメモリアクセス命令の場合の場合で，SIMD幅が2である場合は，入力FP信号B1__FLA__FP__EAITFで転送されてきたFP値と，加算器ADD2で+1加算したFP値とが，セクタL3で選択され出力される。一方，SIMD幅が4である場合は，次の

クロックサイクルで、入力FP信号B1__FLA__FP__EAITFで転送されてきたFP値に加算器ADD1で+2されたFP値と、加算器ADD2で+1加算したFP値とが、セクタL3で選択され出力される。例えば、SIMD幅4であり、SIMDインダイレクトメモリアクセス命令でありかつFP信号B1__FLA__FP__EAITFで転送された値が5であった場合、図14のタイミング6でパイプラインEAGAに生成されたリクエストのFP信号B1__EAGA__FPは5、パイプラインEAGBに生成されたリクエストのFP信号B1__EAGB__FPは6、タイミング7でパイプラインEAGAに生成されたリクエストのFP信号B1__EAGA__FPは7、パイプラインEAGBに生成されたリクエストのFP信号B1__EAGB__FPは8になる。SIMDインダイレクトメモリアクセス命令でない場合は、RSAからのFP信号P__EAGA__FP、P__EAGB__FPがそれぞれセクタL3で選択される。

10

【0094】

図13は、SIMD幅が2のSIMDインダイレクトメモリアクセス命令の場合のパイプラインとアドレス用インタフェース310の入出力信号変化を示す図である。SIMD演算パイプラインFLAの演算用インタフェース333が、タイミング5のサイクルB1で図12の入力信号(B1__FLA__***)を出力し、アドレス用インタフェース310が、それらの入力信号に基づいてタイミング6で図12の出力信号(B1__EAGA__*** , B1__EAGB__***)によるメモリアクセス要求を生成する。

【0095】

タイミング5の入力IID信号B1__FLA__IDD__EAITF(=2)がセクタL1を介してラッチF2でラッチされ、タイミング6の出力IID信号B1__EAGA__IID, B1__EAGB__IIDが共に2になる。

20

【0096】

タイミング5の入力バリッド信号B1__FLA__VALID__EAITF(=1)と入力インダイレクト信号B1__FLA__INDIRECT__EAITF(=1)の論理積がアンドゲートA1を介してラッチF2でラッチされ、オアゲートR1, R2を介して、タイミング6の出力バリッド信号B1__EAGA__VALID__OR, B1__EAGB__VALID__ORが共に1になり、出力インダイレクト信号B1__EAGA__INDIRECT, B1__EAGB__INDIRECTも共に1になる。

【0097】

30

そして、タイミング5の入力FP信号B1__FLA__EAITF(=4)がセクタL2を介してラッチF2__FPでラッチされ、セクタL3を介して、タイミング6の出力FP信号B1__EAGA__FP(=4), B1__EAGB__FP(=5)になる。

【0098】

上記の動作により、SIMD演算用パイプラインFLAは、演算用インタフェース333が出力するフラグ信号により、タイミング6で、アドレス用インタフェース310内の2つのメモリアクセスパイプラインEAGA, EAGBに、2つのメモリアクセス要求を生成する。

【0099】

図14は、SIMD幅が4のSIMDインダイレクトメモリアクセス命令の場合のパイプラインとアドレス用インタフェース回路310の入出力信号変化を示す図である。SIMD演算パイプラインFLAの演算用インタフェース333が、タイミング5のサイクルB1で図12の入力信号(B1__FLA__***)を出力し、アドレス用インタフェース310が、それらの入力信号に基づいて、タイミング6, 7で図12の出力信号(B1__EAGA__*** , B1__EAGB__***)によるメモリアクセス要求を生成する。

40

【0100】

タイミング5の演算用インタフェース333が出力する入力信号と、タイミング6でアドレス用インタフェース310内のメモリアクセスパイプラインEAGA, EAGBに生成される出力信号は、図13のSIMD幅2の場合と同じである。

【0101】

50

ただし，S I M D幅4の場合は，タイミング6のラッチF 2の入力I I D信号をセレクトAL 1を介してラッチF 2が再度ラッチし，タイミング6のアンドゲートA 1の出力と入力4 S I M D信号B 1 __ F L A __ 4 S I M D __ E A I T Fのラッチ信号の論理積を，アンドゲートA 2を介してラッチF 3がラッチする。また，タイミング6の入力F P信号B 1 __ F L A __ F P __ E A I T Fの値に加算器A D D 1で+ 2した値を，セレクトAL 2を介してラッチF 2 __ F Pがラッチする。それに対応して，タイミング7では，タイミング6と同様に，メモリアクセスパイプラインE A G A，E A G Bの出力バリッド信号，出力インダイレクト信号が1を維持し，出力I I D信号が2を維持し，出力F P信号が6，7になる。

【0102】

10

上記の動作により，S I M D演算用パイプラインF L Aは，演算用インタフェース3 3 3の出力する信号により，タイミング6で，アドレス用インタフェース3 1 0内の2つのメモリアクセスパイプラインE A G A，E A G Bに，2つのメモリアクセス要求を生成し，さらに，タイミング7でメモリアクセスパイプラインE A G A，E A G Bにさらに2つのメモリアクセス要求を生成する。

【0103】

図15は，図6の1つのメモリアクセスパイプラインE A G Aを有する場合の演算用インタフェース3 3 1とアドレス用インタフェース3 1 0の構成を示す図である。S I M Dインダイレクトメモリアクセス命令の場合，図12と異なり次のような動作になる。図10も参照して説明する。

20

【0104】

まず，タイミング5の入力バリッド信号B 1 __ F L A __ V A L I D __ E A I T Fと入力インダイレクト信号B 1 __ F L A __ I N D I R E C T __ E A I T Fの論理積が，アンドゲートA 1を介して2つのラッチF 2 __ 1でラッチされ，そのラッチF 2 __ 1の出力がさらに次のタイミングでラッチF 2 __ 2でラッチされ，タイミング6，7で，出力バリッド信号B 1 __ E A G A __ V A L I D __ O Rと出力インダイレクト信号B 1 __ E A G A __ I N D I R E C Tが2サイクルにわたり1を出力する。

【0105】

S I M D幅が4の場合は，さらに，タイミング7のラッチF 2 __ 2の出力と入力4 S I M D信号のラッチF 2 __ 2の出力の論理積が，アンドゲートA 2を介してラッチF 3 __ 1でラッチされ，そのラッチF 3 __ 1の出力がさらに次のタイミングでラッチされ，タイミング8，9で，出力バリッド信号B 1 __ E A G A __ V A L I D __ O Rと出力インダイレクト信号B 1 __ E A G A __ I N D I R E C Tが2サイクルにわたり1を出力する。

30

【0106】

タイミング5の入力I I D信号B 1 __ F L A __ I I D __ E A I T Fは，セレクトAL 1を介してラッチF 2で4回ラッチされ，タイミング6，7，8，9でセレクトAL 4を介して出力I I D信号B 1 __ E A G A __ I I Dとして出力される。

【0107】

タイミング5の入力F P信号B 1 __ F L A __ F P __ E A I T Fは，セレクトAL 2を介してラッチF 2 __ F Pがラッチし，その後，3サイクルで加算器A D D 1でそれぞれ+ 1したフェッチポートF Pの値をラッチF 2 __ F Pがラッチする。そして，タイミング6，7，8，9で，出力F P信号B 1 __ E A G A __ F Pが，入力F P値，それに+ 1，+ 2，+ 3されたF P値になる。

40

【0108】

[衝突を回避するためのR S AとR A Fによる新たなエントリ投入の抑止]

図16，図17は，S I M D幅2の場合と4の場合での後続するR S Aから投入されるメモリアクセスとの衝突を示す図である。いずれも，図11の例で示している。

【0109】

本実施の形態では，R A FがS I M Dインダイレクトメモリアクセス命令のエントリをS I M D演算パイプラインF L Aに投入すると，S I M D演算パイプラインF L Aが，演

50

算用インタフェース 3 3 3 が出力する信号を利用して、アドレス用インタフェース 3 1 0 内のメモリアクセスパイプライン E A G A , E A G B に複数のメモリアクセス要求を生成する。そのため、その生成されたメモリアクセス要求と後続の R S A から投入されるメモリアクセス要求とが衝突する場合がある。図 1 1 の例では、S I M D 幅が 2 の場合は 1 回メモリアクセス要求が生成されるので 1 回衝突する場合があり、S I M D 幅が 4 の場合は 2 回メモリアクセス要求が生成されるので 2 回衝突する場合がある。図 6 のメモリアクセスパイプライン E A G A が 1 つの例では、S I M D 幅 2 では 2 回衝突し、S I M D 幅 4 では 4 回衝突する場合がある。

【 0 1 1 0 】

図 1 1 の例で説明すると以下のとおりである。図 1 6 , 1 7 には衝突が B 1 への取消線で示されている。

【 0 1 1 1 】

(1) 図 1 6 の S I M D 幅 2 の場合は、タイミング 3 で R S F がパイプライン F L A に S I M D 幅 2 の S I M D インダイレクトメモリアクセス命令のエントリを出力し、タイミング 5 で R S A がパイプライン E A G A もしくは E A G B にメモリアクセス命令のエントリを出力した場合、タイミング 6 で、S I M D インダイレクトメモリアクセス命令により生成されるメモリアクセス要求のサイクル B 1 の信号と、R S A から転送されるメモリアクセス要求のサイクル B 1 の信号が衝突する。

【 0 1 1 2 】

(2) 図 1 7 の S I M D 幅 4 の場合は、タイミング 3 で R S F がパイプライン F L A に S I M D 幅 4 のインダイレクト命令のエントリを出力し、タイミング 5 もしくは 6 において R S A がパイプライン E A G A もしくは E A G B にメモリアクセス命令のエントリを出力した場合、次の衝突が発生する。

【 0 1 1 3 】

すなわち、タイミング 5 で R S A がパイプライン E A G A もしくは E A G B にメモリアクセス命令のエントリを出力した場合、タイミング 6 で S I M D インダイレクトメモリアクセス命令により生成されるメモリアクセス要求のサイクル B 1 の信号と、R S A から転送されるメモリアクセス要求のサイクル B 1 の信号が衝突する。

【 0 1 1 4 】

また、タイミング 6 で R S A がパイプライン E A G A もしくは E A G B にメモリアクセス命令のエントリを出力した場合、タイミング 7 で S I M D インダイレクトメモリアクセス命令により生成されるメモリアクセス要求のサイクル B 1 の信号と、R S A から転送されるメモリアクセス要求のサイクル B 1 の信号とが衝突する。

【 0 1 1 5 】

図 1 8 は、S I M D 幅 4 の場合での後続する S I M D インダイレクトメモリアクセス命令のエントリの投入により生成されるメモリアクセス要求との衝突を示す図である。いずれも、図 1 1 の 2 つのメモリアクセスパイプライン E A G A , E A G B を有する例で示している。

【 0 1 1 6 】

本実施の形態では、S I M D インダイレクトメモリアクセス命令のエントリの投入に回答して、S I M D 演算パイプライン F L A が、演算用インタフェース 3 3 3 が出力する信号を利用して、メモリアクセスパイプライン E A G A , E A G B にメモリアクセス要求を生成する。そのため、その生成されたメモリアクセス要求が、後続の S I M D インダイレクトメモリアクセス命令のエントリの投入に回答してメモリアクセスパイプライン E A G A , E A G B に生成されるメモリアクセス要求と衝突する場合がある。図 1 1 の例では、S I M D 幅が 4 の場合に 2 回メモリアクセス要求が生成されるので、後続の S I M D インダイレクトメモリアクセス命令に対応するメモリアクセス要求と、1 回衝突する場合がある。図 6 の例では、S I M D 幅 2 では 1 回衝突し、S I M D 幅 4 では 3 回衝突する場合がある。

【 0 1 1 7 】

図 1 1 の例で説明すると図 1 8 に示されるように以下のとおりである。図 1 8 には衝突が B 1 への取消線で示されている。

【 0 1 1 8 】

(3) タイミング 3 で R S F が S I M D 幅 4 の S I M D インダイレクトメモリアクセス命令のエントリを出力し、タイミング 4 で R S F が S I M D 幅 2 もしくは 4 の S I M D インダイレクトメモリアクセス命令のエントリを出力した場合、次のとおり衝突が発生する。すなわち、タイミング 3 で R S F から出力された 4 S I M D インダイレクトメモリアクセス命令により生成されたメモリアクセス要求のサイクル B 1 の信号と、次のタイミング 4 で R S F から出力された 2 または 4 S I M D インダイレクトメモリアクセス命令により生成されたメモリアクセス要求のサイクル B 1 の信号とが、タイミング 7 で衝突する。

10

【 0 1 1 9 】

図 1 9 は、インダイレクトメモリアクセス要求の衝突を回避する抑止信号を生成する演算用インタフェース 3 3 3 の構成を示す図である。演算用インタフェース 3 3 3 は、R S F が投入する S I M D インダイレクトメモリアクセス命令のエントリの P サイクルのフラグ信号を入力し、ラッチ群 F 1 0 でラッチし、さらにラッチ群 F 1 1 でラッチする。それにより、演算用インタフェース 3 3 3 は、P サイクルから 2 サイクル後の B 1 サイクルの出力信号を、S I M D 演算パイプライン F L A の S I M D 演算器 3 3 0 と、メモリアクセスパイプライン E A G A、E A G B のアドレス用インタフェース 3 1 0 に転送する。演算用インタフェース 3 3 3 が 2 つのラッチ群 F 1 0、F 1 1 を有するのは、例えば、タイミングを調整するためである。

20

【 0 1 2 0 】

そして、演算用インタフェース 3 3 3 は、P サイクルの 3 つのフラグ信号から R S F への後続の S I M D インダイレクトメモリアクセス命令のエントリの投入を抑止する抑止信号 I N H _ F L A _ I N D I R E C T _ O P と、P T サイクルの 2 つのフラグ信号からとともに、B 1 サイクルの 3 つのフラグ信号からも、R S A への後続のメモリアクセス命令の投入を抑止する抑止信号 I N H _ R S A _ P R I O R I T Y を生成する。

【 0 1 2 1 】

演算用インタフェース 3 3 3 の動作は次のとおりである。

【 0 1 2 2 】

入力信号 (バリッド信号) P _ F L A _ V A L I D は、浮動小数点・S I M D パイプラインの P サイクルでパイプライン F L A への S I M D 演算器に演算要求が出力されたときに 1 となる。

30

【 0 1 2 3 】

入力信号 (インダイレクト信号) P _ F L A _ I N D I R E C T は、入力バリッド信号 P _ F L A _ V A L I D が 1 のときに有効となる信号であり、演算要求が S I M D インダイレクトメモリアクセス命令の場合に、浮動小数点・S I M D パイプラインの P サイクルで 1 となる。

【 0 1 2 4 】

入力信号 (4 S I M D 信号) P _ F L A _ 4 S I M D は、入力バリッド信号 P _ F L A _ V A L I D が 1 のときに有効となる信号であり、S I M D 演算器の演算幅が 4 であるときに浮動小数点・S I M D パイプラインの P サイクルで 1 となる。

40

【 0 1 2 5 】

入力信号 (I D D 信号) P _ F L A _ I I D は、入力バリッド信号 P _ F L A _ V A L I D が 1 のときに有効となる信号であり、パイプライン F L A で実行される演算の C S E のエントリ番号を示す。

【 0 1 2 6 】

入力信号 (F P 信号) P _ F L A _ F P は、入力バリッド信号 P _ F L A _ V A L I D が 1 かつ入力インダイレクト信号 P _ F L A _ I N D I R E C T が 1 のときに有効となる信号であり、S I M D インダイレクトメモリアクセス命令において命令デコーダで確保された一次データキャッシュ内のフェッチポート F P の先頭番号を示す。

50

【0127】

演算用インタフェース333は、5つの入力信号をラッチF10、F11でラッチして中継し、5つの出力信号B1__FLA__VALID__EAITF、B1__FLA__INDIRECT__EAITF、B1__FLA__4SIMD__EAITF、B1__FLA__IID__EAITF、B1__FLA__FP__EAITFを、アドレス用インタフェース310に転送し、メモリアクセスのリクエストを生成させる。

【0128】

同様に、演算用インタフェース333は、4つの入力信号をラッチF10、F11でラッチして中継し、4つの出力信号B1__FLA__VALID、B1__FLA__INDIRECT、B1__FLA__4SIMD、B1__FLA__IIDを、SIMD演算器に転送する。

10

【0129】

演算用インタフェース333では、アンドゲートA4がPサイクルの2つの入力信号P__FLA__VALID、P__FLA__INDIRECTの論理積を後続の通常メモリアクセス命令の抑止信号INH__RSA__PRIORITYとして生成し、RSAに転送する。これにより、RSAは、後続のメモリアクセス命令のエントリのメモリアクセスパイプラインEAGA、EAGBへの投入を抑止する。

【0130】

図16に示されるとおり、タイミング3のPサイクルの2つの信号が全て1の場合に、タイミング4で抑止信号INH__RSA__PRIORITYが1になり、タイミング5においてRSAがメモリアクセス命令のエントリのパイプラインEAGA、EAGBへの投入を抑止する。これにより、タイミング6でB1サイクルの信号が発生せず、衝突が回避される。

20

【0131】

さらに、演算用インタフェース333では、アンドゲートA5がPサイクルの3つの入力信号P__FLA__VALID、P__FLA__INDIRECT、P__FLA__4SIMDの論理積を後続の通常メモリアクセス命令の抑止信号INH__RSA__PRIORITYとして生成し、RSAに転送する。これにより、RSAは、後続のメモリアクセス命令のメモリアクセスパイプラインEAGA、EAGBへの投入を抑止する。

【0132】

30

図17に示されるとおり、タイミング3のPサイクルの3つの信号が全て1の場合に、タイミング5で抑止信号INH__RSA__PRIORITYが1になり、タイミング6においてRSAがメモリアクセス命令のパイプラインEAGA、EAGBへの投入を抑止する。これにより、タイミング7でB1サイクルの信号が発生せず、衝突が回避される。図17では、図16と同様にして、タイミング4で抑止信号INH__RSA__PRIORITYが1になり、タイミング5におけるRSAでのメモリアクセス命令の投入が抑止されている。

【0133】

そして、演算用インタフェース333では、アンドゲートA3がPサイクルの3つの入力信号P__FLA__VALID、P__FLA__INDIRECT、P__FLA__4SIMDの論理積を後続のSIMDインダイレクトメモリアクセス命令の抑止信号INH__FLA__INDIRECT__OPとして生成し、RSFに転送する。これにより、RSFは、後続のSIMDインダイレクトメモリアクセス命令のエントリのSIMD演算パイプラインFLAへの投入を抑止する。

40

【0134】

図18に示されるとおり、タイミング3のPサイクルの3つの信号が全て1の場合に、抑止信号INH__FLA__INDIRECT__OPが1になり、次のタイミング4においてRSFがSIMDインダイレクトメモリアクセス命令のエントリのパイプラインFLAへの投入を抑止する。これにより、タイミング7で生成されたB1サイクルの信号が発生せず、衝突が回避される。

50

【 0 1 3 5 】

図 20 は、R S F とその S I M D インダイレクトメモリアクセス命令のエントリの出力抑止回路を示す図である。R S F は、例えば 20 個のエントリ保持部 3 3 7 を有し、リザベーションステーション R S F に生成された命令のエントリに対応するフラグが格納されている。フラグの例は図 7 に示されている。

【 0 1 3 6 】

各エントリ保持部 3 3 7 に対応する R S F エントリ出力条件検出回路 3 3 8 は、これらのフラグを用い、R S F 内のエントリそれぞれについてパイプラインへの出力可能条件が成立したことを検出する。この R S F エントリ出力条件検出回路 3 3 8 は、R S F それぞれに格納された命令のエントリが処理可能となった場合に 1 を出力し、出力が可能でない場合は 0 を出力する。

10

【 0 1 3 7 】

抑止回路 3 3 9 は、演算用インタフェース 3 3 3 で生成された抑止信号 $INH_FLA_INDIRECT_OP$ と、R S F のエントリ保持部 3 3 7 に格納されている $INDIRECT$ フラグが共に 1 の場合に、出力条件検出回路 3 3 8 の出力を強制的に 0 にする。これにより、対応する R S F エントリが出力可能であるか否かを示す $READY$ 信号がラッチ RSF_xx_READY にラッチされる。 xx は 00 - 19 である。

【 0 1 3 8 】

FLA 出力選択回路 3 40 は、この $READY$ 信号が 1 の R S F エントリから、次に出力する R S F エントリを選択し、演算用インタフェース 3 3 3 へ出力する。ただし、S I M D インダイレクトメモリアクセス命令の場合は、 $INDIRECT$ フラグが 1 になるので、抑止信号 $INH_FLA_INDIRECT_OP$ が 1 となったとき、そのエントリの $READY$ 信号が 0 となるため、 FLA 出力選択回路 3 40 はその S I M D インダイレクトメモリアクセス命令のエントリを選択することはない。S I M D インダイレクトメモリアクセス命令以外の命令の場合は、 $INDIRECT$ フラグが 0 になるので、エントリ出力条件検出回路 3 3 8 の出力が $READY$ 信号として使用される。したがって、S I M D インダイレクトメモリアクセス命令以外の命令については、必要な資源が準備されたエントリがあればその命令のエントリが出力される。これにより R S F が、抑止信号 $INH_FLA_INDIRECT_OP$ に応じて、S I M D インダイレクトメモリアクセス命令のエントリの出力を抑止する。

20

30

【 0 1 3 9 】

図 21 は、R S A とその通常のメモリアクセス命令のエントリの出力抑止回路を示す図である。R S A は、例えば 20 個のエントリ保持部 3 14 を有する。各エントリ保持部 3 14 に対応する R S A エントリ出力条件検出回路 3 15 は、R S A エントリそれぞれについてパイプラインへの出力可能条件が成立したことを検出する。この R S A エントリ出力条件検出回路 3 15 は、R S A それぞれに格納された命令が処理可能となった場合に 1 を出力し、出力が可能でない場合は 0 を出力する。

【 0 1 4 0 】

抑止回路 3 16 は、演算用インタフェース 3 3 3 で生成された抑止信号 $INH_RSA_PRIORITY$ が 1 の場合に、R S A エントリ出力条件検出回路 3 15 の出力を強制的に 0 にする。これにより、対応する R S A エントリが出力可能であるか否かを示す $READY$ 信号がラッチ RSA_xx_READY にラッチされる。

40

【 0 1 4 1 】

$EAGA/EAGB$ 出力選択回路 3 17 は、 $READY$ 信号が 1 の R S A エントリから出力する R S A エントリを選択し、メモリアクセスパイプライン $EAGA$ または $EAGB$ に出力し、アドレス用インタフェースへ転送する。抑止信号 $INH_RSA_PRIORITY$ が 1 のとき、R S A エントリ出力条件検出回路 3 15 から出力された値に関わらず、すべての R S A の $READY$ 信号が 0 になる。これにより $EAGA/EAGB$ 出力選択回路 3 15 は出力可能なエントリがないため、メモリリクエストをメモリアクセスパイプライン $EAGA$ 、 $EAGB$ にエントリを出力しない。これにより R S A が、抑止信号 INH

50

H_RSA_PRIORITYに応じて、メモリアクセス命令のエントリの出力を抑止する。

【0142】

図22は、CSE内の完了待ち合わせ回路を示す図である。図22には、CSEの1つのエントリに対する完了待ち合わせ回路が示されている。

【0143】

まず、CSEのエントリにインダイレクトフラグCSE_INDIRECTが含まれている。CSEのエントリがSIMDインダイレクトメモリアクセス命令の場合、そのエントリのインダイレクトフラグCSE_INDIRECTが1になる。また、その命令が4SIMDの場合に4SIMD信号CSE_4SIMDが1になる。CSEにエントリされた命令がSIMDインダイレクトメモリアクセス命令であった場合、一次データキャッシュ312が同じCSEのエントリ番号IIDに対して2SIMDなら2回、4SIMDなら4回の完了報告をCSEに行く。

【0144】

入力信号(インダイレクト信号)CSE_INDIRECT、入力信号(4SIMD信号)CSE_4SIMDは、命令デコーダ305によりCSEに登録されたエントリのフラグである。入力信号CSE_INDIRECTが1のときCSEのエントリがSIMDインダイレクトメモリアクセス命令であることを示す。入力信号CSE_4SIMDが1のとき、CSEのエントリのSIMD幅が4であることを示し、0のときSIMD幅が2であることを示す。

【0145】

本実施の形態の一次データキャッシュ312は、2つの独立したメモリアクセスを同時に処理する。そのため、一次データキャッシュ312は、メモリアクセス完了信号を2つ独立して通知する。

【0146】

入力信号RT_STV_0、RT_STV_1は一次データキャッシュから転送されるメモリアクセスの完了信号である。

【0147】

入力信号RT_STV_0_CSE_SEL、RT_STV_1_CSE_SELは、一次データキャッシュにおいて処理中のエントリ番号IIDが、CSEのエントリ番号と一致したとき1となる。

【0148】

RT_STV_0とRT_STV_0_CSE_SELが1となったとき、もしくはRT_STV_1とRT_STV_1_CSE_SELが1となったとき、アンドゲートA8またはA9の出力により、CSEへのメモリアクセス完了報告が有効となる。メモリアクセス完了報告が有効になると、加算器351が3ビットの入力信号に+1加算してメモリアクセス完了回数記憶素子353に出力する。

【0149】

命令デコードがCSEにエントリを作成したときに、メモリアクセス完了回数記憶素子353を0にリセットする。その後、一次データキャッシュ312からの完了報告により、RT_STV_0とRT_STV_0_CSE_SEL両方が1になった場合、もしくはRT_STV_1とRT_STV_1_CSE_SEL両方が1となった場合、加算器351がメモリアクセス完了回数を+1加算する。

【0150】

メモリアクセス命令の種類により、メモリアクセス完了回数が規定の値(1, 2, 4回)となったとき、出力信号(完了信号)CSE_MEM_COMPが1となる。アンドゲートA6により、SIMDインダイレクトメモリアクセス命令かつSIMD幅が4の場合、4回のメモリアクセス完了が通知されたとき、加算器351のビット2の出力が1になり、完了信号CSE_MEM_COMPが1となる。SIMDインダイレクトメモリアクセス命令かつSIMD幅が2の場合、2回のメモリアクセス完了が通知されたとき、加算

10

20

30

40

50

器 3 5 1 のビット 1 の出力が 1 になり，完了信号が 1 となる。そして，S I M D インダイレクト命令でないメモリアクセス命令の場合は，1 回メモリアクセス完了が通知されたとき，加算器 3 5 1 のビット 0 の出力が 1 になり，完了信号 C S E _ M E M _ C O M P が 1 となる。

【 0 1 5 1 】

完了判定回路 3 5 4 は，この完了信号 C S E _ M E M _ C O M P を入力し，命令が完了可能となったことを示す信号を生成する。完了判定回路 3 5 4 は，処理が完了した命令をプログラムの順番に完了したと判定し，例えばリネーミングレジスタからレジスタに処理結果を転送し，エントリを開放する。

【 0 1 5 2 】

以上の通り，本実施の形態によれば，S I M D インダイレクトメモリアクセス命令のエントリを R S F に生成し，そのエントリが S I M D 演算用パイプライン F L A に出力されると，メモリアクセスパイプライン E A G A，E G A B に S I M D 幅に応じた数のメモリアクセスを生成し，S I M D 演算器 3 3 0 が複数の S I M D レジスタ 3 3 2 に格納されている独立した複数のアドレスを取得してメモリアクセスパイプライン E A G A，E G A B に転送し，一次データキャッシュ 3 1 2 がその複数のアドレスを使用して複数の S I M D レジスタ 3 3 2 に格納されている複数のデータについてメモリアクセスを行う。したがって，命令デコーダや C S E，R S A，R S F のエントリなどの資源を効率的に使用して S I M D インダイレクトメモリアクセス命令を実行する。

【 0 1 5 3 】

以上の実施の形態をまとめると，次の付記のとおりである。

【 0 1 5 4 】

(付記 1)

命令をデコードする命令デコーダと，

前記命令デコーダによりメモリアクセス命令のエントリを生成されるメモリアクセスエントリ部と，

前記メモリアクセスエントリ部から出力された前記メモリアクセス命令のエントリをメモリに対して実行するメモリアクセスパイプラインと，

前記命令デコーダにより複数のデータを 1 つの命令で処理するマルチデータ命令のエントリを生成されるマルチデータ命令エントリ部と，

複数の演算器と複数のマルチデータ命令用レジスタとを有し，前記マルチデータ命令エントリ部から出力された前記マルチデータ命令のエントリの処理を前記複数の演算器により並列に処理し，前記複数のマルチデータ命令用レジスタに演算結果を格納する演算パイプラインとを有し，

前記演算パイプラインは，前記複数のマルチデータ命令用レジスタに格納されている複数のメモリアドレスについて前記メモリにメモリアクセスするマルチデータインダイレクトメモリアクセス命令のエントリの出力に应答して，前記メモリアクセスパイプラインに前記マルチデータインダイレクトメモリアクセス命令に対応する複数のメモリアクセス要求を生成し，前記複数の演算器が前記複数のマルチデータ命令用レジスタから取得した前記複数のメモリアドレスを前記メモリアクセスパイプラインに供給する演算処理装置。

【 0 1 5 5 】

(付記 2)

前記演算パイプラインは，前記メモリアクセスパイプラインの第 1 サイクルのステージに前記複数のメモリアクセス要求を生成し，前記メモリアクセスパイプラインの前記第 1 サイクルより後の第 2 サイクルのステージに前記複数のメモリアドレスを供給する付記 1 に記載された演算処理装置。

【 0 1 5 6 】

(付記 3)

前記演算パイプラインは，前記メモリアクセスパイプラインに生成した複数のメモリアクセス要求のパイプライン転送タイミングにあわせて前記複数のメモリアドレスを供給す

10

20

30

40

50

る

付記 2 に記載された演算処理装置。

【 0 1 5 7 】

(付記 4)

さらに、前記メモリアクセスパイプラインに接続されたキャッシュユニットを有し、
前記演算パイプラインは、前記メモリアクセスパイプラインに生成する前記複数のメモリアクセス要求に前記キャッシュユニット内のアクセス先メモリアドレスを格納する複数のフェッチポートの識別情報を含める

付記 1 に記載された演算処理装置。

【 0 1 5 8 】

(付記 5)

前記演算パイプラインは、前記メモリアクセスパイプラインに、前記複数のメモリアクセス要求をシリアルに生成し、前記複数のメモリアドレスをシリアルに供給する

付記 1 , 2 , 3 のいずれかに記載された演算処理装置。

【 0 1 5 9 】

(付記 6)

前記メモリアクセスパイプラインが複数設けられ、
前記演算パイプラインは、前記複数のメモリアクセスパイプラインに、前記複数のメモリアクセス要求の少なくとも一部のメモリアクセス要求を並列に生成し、前記複数のメモリアドレスの少なくとも一部のアドレスを並列に供給する

付記 1 , 2 , 3 のいずれかに記載された演算処理装置。

【 0 1 6 0 】

(付記 7)

さらに、前記メモリアクセスパイプラインに接続されたキャッシュユニットを有し、
前記キャッシュユニットは、前記複数のメモリアクセス要求に応答して、前記複数のマルチデータ命令用レジスタとの間でデータ転送を行う

付記 1 に記載された演算処理装置。

【 0 1 6 1 】

(付記 8)

前記演算パイプラインは、前記メモリアクセスエントリ部に抑止信号を出力して、前記メモリアクセスエントリ部に、前記メモリアクセスパイプラインに生成する前記複数のメモリアクセス要求と衝突するメモリアクセス命令のエントリの出力を抑止させる

付記 1 に記載された演算処理装置。

【 0 1 6 2 】

(付記 9)

前記演算パイプラインは、前記マルチデータ命令エントリ部に抑止信号を出力して、前記マルチデータ命令エントリ部に、前記メモリアクセスパイプラインにシリアルに生成する前記メモリアクセス要求と衝突するマルチデータインダイレクトメモリアクセス命令のエントリの出力を抑止させる

付記 5 に記載された演算処理装置。

【 0 1 6 3 】

(付記 1 0)

前記演算パイプラインに出力される前記マルチデータインダイレクトメモリアクセス命令のエントリは、マルチデータインダイレクトメモリアクセスを示すインダイレクトメモリアクセス信号と、前記複数のデータの数を示すマルチデータ幅情報信号とを有し、

前記演算パイプラインは、前記メモリアクセスパイプラインに、前記マルチデータ幅情報信号が示す数の前記メモリアクセス要求を生成し、前記マルチデータ幅情報信号が示す数の前記複数のメモリアドレスを供給する

付記 1 に記載された演算処理装置。

【 0 1 6 4 】

10

20

30

40

50

(付記 1 1)

命令をデコードする命令デコーダと、

前記命令デコーダによりメモリアクセス命令のエントリを生成されるメモリアクセスエントリ部と、

前記メモリアクセスエントリ部から出力された前記メモリアクセス命令のエントリをメモリに対して実行するメモリアクセスパイプラインと、

前記命令デコーダにより複数のデータを1つの命令で処理するマルチデータ命令のエントリを生成されるマルチデータ命令エントリ部と、

複数の演算器と複数のマルチデータ命令用レジスタとを有し、前記マルチデータ命令エントリ部から出力された前記マルチデータ命令のエントリの処理を前記複数の演算器により並列に処理し、前記複数のマルチデータ命令用レジスタに演算結果を格納する演算パイプラインとを有する演算処理装置の制御方法において、

前記演算パイプラインが、前記複数のマルチデータ命令用レジスタに格納されている複数のメモリアドレスについて前記メモリにメモリアクセスするマルチデータインダイレクトメモリアクセス命令のエントリの投入に応答して、前記メモリアクセスパイプラインに前記マルチデータインダイレクトメモリアクセス命令に対応する複数のメモリアクセス要求を生成し、

前記演算パイプラインが、前記複数の演算器が前記複数のマルチデータ命令用レジスタから取得した前記複数のメモリアドレスを前記メモリアクセスパイプラインに供給する演算処理装置の制御方法。

【 0 1 6 5 】

(付記 1 2)

前記演算パイプラインが、前記メモリアクセスパイプラインの第1サイクルのステージに前記複数のメモリアクセス要求を生成し、前記メモリアクセスパイプラインの前記第1サイクルより後の第2サイクルのステージに前記複数のメモリアドレスを供給する付記11に記載された演算処理装置の制御方法。

【 符号の説明 】

【 0 1 6 6 】

3 0 1 : 命令フェッチアドレス生成器

3 0 2 : 分岐予測機構

3 0 3 : 一次命令キャッシュ

3 0 4 : 命令バッファ

3 0 5 : 命令デコーダ

3 0 6 : レジスタリネーミング部

R S A : メモリアクセス用リザベーションステーション (アドレス生成リザベーションステーション) , メモリアクセスエントリ部

3 1 0 : アドレス用インタフェース

3 1 1 : オペランドアドレス生成器

3 1 2 : 一次データキャッシュ

3 1 3 : アドレス選択回路

E A G A , E A G B : オペランドアドレス生成器 , メモリアクセスパイプライン

S T B : ストアバッファ

R S E : 固定小数点演算用リザベーションステーション

3 2 0 : 固定小数点演算器

3 2 2 : 固定小数点レジスタ

3 2 1 : 固定小数点リネーミングレジスタ

R S F : 浮動小数点演算用リザベーションステーション , マルチデータ命令エントリ部

3 3 0 : 浮動小数点 S I M D 演算器 , マルチデータ命令用演算器

3 3 2 : 浮動小数点 S I M D レジスタ , マルチデータ命令用レジスタ

3 3 1 : 浮動小数点 S I M D リネーミングレジスタ

10

20

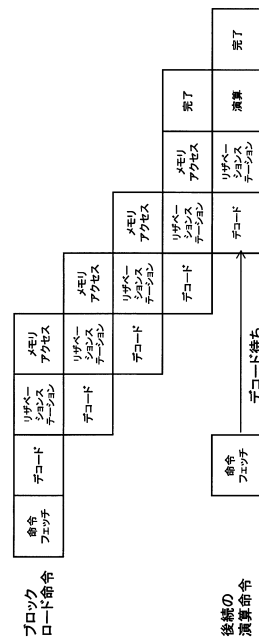
30

40

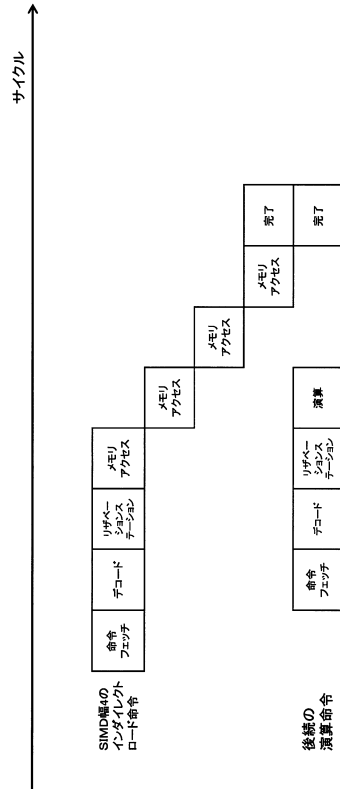
50

PC : プログラムカウンタ

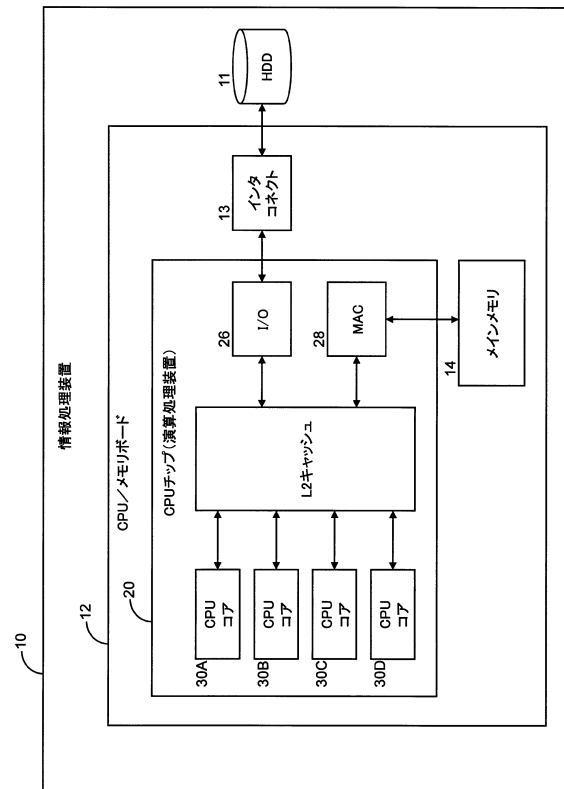
【圖 2】



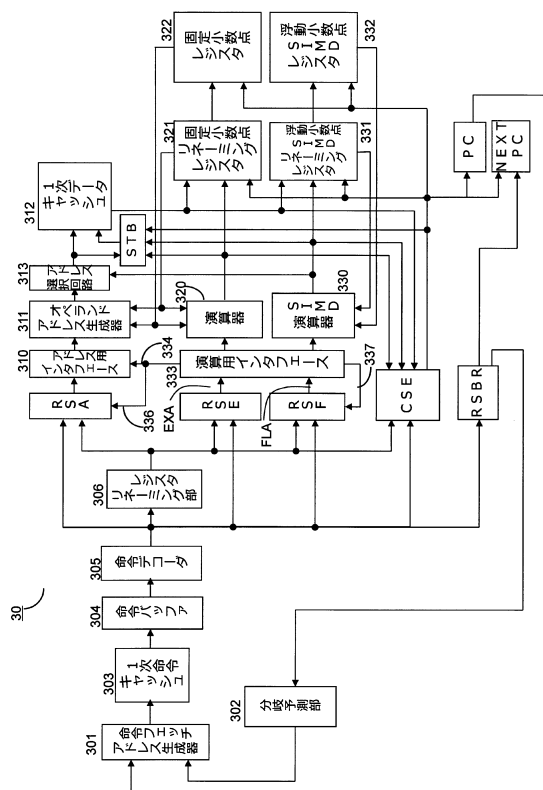
【 図 3 】



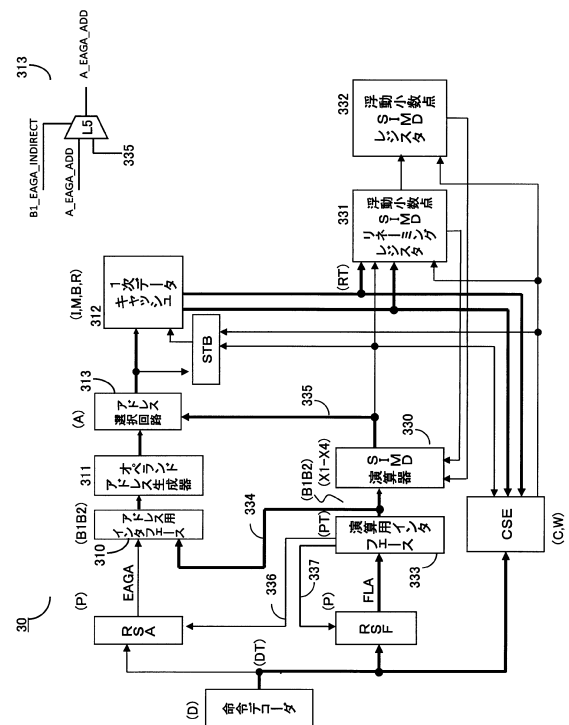
【 図 4 】



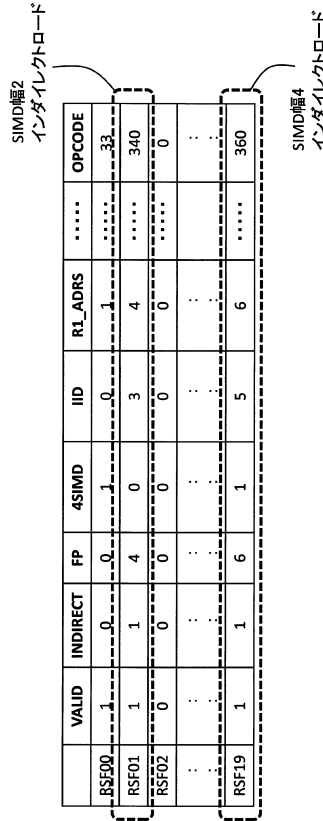
【 図 5 】



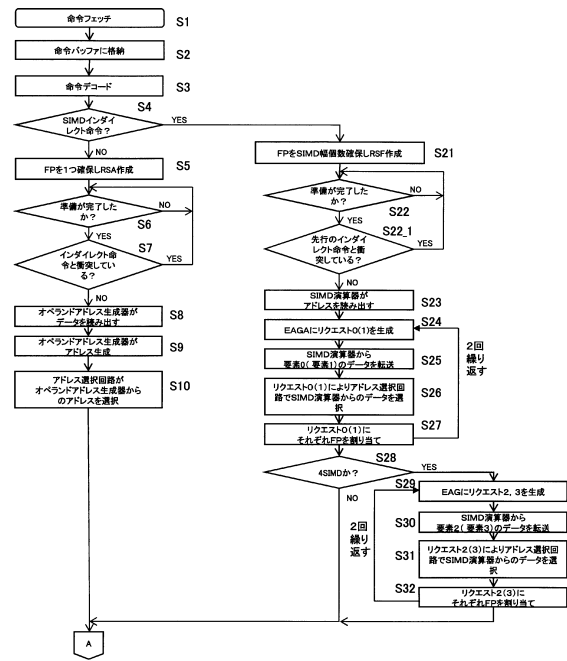
【 図 6 】



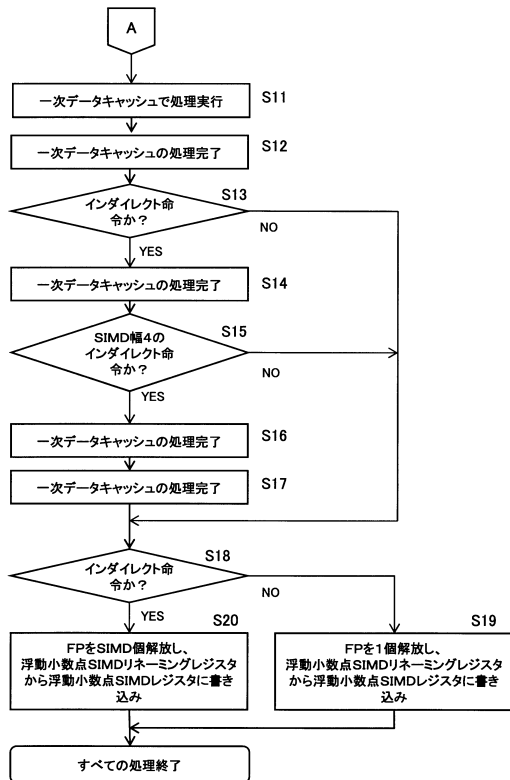
【図 7】



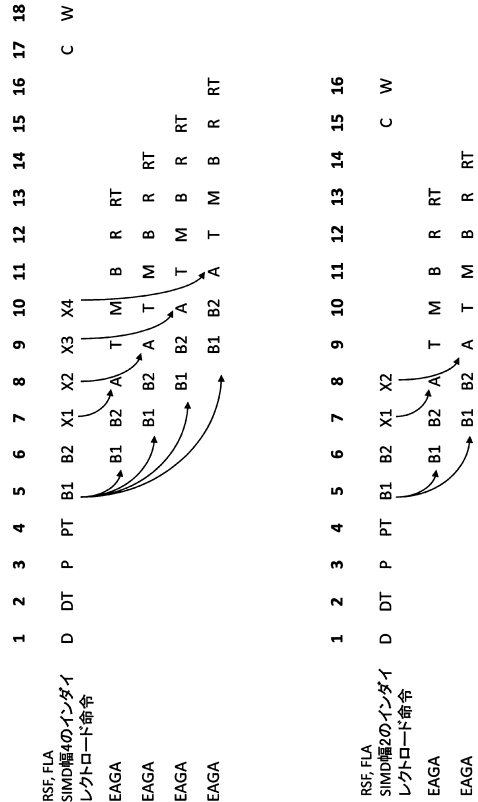
【図 8】



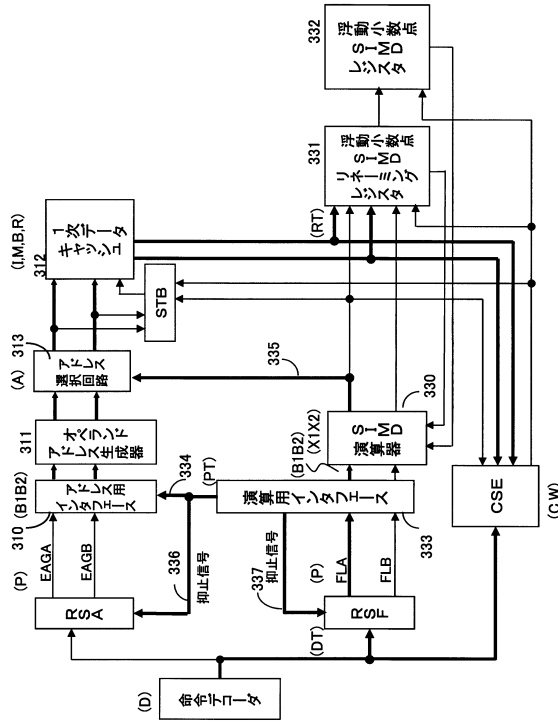
【図 9】



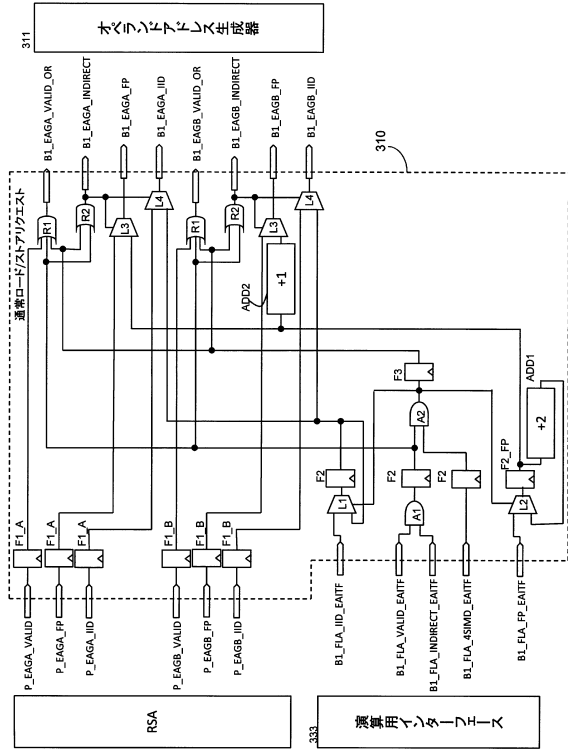
【図 10】



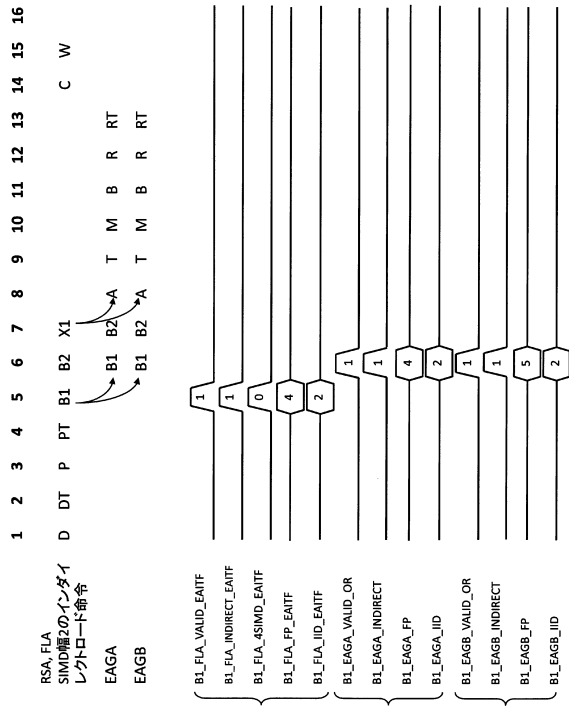
【図 1 1】



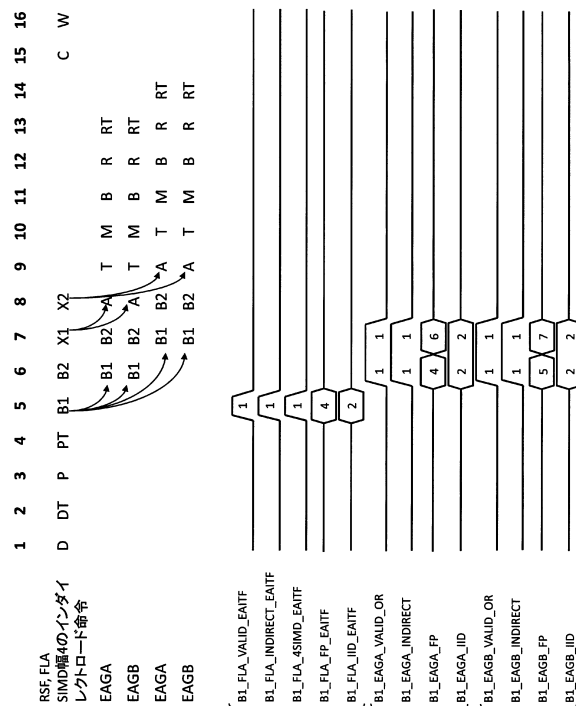
【図 1 2】



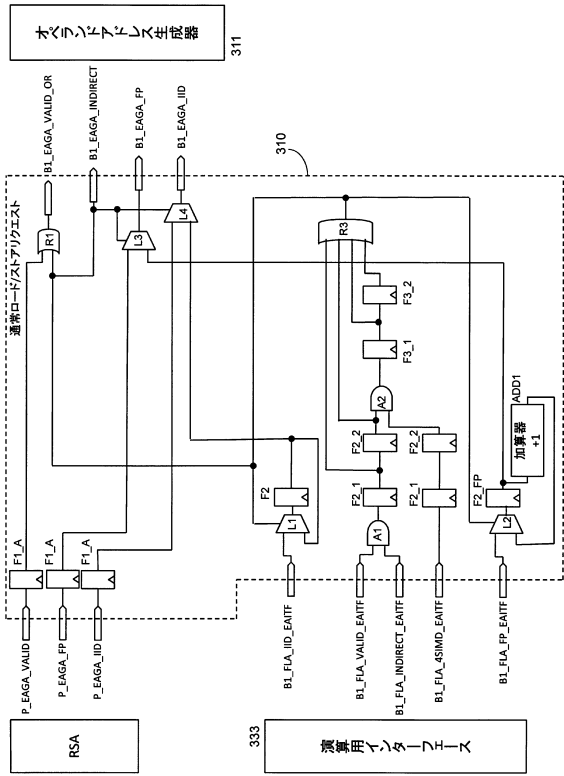
【図 1 3】



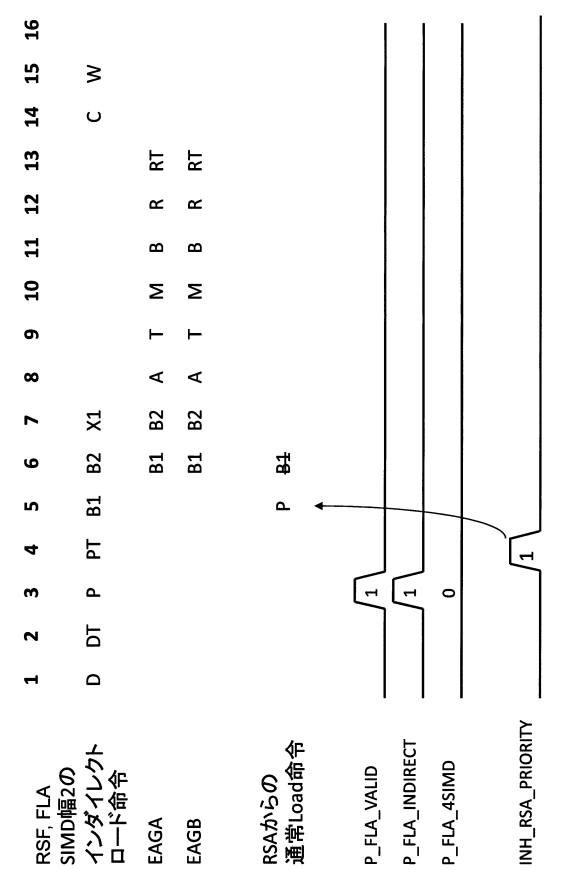
【図 1 4】



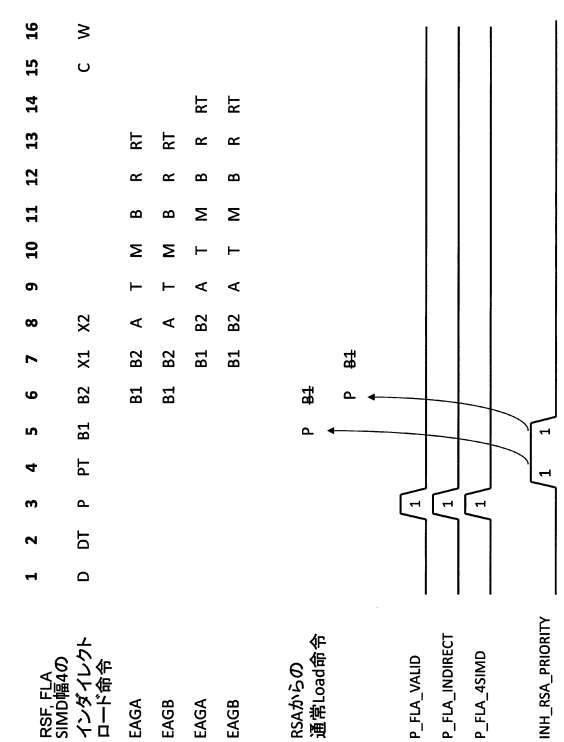
【図 15】



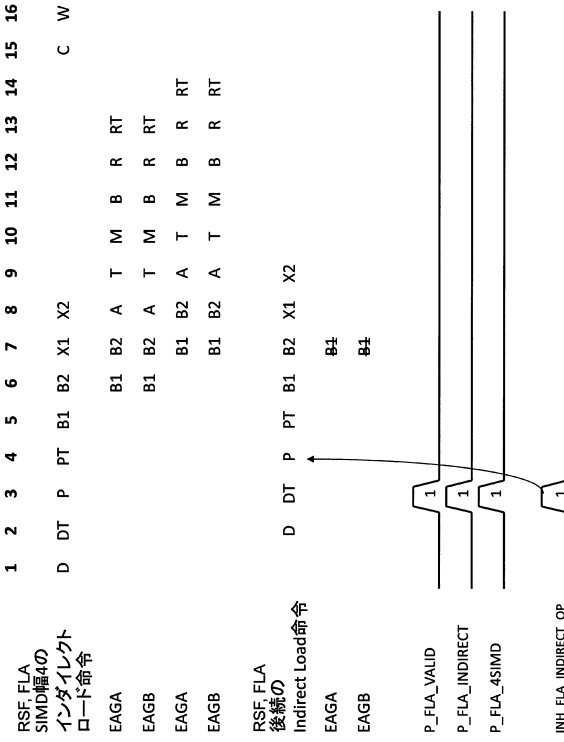
【図 16】



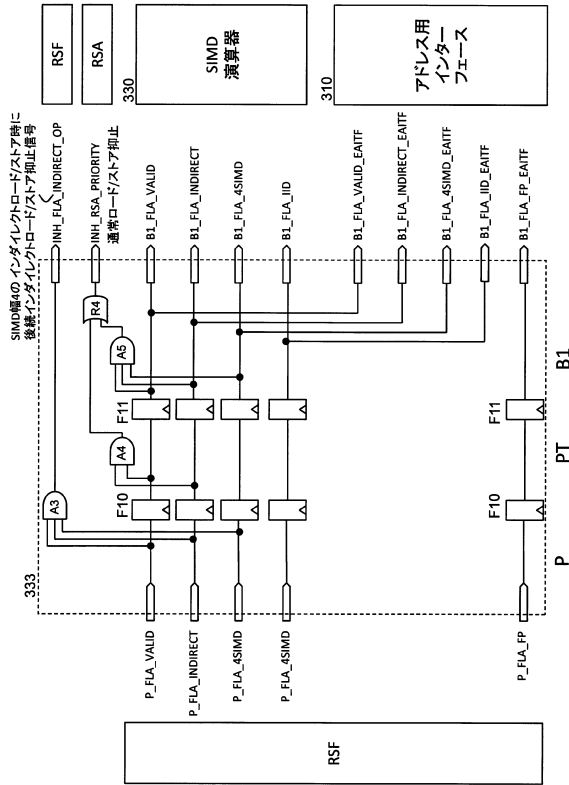
【図 17】



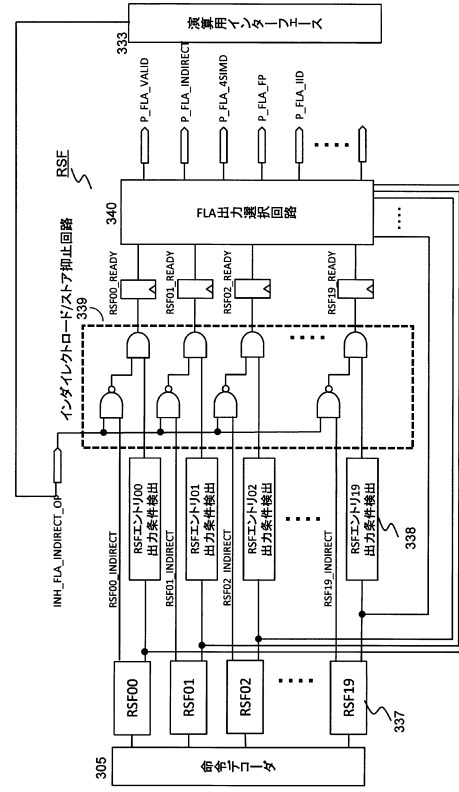
【図 18】



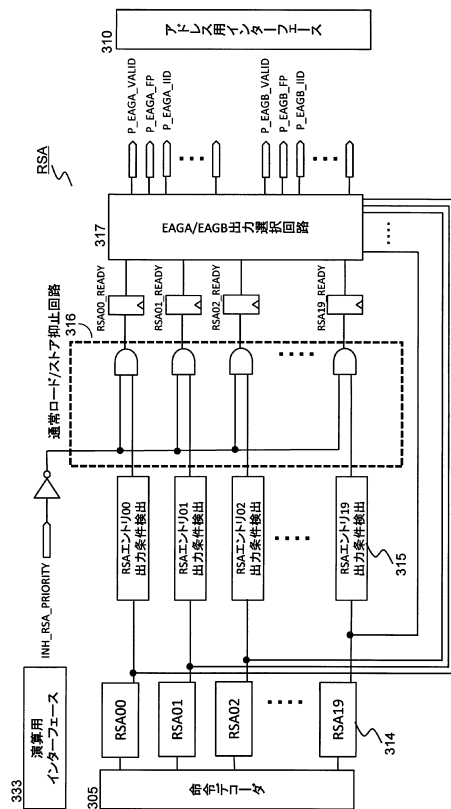
【図 19】



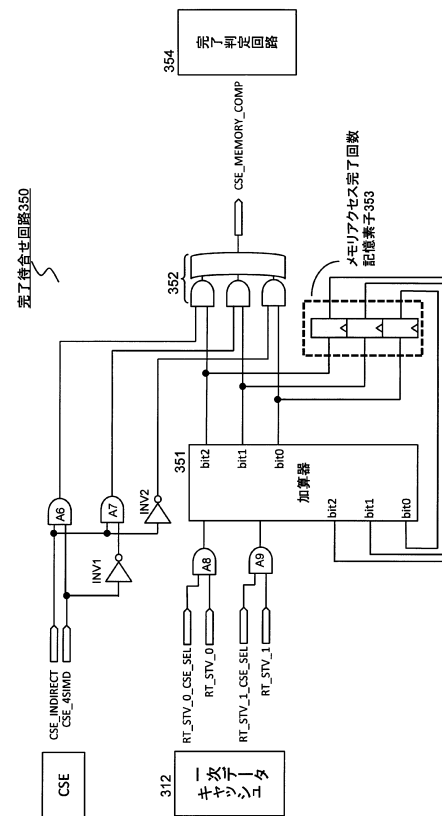
【図 20】



【図 21】



【図 22】



フロントページの続き

(72)発明者 秋月 康伸

神奈川県川崎市中原区上小田中4丁目1番1号 富士通株式会社内

審査官 井上 宏一

(56)参考文献 米国特許出願公開第2012/0060016(US, A1)

特開2012-150634(JP, A)

吉田 利雄, 外3名, 「スーパーコンピュータ「京」のCPU SPARC64 VIIIfx」, FUJITSU, 富士通株式会社, 2012年 5月10日, 第63巻, 第3号, pp.254-259

(58)調査した分野(Int.Cl., DB名)

G06F 9/30 - 9/42