

(19) 日本国特許庁(JP)

(12) 特 許 公 報(B2)

(11) 特許番号

特許第4015411号  
(P4015411)

(45) 発行日 平成19年11月28日(2007.11.28)

(24) 登録日 平成19年9月21日(2007.9.21)

(51) Int. Cl.

F I

G 0 6 F 7/00 (2006.01)

G 0 6 F 7/00 2 0 1 A

G 0 6 F 9/30 (2006.01)

G 0 6 F 9/30 3 7 0

請求項の数 4 (全 25 頁)

(21) 出願番号 特願2001-379549 (P2001-379549)  
 (22) 出願日 平成13年12月13日(2001.12.13)  
 (65) 公開番号 特開2003-177911 (P2003-177911A)  
 (43) 公開日 平成15年6月27日(2003.6.27)  
 審査請求日 平成16年12月10日(2004.12.10)

(73) 特許権者 503121103  
 株式会社ルネサステクノロジ  
 東京都千代田区大手町二丁目6番2号  
 (74) 代理人 110000350  
 ポレール特許業務法人  
 (74) 代理人 100068504  
 弁理士 小川 勝男  
 (72) 発明者 山田 哲也  
 東京都国分寺市東恋ヶ窪一丁目280番地  
 株式会社日立製作所 中央研究所内  
 (72) 発明者 荒川 文男  
 東京都国分寺市東恋ヶ窪一丁目280番地  
 株式会社日立製作所 中央研究所内

審査官 石井 茂和

最終頁に続く

(54) 【発明の名称】 演算装置及びその演算装置を用いた情報処理装置

(57) 【特許請求の範囲】

【請求項1】

入力される第1及び第2のデータをそれぞれ上位側と下位側に分割し、  
 前記第1及び第2のデータに上位側と下位側の有効の情報を有するサイズ情報を有し、  
 前記第1及び第2のデータの上位側と、下位側と、サイズ情報とを保持する記憶手段と  
 、  
 前記第2のデータを反転する反転手段と、  
 前記第2の上位側のデータの反転を抑止する反転抑止手段と、  
 演算の制御を行う制御手段と、  
 前記第1のデータの下位側と、前記第2のデータの下位側の反転手段出力と、前記制御  
 手段の制御信号とを入力とする第1の演算手段と、  
 前記第1及び第2の下位側のデータの最上位ビットを上位側のデータとして使用するた  
 めの論理手段と、  
 前記論理手段出力に対し、前記第1の上位側のデータと、反転抑止手段とを選択する上  
 位側入力選択手段と、  
 前記第1の演算手段の桁上げと、前記第1及び第2のデータのサイズ情報と、前記制御  
 手段の制御信号とを入力とする桁上げ生成手段と、  
 前記桁上げ生成手段出力の桁上げを記憶する桁上げ記憶手段と、前記上位側入力選択手  
 段の出力と前記桁上げ生成手段出力とを入力とする第2の演算手段と、  
 前記第1及び第2のデータのサイズ情報から結果のサイズ情報を生成するサイズ情報生

10

20

成手段と、  
から構成され、

前記第 1 の演算手段から演算結果の下位側と、前記第 2 の演算手段から演算結果の上位側とを生成することを特徴とする演算装置。

【請求項 2】

入力される第 1 及び第 2 のデータをそれぞれ上位側と下位側に分割し、  
前記第 1 及び第 2 のデータに上位側と下位側の有効の情報を有するサイズ情報を有し、  
前記第 1 及び第 2 のデータの上位側と、下位側と、サイズ情報とを保持する記憶手段と

、  
前記第 2 のデータを反転する反転手段と、 10  
前記第 2 の上位側のデータの反転を抑止する反転抑止手段と、  
演算の制御を行う制御手段と、  
前記第 1 のデータの下位側と、前記第 2 のデータの下位側の反転手段出力と、前記制御手段の制御信号とを入力とする第 1 の演算手段と、  
前記第 1 の演算手段の桁上げと、前記第 1 及び第 2 のデータのサイズ情報と、前記制御手段の制御信号とを入力とし、桁上げを生成する桁上げ生成手段と、  
前記第 1 及び第 2 の下位側のデータの最上位ビットと前記第 1 及び第 2 のデータのサイズ情報と前記桁上げ生成手段出力の桁上げと、前記第 1 の上位側のデータと、  
反転抑止手段の出力とを入力とする第 2 の演算手段と、  
前記第 1 の上位側のデータと、反転抑止手段と、前記桁上げ生成手段出力の桁上げとを 20  
入力とする第 3 の演算手段と、  
前記桁上げ生成手段出力の桁上げを記憶する桁上げ記憶手段と、  
前記第 2 及び第 3 の演算手段から結果を選択する上位演算結果選択手段と、  
前記第 1 及び第 2 のデータのサイズ情報から結果のサイズ情報を生成するサイズ情報生成手段と、から構成され、  
前記第 1 の演算手段から演算結果の下位側と、前記上位演算結果選択手段から演算結果の上位側とを生成することを特徴とする演算装置。

【請求項 3】

入力される第 1 及び第 2 のデータをそれぞれ上位側と下位側に分割し、  
前記第 1 及び第 2 のデータに上位側及び下位側の有効の情報を有するサイズ情報を有し 30  
、  
前記第 1 及び第 2 のデータの上位側及び下位側と、前記サイズ情報とを保持する記憶手段と、  
前記第 2 のデータを反転する反転手段と、  
前記第 2 の下位側のデータの反転を抑止する反転抑止手段と、  
演算の制御を行う制御手段と、  
前記制御手段の制御信号を保持する制御信号記憶手段と、  
前記制御手段の制御信号と前記制御信号記憶手段出力とを入力とする第 1 の選択手段と  
、  
前記第 1 のデータの下位側と、前記反転抑止手段の出力と、 40  
前記第 1 の選択手段出力とを入力とする第 1 の演算手段と、  
前記第 1 及び第 2 の下位側のデータと、前記第 1 のデータのサイズ情報とを入力とする第 2 の選択手段と、  
前記第 1 及び第 2 のデータのサイズ情報を入力として、前記第 1 の演算手段と前記第 2 の選択手段を選択する第 3 の選択手段と、  
前記第 1 の演算手段の桁上げと、前記第 1 及び第 2 のデータのサイズ情報と、前記制御手段の制御信号とを入力として、桁上げを生成する桁上げ生成手段と、  
前記第 1 の上位側のデータと、前記第 2 のデータの上位側の反転手段出力と、前記桁上げ生成手段出力の桁上げとを入力とする第 2 の演算手段と、  
前記第 1 及び第 2 のデータのサイズ情報から結果のサイズ情報を生成するサイズ情報生 50

成手段と、  
から構成され、

前記第3の選択手段から演算結果の下位側と、前記第2の演算手段から演算結果の上位側とを生成することを特徴とする演算装置。

【請求項4】

請求項1～3のいずれか1項に記載の演算装置と、

バス上のデータの入力を行う際に、サイズ情報を付加するサイズ情報付加手段と、

前記バス上のデータの出力を行う際に、前記サイズ情報をデータに反映するためにデータを変更する論理手段を持つデータ変更論理手段と、

前記演算装置と、前記サイズ情報付加手段と、前記データ変更論理手段とを接続して、データの記憶とサイズ情報の更新を行うサイズ情報更新手段と、  
から構成されることを特徴とする情報処理装置。 10

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明は、マイクロプロセッサやデジタルシグナルプロセッサ等に使用される演算装置およびその演算装置を用いた情報処理装置に関する。

【0002】

【従来の技術】

近年のマイクロプロセッサでは、レジスタファイルや演算器群、それらを結ぶ内部バスを同ビット幅とし、制御論理を簡単化することで、ハードウェアの複雑化を避け、大幅な周波数向上を遂げてきた。一方、機器組み込み向けアプリケーションプログラムでは、コストに直接影響するメモリ容量を削減するために、必要最低限のビット幅のデータを使用することが多い。 20

【0003】

そこで、メモリからのロードデータや即値の有効ビット幅が演算器やレジスタファイルのビット幅より小さい場合、演算器やレジスタファイルのビット幅に合うよう符号拡張される。例えば、画像情報で有効ビット幅が8ビットのとき、演算器ビット幅が32ビットであれば、上位24ビットを符号拡張し、32ビットデータとして演算される。有効ビット8ビットのみ動作させる場合に対し、上位24ビット分の電力が多く消費される。 30

【0004】

このため、消費電力を抑えるために演算器を有効ビット幅で動作させるいくつかの方法が知られている。

【0005】

例えば、特開平7-248895号公報（従来例1）に示されるように、Nビットの演算器を上位側の（N-K）ビットと下位側Kビットに分割し、入力データがKビットのみ有効である場合は、演算器の入力データの（N-K）ビットを“0”（論理レベル）とし、キャリー伝搬を抑止することで、上位側の演算器を停止させる。

【0006】

また、特開平7-271554号公報（従来例2）に示されるように、入力データの上位側のビットを“0”とする代わりに、上位側のラッチ更新を抑止することによって同様の低消費電力化が行われる。この場合、データの有効ビット幅の制御は、低消費電力のためのモードビットを参照して行っている。 40

【0007】

別の方法として、特開2000-112712号公報（従来例3）に示されるように、入力データから有効ビット幅を検出する方法がある。例えば8ビット加算器で、入力データから有効桁をチェックし、下位4ビットのみが有効とすると、上位側のビットを“0”に固定する。そして、加算結果は下位4ビットと下位の最上位ビットの符号拡張から得られる。但し、下位桁から上位桁へのキャリー伝搬は抑止されていない。

【0008】

このように、低消費電力のために演算器の動作ビット数を減少する方法がいくつか知られている。すなわち、従来の演算器の制御方法として、キャリー伝搬抑止に加え、  
制御方法 A : 上位側加算器の入力を“0”固定する方法(従来例1や従来例3に相当)

、  
制御方法 B : 上位側ラッチ更新抑止による上位側データを固定する方法(従来例2に相当)、が知られている。

ここで、キャリー伝搬抑止とは、下位側のみ有効の場合に、上位側にキャリー“0”を出力することを指す。

【0009】

また、従来の演算器の制御単位としては、

制御単位 A : 特殊レジスタに設定するモードビットを参照する方法、

制御単位 B : 命令をデコードし、命令単位で制御する方法、

制御単位 C : データによって制御する方法、が知られている。

【0010】

【発明が解決しようとする課題】

しかしながら、前述した上記の従来の演算器の制御方法 A と B によれば、次のような問題点がある。

上位側加算器の入力を“0”固定する制御方法 A の場合、上位側を使用しない下位桁のみ有効命令であっても、最初のサイクルの電力消費は抑止できないという問題点がある。なお、2 サイクル目以降、下位桁のみ有効な命令が連続する場合には電力が消費されない。  
上位側ラッチ更新抑止による上位側データを固定する制御方法 B の場合、上位側を使用しない下位桁のみ有効命令であっても、直前の命令で下位加算器から上位加算器へのキャリー信号が“1”(論理レベル)であったとき、最初のサイクルの電力消費を抑止することができないという問題点がある。下位桁のみ有効な命令では、常にキャリー信号として“0”が供給されるためである。なお、2 サイクル目以降、下位桁のみ有効な命令が連続する場合には電力が消費されない。

【0011】

また、前述した従来の演算器の制御単位 A , B , C によれば、次のような難点がある。

モードビットを参照する制御単位 A の制御方法の場合、実現は最も容易であるが、モードビット設定のためのオーバーヘッドがあるという点である。モードビットを設定するためには、モードビットを設定するための命令を実行する必要がある。

命令単位で制御する制御単位 B の場合、各命令を制御部にてデコードし、必要ビット幅の制御を出力する。モードビット参照の制御単位 A の方法に比べれば、命令単位で制御が行えるので電力削減の効果が大きい。演算器を最大幅使用すると定義した命令では、データの有効桁によらず常に演算器は最大幅で動作するという問題がある。

データ単位で制御する制御単位 C の制御方法が、最も電力削減効果が高い方法である。データの有効桁が命令の有効桁より小さい場合は、命令で定義した演算器の使用幅より小さい幅でしか動作しないという利点がある。実現方法は入力データをチェックし、符号拡張部分を調べ、最小のビット幅で演算させればよい。

【0012】

ところで、CPU (Central Processing Unit) あるいは DSP (Digital Signal Processor) の電力としては、演算器のみではなく、レジスタファイルやレジスタファイルと演算器が接続される内部バスで消費される電力も無視できない。そして、レジスタファイルの上位ビットが固定された場合、レジスタファイルの上位ビットの電力が消費しない利点がある。また、内部バスにおいても、上位ビットが固定された場合、内部バスの上位ビットの電力が消費しない利点がある。

【0013】

そこで、本発明の目的は、下位桁のみ有効な命令が実行されるとき、最初のサイクルの電力消費をも抑止することができる演算装置を提供することである。

【0014】

10

20

30

40

50

また、この本発明に係る演算装置を用いて、データ単位で演算器のみでなく、CPUあるいはDSP全体の消費電力を削減することができる情報装置を提供することも本発明の目的の一つである。

【0015】

【課題を解決するための手段】

本発明に係る演算装置は、入力される第1及び第2のデータをそれぞれ上位側と下位側に分割し、

前記第1及び第2のデータに上位側と下位側の有効の情報を有するサイズ情報を有し、

前記第1及び第2のデータの上位側と、下位側と、サイズ情報とを保持する記憶手段と、

前記第2のデータを反転する反転手段と、

前記第2の上位側のデータの反転を抑止する反転抑止手段と、

演算の制御を行う制御手段と、

前記第1のデータの下位側と、前記第2のデータの下位側の反転手段出力と、前記制御手段の制御信号とを入力とする第1の演算手段と、

前記第1及び第2の下位側のデータの最上位ビットを上位側のデータとして使用するための論理手段と、

前記論理手段出力に対し、前記第1の上位側のデータと、反転抑止手段とを選択する上位側入力選択手段と、

前記桁上げ記憶手段と、前記第1の演算手段の桁上げと、前記第1及び第2のデータのサイズ情報と、前記制御手段の制御信号とを入力とする前記第2の演算手段の入力の桁上げを生成する桁上げ生成手段と、

前記桁上げ生成手段出力の桁上げを記憶する桁上げ記憶手段と、

前記上位側入力選択手段の出力と前記桁上げ生成手段出力とを入力とする第2の演算手段と、

前記第1及び第2のデータのサイズ情報から結果のサイズ情報を生成するサイズ情報生成手段と、

から構成され、

前記第1の演算手段から演算結果の下位側と、前記第2の演算手段から演算結果の上位側とを生成することを特徴とするものである。

【0016】

また、本発明に係る演算装置は、入力される第1及び第2のデータをそれぞれ上位側と下位側に分割し、

前記第1及び第2のデータに上位側と下位側の有効の情報を有するサイズ情報を有し、

前記第1及び第2のデータの上位側と、下位側と、サイズ情報とを保持する記憶手段と、

前記第2のデータを反転する反転手段と、

前記第2の上位側のデータの反転を抑止する反転抑止手段と、

演算の制御を行う制御手段と、

前記第1のデータの下位側と、前記第2のデータの下位側の反転手段出力と、

前記制御手段の制御信号とを入力とする第1の演算手段と、

前記第1の演算手段の桁上げと、前記第1及び第2のデータのサイズ情報と、

前記制御手段の制御信号とを入力とし、桁上げを生成する桁上げ生成手段と、

前記第1及び第2の下位側のデータの最上位ビットと前記第1及び第2のデータのサイズ情報と前記桁上げ生成手段出力の桁上げを入力とする第2の演算手段と、

前記第1の上位側のデータと、反転抑止手段と、前記桁上げ生成手段出力の桁上げとを入力とする第3の演算手段と、

前記桁上げ生成手段出力の桁上げを記憶する桁上げ記憶手段と、

前記第2及び第3の演算手段から結果を選択する上位演算結果選択手段と、

前記第1及び第2のデータのサイズ情報から結果のサイズ情報を生成するサイズ情報生成手段と、

から構成され、

10

20

30

40

50

前記第 1 の演算手段から演算結果の下位側と、前記上位演算結果選択手段から演算結果の上位側とを生成する構成としてもよい。

【 0 0 1 7 】

また、本発明に係る演算装置は、入力される第 1 及び第 2 のデータをそれぞれ上位側と下位側に分割し、

前記第 1 及び第 2 のデータに上位側及び下位側の有効の情報を有するサイズ情報を有し、前記第 1 及び第 2 のデータの上位側及び下位側と、前記サイズ情報とを保持する記憶手段と、

前記第 2 のデータを反転する反転手段と、

前記第 2 の上位側のデータの反転を抑止する反転抑止手段と、

10

演算の制御を行う制御手段と、

前記制御手段の制御信号を保持する制御信号記憶手段と、

前記制御手段の制御信号と前記制御信号記憶手段出力を入力とする第 1 の選択手段と、

前記第 1 のデータの下位側と、前記第 2 のデータの下位側の反転手段出力と、

前記第 1 の選択手段出力とを入力とする第 1 の演算手段と、

前記第 1 及び第 2 の下位側のデータと、前記第 1 のデータのサイズ情報とを入力とする第 2 の選択手段と、

前記第 1 及び第 2 のデータのサイズ情報を入力として、前記第 1 の演算手段と前記第 2 の選択手段を選択する第 3 の選択手段と、

前記第 1 の演算手段の桁上げと、前記第 1 及び第 2 のデータのサイズ情報と、

20

前記制御手段の制御信号とを入力として、桁上げを生成する桁上げ生成手段と、

前記第 1 の上位側のデータと、前記反転抑止手段の出力と、前記桁上げ生成手段出力の桁上げとを入力とする第 2 の演算手段と、

前記第 1 及び第 2 のデータのサイズ情報から結果のサイズ情報を生成するサイズ情報生成手段と、

から構成され、

前記第 3 の選択手段から演算結果の下位側と、前記第 2 の演算手段から演算結果の上位側とを生成する構成であってもよい。

【 0 0 1 8 】

また、本発明に係る情報処理装置すなわちマイクロプロセッサやデジタルシグナルプロセッサは、上記のいずれかの演算装置と、

30

バス上のデータの入力を行う際に、サイズ情報を付加するサイズ情報付加手段と、

前記バス上のデータの出力を行う際に、前記サイズ情報をデータに反映するためにデータを変更する論理手段を持つデータ変更論理手段と、

前記演算装置と、前記サイズ情報付加手段と、前記データ変更論理手段とを接続して、データの記憶とサイズ情報の更新を行うサイズ情報更新手段と、

から構成されることを特徴とするものである。

【 0 0 1 9 】

【 発明の実施の形態 】

以下、本発明の好適な実施の形態について、添付図面を参照しながら詳細に説明する。

40

【 0 0 2 0 】

< 実施の形態 1 >

図 1 は、本発明に係る演算装置の第 1 の実施の形態を示す加減算回路のブロック図である。加減算回路の 2 つの入力データを、データ 1 及びデータ 2 とする。

図 1 において、参照符号 1 0 1 及び 1 0 2 はデータ 1 を上位側と下位側に 2 分する際の上位レジスタと下位レジスタである。例としてデータ 1 のビット幅を 3 2 ビットとすると、レジスタ 1 0 1 は 1 6 ビットの上位側、レジスタ 1 0 2 は下位側に相当する。データ 1 には、サイズ情報 S 1 を示すサイズビットレジスタ 1 0 3 がある。サイズ情報 S 1 はデータと常に付随する情報として扱われる。同様に、もう一つの入力データであるデータ 2 を上位側と下位側に 2 分する際の上位レジスタ 1 0 4 及び下位レジスタ 1 0 5 と、データ 2 の

50

サイズ情報 S 2 を示すビットレジスタ 1 0 6 がある。上記レジスタ 1 0 1 ~ 1 0 6 は全てフリップフロップで構成され、クロック (clock) 1 の立ち上がりで動作する。

【 0 0 2 1 】

1 0 7 及び 1 0 8 はそれぞれデータ 1、データ 2 の上位側、下位側の加算器である。1 0 9 は制御部であり、演算の減算を示す S U B 信号と並列演算を示す S M D 信号を出力し、演算の制御を行う。また、図 1 では省略してあるが、制御部 1 0 9 は上記レジスタ 1 0 1 ~ 1 0 6 のイネーブル信号も命令単位で出力する。

【 0 0 2 2 】

1 1 0 及び 1 1 1 は、減算を行う際に使用するデータ 2 と S U B 信号を入力とする E O R (排他的論理和) ゲートであり、それぞれデータ 2 の上位側、下位側に対応する。1 1 2 はデータ 2 のサイズビット S 2 に従って S U B 信号をマスクするための A N D ゲートである。

【 0 0 2 3 】

1 1 3 はキャリー制御部であり、データ 1 とデータ 2 のサイズビットレジスタ 1 0 3 , 1 0 6 に格納されたサイズ情報 S 1 , S 2 と S U B 信号及び S M D 信号を入力とし、下位側の加算器 1 0 8 から上位側の加算器 1 0 7 への桁上げ信号の制御を行う。キャリー制御用のために、キャリー保持レジスタ 1 1 4 により以前のキャリー C を保持する。サイズ情報が異なるデータの演算を行うときは、サイズを合わせるためにビット長の小さいデータを符号拡張する必要がある。符号拡張制御部 1 1 5 にてサイズビットレジスタ 1 0 3 , 1 0 6 を入力として符号拡張制御を行う。加算器 1 0 7 , 1 0 8 の出力ライン 1 1 6 , 1 1 7 に、データ上位側及び下位側の演算結果データ D 3 H 及び D 3 L をそれぞれ出力する。演算結果に付随する情報として、信号ライン 1 1 8 に結果サイズ情報 S 3 を出力する。

【 0 0 2 4 】

ここで、図 1 に示す本実施の形態における加減算回路の動作を説明する前に、表 1 と図 2 を参照して、加減算の方法を説明する。データ 1 とデータ 2 は、3 2 ビット符号付整数とし、上位 1 6 ビットをそれぞれ D 1 H , D 2 H と表し、下位 1 6 ビットをそれぞれ D 1 L , D 2 L と表す。サイズビットは 1 ビットであり、“ 1 ” のとき 3 2 ビット、“ 0 ” のとき 1 6 ビットを示す。

【 0 0 2 5 】

【表 1】

表 1

| 命令    | 内容                  | CIU | CIL |
|-------|---------------------|-----|-----|
| ADD32 | 32+32, 32+16, 16+32 | CO  | 0   |
| ADD16 | 16+16               | C   | 0   |
| SUB32 | 32-32, 32-16, 16-32 | CO  | 1   |
| SUB16 | 16-16               | C   | 1   |
| ADDS  | 16+16(SIMD)         | 0   | 0   |
| SUBS  | 16-16(SIMD)         | 1   | 1   |

表 1 には、本実施の形態における加減算回路で実行される命令と、命令の内容と、上位側加算器 1 0 7 へのキャリーイン C I U と、下位側加算器 1 0 8 へのキャリーイン C I L とを示している。減算命令は“データ 1” - “データ 2”のみ用意されているが、“データ 2” - “データ 1”の拡張は容易であり、説明を簡単にするために、“データ 1” - “データ 2”を実現する回路となっている。

## 【 0 0 2 6 】

表 1 において、C O とは下位側加算器 1 0 8 のキャリーアウトを示し、C とはキャリー保持レジスタ 1 1 4 に保持されたキャリーを示す。A D D 3 2 では少なくとも一方が 3 2 ビットのデータでの加算命令を示し、サイズ情報の組合せにより、 $32 + 32$ 、 $16 + 32$ 、 $32 + 16$  の 3 通りの演算が起こり得る。

S U B 3 2 は、A D D 3 2 の加算を減算に置換えた命令である。A D D 1 6 と S U B 1 6 とはそれぞれ、2 つの下位 1 6 ビットでの加算と減算を行う命令である。A D D S と S U B S とはそれぞれ、3 2 ビットデータで上位 1 6 ビット、下位 1 6 ビットを並列に加算と減算を行う命令である。

## 【 0 0 2 7 】

10

図 2 は、表 1 の命令のうち 4 通りの加減算の例を示している。

(1) データ 1 が 3 2 ビット、データ 2 が 1 6 ビットの加算 (A D D 3 2) の場合：

データ 1 は 3 2 ビットなので、サイズビットが “ 1 ”、データ 2 は 1 6 ビットなので、サイズビットは “ 0 ” である。データ幅が異なる場合、データのビット長が大きいデータとビット長を合わせるために、ビット長が小さいデータは符号拡張を行う。従って、データ 2 は符号拡張される。3 2 ビット加算なので下位側加算のキャリーイン C I L は “ 0 ” であり、上位側加算のキャリーイン C I U は下位側からの桁上げ (キャリーアウト) C O である。結果は 3 2 ビットであり、サイズは “ 1 ” となる。

## 【 0 0 2 8 】

(2) データ 1 が 3 2 ビット、データ 2 が 1 6 ビットの減算 (S U B 3 2) の場合：

20

データ 1 は 3 2 ビットなので、サイズビットが “ 1 ”、データ 2 は 1 6 ビットなので、サイズビットは “ 0 ” である。3 2 ビット減算なので、データ 2 は符号拡張と反転を行い、下位側加算のキャリーイン C I L は “ 1 ” であり、上位側加算のキャリーイン C I U は下位側からの桁上げ C O である。結果は、3 2 ビットであり、サイズは “ 1 ” となる。

## 【 0 0 2 9 】

(3) データ 1、データ 2 が 1 6 ビットの加算 (A D D 1 6) の場合：

データ 1、データ 2 は 1 6 ビットなのでサイズビットは “ 0 ” である。上位側は演算に無関係なため、ドントケアである。そこで、上位側は前サイクルと同じ結果となるようにする。入力データを前サイクルと同じデータにし、上位側加算のキャリーイン C I U を前サイクルと同じキャリー C とすればよい。下位側加算のキャリーイン C I L を “ 0 ” とすれば、結果の下位 1 6 ビットは正しい演算結果が得られる。結果のサイズは “ 0 ” となる。

30

## 【 0 0 3 0 】

(4) データ 1、データ 2 が 1 6 ビットの並列減算 (S U B S 1 6) の場合：

データ 1、データ 2 は 3 2 ビットなのでサイズビットは “ 1 ” である。上位側と下位側を並列に演算させるため、データ 2 の入力を反転し、上位側加算のキャリーイン C I U、下位側加算のキャリーイン C I L をどちらも “ 1 ” とする。結果は 3 2 ビットであり、サイズは “ 1 ” となる。

## 【 0 0 3 1 】

次に、図 1 の加減算器の動作を図 2 の (2) の S U B 3 2 (3 2 ビット - 1 6 ビット) を例にとり説明する。データ 1 入力 は 3 2 ビットなので、データ 1 のサイズビットレジスタ 1 0 3 に “ 1 ”、データ 1 の上位レジスタ 1 0 1 と下位レジスタ 1 0 2 にそれぞれ D 1 H と D 1 L が格納される。

40

## 【 0 0 3 2 】

データ 2 入力 は 1 6 ビットなので、データ 2 のサイズビットレジスタ 1 0 6 に “ 0 ”、データ 2 の下位レジスタ 1 0 5 に D 2 L が格納され、上位レジスタ 1 0 4 は変化しない。制御部 1 0 9 から減算用の制御信号である S U B 信号には “ 1 ” が供給される。S U B 信号により、データ 2 の下位レジスタ 1 0 5 に格納されたデータの出力 D 2 L が E O R ゲート 1 1 1 により反転されて下位側の加算器 1 0 8 に与えられると共に、下位側の加算器 1 0 8 にキャリーイン C I L として S U B 信号の “ 1 ” が与えられる。下位側の加算器 1 0 8 において、データ 1 の下位レジスタ 1 0 2 に格納された D 1 L と、データ 2 の下位レジス

50



タ 1 0 5 の反転出力 D 2 L ( は、反転記号の代用である )、キャリーイン C I L = “ 1 ” が入力として与えられ、加算した後、下位側加算器 1 0 8 の結果 D 3 L が出力ライン 1 1 7 に出力される。下位側加算器 1 0 8 のキャリーアウト C O は、キャリー制御部 1 1 3 に入力されるが、スルー出力で上位側加算器 1 0 7 のキャリーイン C I U として入力される。データ 1 とデータ 2 のビット幅を合わせるために、データ 2 は下位側の最上位ビット ( M S B : Most Significant Bit ) が符号拡張され、符号拡張制御部 1 1 5 に入力される。E O R ゲート 1 1 1 の出力の M S B を使用するため、データ 2 の下位レジスタ 1 0 5 の M S B に対し、反転された符号が拡張されている。符号拡張制御部 1 1 5 では、サイズ情報としてデータ 1、データ 2 のサイズビットレジスタ 1 0 3、1 0 6 と上位側データ D 1 H, D 2 H と符号拡張された下位側データ D 1 L, D 2 L とを入力し、データ幅が同じ場合は上位側データを、データ幅が異なる場合は、データ幅が小さい方のセレクトは符号拡張下位側データを選択する。

10

#### 【 0 0 3 3 】

図 2 の ( 2 ) の例では、データ 1 側はデータ 1 の上位レジスタ 1 0 1 の出力 D 1 H を選択し、データ 2 側は符号拡張下位側データを選択して、それぞれ上位側加算器 1 0 7 に入力される。上位側加算器で加算し、結果を上位側データ D 3 H として出力ライン 1 1 6 に出力する。信号ライン 1 1 8 には結果のサイズ S 3 は、入力サイズのどちらかが 3 2 ビットのときは “ 1 ”、どちらも 1 6 ビットデータのときは “ 0 ” となる。図 2 の ( 2 ) の例では結果サイズ S 3 として “ 1 ” を出力する。

#### 【 0 0 3 4 】

20

図 2 の ( 1 ) の A D D 3 2 ( 3 2 ビット + 1 6 ビット ) は上記 S U B 3 2 に対し、制御部 1 0 9 の S U B 信号が “ 0 ” となることにより実行される。下位側加算器 1 0 8 の変更点は、データ 2 の入力とキャリーインであるが、S U B 信号が “ 0 ” になることにより下位側加算器 1 0 8 のデータ 2 の入力はスルーとなり、キャリーイン C I L は “ 0 ” となる。上位側加算器 1 0 7 のデータ 2 側のデータとしては、データ 2 の下位レジスタ 1 0 5 の M S B と同じ値が符号拡張され、供給される。

#### 【 0 0 3 5 】

A D D 3 2 命令と S U B 3 2 命令の上記と異なるデータ幅においては、2 つの入力データが 3 2 ビットのときは、符号拡張の代わりに上位側のデータを使用することにより実現できる。データ 1 のビット幅が 1 6 ビット、データ 2 のビット幅が 3 2 ビットのときは、データ 1 の下位側データ D 1 L の M S B を符号拡張することにより実現できる。

30

#### 【 0 0 3 6 】

図 2 の ( 3 ) の A D D 1 6 ( 1 6 ビット + 1 6 ビット ) は、本実施の形態において、上位側の加算器 1 0 7 のトランジスタのスイッチングを抑止する点で、従来例に対し低消費電力を達成する好適な例である。2 つの入力はどちらも 1 6 ビットであるため、データ 1、データ 2 のサイズビットレジスタ 1 0 3, 1 0 6 は両方とも “ 0 ” である。したがって、データ 1、データ 2 の上位側レジスタ 1 0 1, 1 0 4 は変更されず、前の値を保持する。データ 1、データ 2 の下位側レジスタ 1 0 2, 1 0 5 に入力データが格納される。加算のため、制御部 1 0 9 から S U B 信号は “ 0 ” が出力される。

#### 【 0 0 3 7 】

40

S U B 信号の “ 0 ” に従って、下位側加算器 1 0 8 の入力として、データ 1 の下位レジスタ 1 0 2 の出力 D 1 L と、データ 2 の E O R ゲート 1 1 1 のスルー出力と、キャリーイン C I L の “ 0 ” とが与えられ、加算が実行される。加算結果は 1 6 ビットであり、結果サイズビット S 3 は “ 0 ” となる。

#### 【 0 0 3 8 】

ここで、A D D 1 6 命令においては、演算結果の実行に、上位側の加算器 1 0 7 を使用しない。上位側の加算器のスイッチングを停止することで、消費電力が削減できる。上位側加算器の消費電力を削減する方法を説明する。上位側の加算器 1 0 7 のスイッチングを停止するためには、加算器の 2 つの入力とキャリーインを固定すればよい。固定の方法は次のように行う。

50

## 【0039】

まず、データ1、データ2のサイズビットレジスタ103, 106がどちらも“0”であることから、符号拡張制御部115のセレクトは、H側すなわちデータ1、データ2の上位側D1H, D2Hが選択される。データ1の上位側レジスタ101は前の値が保持されているため固定、データ2側は、データ2のサイズビットレジスタ106が“0”であることからSUB信号がマスクされ、EORゲート111は、前の値を保持するデータ2の上位側レジスタ104のデータD2Hをスルーで出力するため固定であり、上位側加算器107の2つの入力データは固定となる。キャリーイン信号に関しては、並列実行を示すSMD信号が“0”、2つのサイズビットレジスタ103, 106が“0”のため、前のキャリーを保持するキャリー保持レジスタ114の出力Cが上位加算器107のキャリーイン信号CIUとして選択される。キャリー保持レジスタ114の出力が固定値のため、上位加算器107のキャリーイン信号も固定値である。従って、上位加算器107の入力が全て固定され、トランジスタのスイッチングが抑止され、電力を消費しない。

10

## 【0040】

SUB16命令では、SUB信号が“1”となるため、下位側のデータ2とキャリーインCILとが反転される点異なるが、他はADD16命令と同様の処理で実現される。上位側の加算器107に関してはADD16命令と同様に電力を消費しない。

## 【0041】

更に、ADD16命令かSUB16命令が連続するとき、制御部109からのSUB信号は命令に従って変化するが、データ2のサイズビットレジスタ106の出力S2によってANDゲート112でマスクされることにより、常に固定値の入力が上位加算器107に供給され、電力は消費されない。

20

## 【0042】

図2の(4)のSUBS16(“16ビット” - “16ビット”の並列実行)命令は、上位側データと下位側データを次の方法で並列に実行する。下位側の加算器108に、データ1の下位レジスタ102の出力D1Lと、データ2の下位レジスタ105の反転出力D2Lと、SUB信号“1”によるキャリーインCILの入力とが与えられ、16ビット結果を出力する。2入力のデータサイズはどちらも“1”であることから、符号拡張制御部115では、データ1の上位レジスタ101の出力D1Hとデータ2の上位レジスタ104の反転出力D2Hが選択され、上位側加算器107の入力データとなる。上位側加算器のキャリーイン信号として、下位側の加算器108からの桁上げ信号伝搬を抑止する必要があるが、制御部109からの並列実行信号SMD信号により下位側からの桁上げ信号伝搬の抑止を行い、SUB信号により強制的に“1”を上位側加算器107のキャリーイン信号CIUとして与えている。

30

このことにより、下位桁の影響を受けずに並列に16ビット減算を実行することができる。

## 【0043】

ADDS16命令に関しては、SUB信号が“0”となることにより、上位側および下位側の加算器にそれぞれデータ2の上位レジスタ104のスルー出力D2H、下位レジスタ105のスルー出力をデータ2側の入力として与え、キャリーインCIUとして強制的に“0”とすることにより実現できる。

40

## 【0044】

上記のように本実施の形態によれば、データ1、データ2の下位16ビット演算命令が連続するとき、上位加算器107の入力データとキャリーが固定されるため、電力を消費しない。特に、下位16ビット加算命令と下位16ビット減算命令が1命令ずつ交互に実行される場合においても、SUB信号によるデータ2の上位側データのスルーまたは反転がANDゲート112により抑止されるため、上位加算器107の入力データとキャリーが固定され、電力を消費しない。

## 【0045】

また、下位16ビット演算命令と32ビット演算命令が1命令ずつ交互に実行されるとき

50

は、下位 16 ビット演算命令が実行するとき上位加算器 107 の電力消費が行われない。従って、下位 16 ビットのみ有効な命令が実行されるとき、最初のサイクルから電力消費を抑止することができるといえる。

【0046】

従来例の制御方法 A の上位加算器の入力を “0” 固定する方法と、制御方法 B の上位側データを固定する方法すなわち上位レジスタを固定する方法とでは、以下のような場合に電力消費を必要とするのに対し、本実施の形態の加減算回路では上述したように電力が消費されない利点がある。

【0047】

制御方法 A の上位加算器の入力を “0” 固定する方法では、下位 16 ビット演算命令が連続する場合、最初の下位 16 ビット演算命令を実行するとき、上位加算器 107 の電力を消費するが、その後の下位 16 ビット演算命令では上位加算器 107 の電力消費は抑止される。下位 16 ビット演算命令と 32 ビット演算命令が 1 命令ずつ交互に実行される場合、下位 16 ビット命令を実行するとき、必ず上位加算器 107 の入力を “0” にするため、前命令の入力が “0” 以外のときは常に上位加算器 107 の電力を消費する。

10

【0048】

制御方法 B の上位レジスタを固定する方法では、下位 16 ビット演算命令を実行するとき、上位レジスタを固定し、下位加算器から上位加算器 107 へのキャリーを強制的に “0” にする。下位 16 ビット演算命令が連続する場合、最初の下位 16 ビット演算命令を実行するとき、下位加算器 108 から上位加算器 107 へのキャリーを強制的に “0” にするため、前サイクルのキャリーが “1” であると、上位加算器 107 は電力を消費する。

20

【0049】

下位 16 ビット加算命令と下位 16 ビット減算命令が 1 命令ずつ交互に実行する場合、データ 1 とデータ 2 の上位レジスタを固定しても、データ 2 が減算命令では反転されるため、上位加算器 107 のデータ 2 側の入力が常に反転することになり、消費電力を抑止できない。

【0050】

また、下位 16 ビット演算命令と 32 ビット演算命令が 1 命令ずつ交互に実行される場合、32 ビット演算命令実行時に、常に下位加算器 108 から上位加算器 107 へのキャリーが “1” であると、下位 16 ビット演算命令実行時にキャリーを強制的に “0” にするため、下位 16 ビット演算命令においても常に上位加算器 107 の電力を消費する。

30

【0051】

以上のように、下位 16 ビット演算命令が連続する場合、下位 16 ビット演算命令と 32 ビット命令が交互に実行される場合においても、従来に対し消費電力を低く抑えられることを示した。

【0052】

図 1 では、簡単のために加減算器を示したが、論理ゲートを追加することにより、容易に算術論理演算装置 (ALU: Arithmetic and Logic Unit) を実現することができる。

【0053】

図 3 に、図 1 の加減算回路を用いたマイクロプロセッサの例として構成図を示す。

40

【0054】

DRAM や SDRAM 等の外部メモリ 301 は、外部データバス 302 および外部アドレスバス 303 によりマイクロプロセッサ 304 に接続されている。マイクロプロセッサ (MPU) 304 は、CPU 305、および外部とのデータの授受を行うバスインタフェースユニット 306 から構成され、CPU とバスインタフェースユニット間はロードストアバス 307 にて接続される。尚、上記 DRAM や SDRAM 等のメインメモリは外部に設けるのではなく、マイクロプロセッサ 304 と一緒に 1 枚の半導体基板内に内蔵されていてもよい。その場合は、バス 302 や 303 は同一チップ内の内部バスで構成される。CPU 305 は、制御部 308 とデータバス部 309 から構成される。

50

## 【 0 0 5 5 】

データバス部は、データのロードおよびストアを行うロードストアユニット 3 1 0 と、図 1 で示される構成の A L U 3 1 1 と、レジスタファイル 3 1 2 とから構成される。図 1 は加減算器であるが、A L U に容易に機能拡張可能なため、以後 A L U と呼ぶことにする。

## 【 0 0 5 6 】

図 4 にレジスタファイル 3 1 2 の構成例を示す。このレジスタファイル 3 1 2 は、ロードデータと A L U 演算結果のどちらかを選択する入力セクタ 4 0 2 と、この入力セクタ部 4 0 2 で選択されたデータのサイズチェックを行うデータチェック部 4 0 3 と、データを保持するデータ保持部 4 0 4 と、A L U の入力データ D 1 およびデータ D 2 とストアデータをデータ保持部 4 0 4 から選択する出力セクタ 4 0 8 とから構成される。

10

## 【 0 0 5 7 】

データ保持部 4 0 4 はレジスタ数は 8 本で、サイズビットレジスタファイル 4 0 5 と、上位側データレジスタファイル 4 0 6 と、下位側データレジスタファイル 4 0 7 とから構成される。データ保持部 4 0 4 のデータ幅は 3 2 ビットで、上位 1 6 ビット、下位 1 6 ビットである。1 ビットのサイズビットは、“ 0 ” のとき下位 1 6 ビットのみ有効であり、“ 1 ” のとき 3 2 ビットが有効であることを示す。出力セクタ 4 0 8 で、A L U データ D 1 はレジスタ [ 3 : 0 ]、A L U データ D 2 はレジスタ [ 7 : 4 ]、ストアデータはレジスタ [ 7 : 0 ] のいずれかから任意に選択できることとする。

## 【 0 0 5 8 】

図 4 のレジスタファイル 3 1 2 の動作を説明する。ロードデータを A L U のデータ D 1 として読み出すことを考える。図 3 に示した C P U 3 0 5 では、ロードストア部 3 1 0 と A L U 3 1 1 にフォワーディングセクタを備えていないことを前提としているので、図 4 ではロードストア部 3 1 0 と A L U 3 1 1 はレジスタファイル 3 1 2 と接続される。

20

## 【 0 0 5 9 】

まず、入力セクタ 4 0 2 において、制御部 3 0 8 からロードデータを選択するセクタ信号が供給され、ロードデータがデータチェック部 4 0 3 の入力に与えられる。

## 【 0 0 6 0 】

データチェック部 4 0 3 では、上位側データが符号拡張であるかどうかをチェックし、入力のサイズが “ 0 ” か上位側が符号拡張であれば、サイズを強制的に 0 にし、それ以外は “ 1 ” とする。

30

## 【 0 0 6 1 】

データ保存部 4 0 4 では、レジスタ [ 7 : 0 ] のいずれかにデータがライト(書込み)される。例として、レジスタ [ 0 ] にライトされるとすると、サイズビットレジスタ [ 0 ] にサイズビットがライトされ、サイズビットが “ 0 ” のとき下位側データレジスタ 4 0 7 のみライトされ、サイズビットが “ 1 ” のとき上位側と下位側データレジスタ 4 0 6 , 4 0 7 にライトされる。出力セクタ部 4 0 8 において、A L U データ D 1 側のレジスタが [ 3 : 0 ] のいずれかから選択される。

ここでは、[ 0 ] レジスタが選択され、A L U データ D 1 側に [ 0 ] レジスタが出力される。

## 【 0 0 6 2 】

40

図 5 ( a ) に、図 4 のデータチェック部 4 0 3 の構成図を示す。入力および出力は、どちらもサイズビット S と、データ上位側と、データ下位側であり、ビット幅はそれぞれ 1 ビット、1 6 ビット、1 6 ビットである。サイズビットはデータ下位側の M S B とデータ上位側のオール ( A 1 1 ) “ 1 ” 検出 5 0 1、データ下位側の M S B と上位側の A 1 1 “ 0 ” 検出 5 0 2、或いはサイズビットの “ 0 ” 検出を行い、いずれかの条件が成立するとき、サイズビットは “ 0 ” となる。すなわち、入力が 1 6 ビットデータか、入力が 3 2 ビットデータで、かつ、データ上位側が符号拡張のときにサイズビットが “ 0 ” となる。

A 1 1 “ 1 ” 検出 5 0 1 と A 1 1 “ 0 ” 検出 5 0 2 は、図 5 ( b ) , ( c ) に示されるように、それぞれ A N D ツリー、N O R ツリーを構成することで実現できる。同図 ( b ) が A 1 1 “ 1 ” 検出、( c ) が A 1 1 “ 0 ” 検出である。

50

## 【 0 0 6 3 】

図 6 に、図 4 のデータ保持部 4 0 4 のうちレジスタ [ i ] ( 0 i 7 ) の構成図を示す。

サイズビット 6 0 1、データ上位側 6 0 2、データ下位側 6 0 3 は、 1 ( 1 の反転 ) のクロックが供給されるラッチで構成される。ビット幅はそれぞれ 1 ビット、16 ビット、16 ビットである。サイズビット 6 0 1、データ下位側のデータ 6 0 3 のラッチは、レジスタ [ i ] のライトイネーブル信号を表す制御部からのライトイネーブル [ i ] と、 1 のゲーテッドクロックである。データ上位側は、ライトイネーブル [ i ] とゲーテッドクロック 1 に加え、サイズビットとのゲーテッドクロックである。すなわち、データ上位側は、サイズビットが “ 1 ” で、かつ、上位側が符号拡張でなく、イネーブルなデータのときのみライトされる。

10

## 【 0 0 6 4 】

レジスタファイルはビット幅が大きいので、サイズビットが “ 0 ” のときのデータ上位側のライト抑止による電力低減の効果は大きい。データが全て 16 ビットのプログラムを実行した場合、サイズビットが全て “ 0 ” となることから、レジスタファイル部において約 50 % の電力削減が可能である。

## 【 0 0 6 5 】

図 7 に、図 3 のロードストア部 3 1 0 の構成例を示す。ロードストア部 7 0 1 はロード部 7 0 2 とストア部 7 0 5 から成り、ロードデータ、ストアデータはレジスタファイル 3 1 2 と接続される。ロードストアバス 7 0 9 はデータの上位、下位ビットの 32 ビットのビット幅で、CPU 内の内部データ信号はサイズビットと、データ上位、下位ビットとの 33 ビットのビット幅で接続される。ロード出力とストア入力はどちらも 33 ビットである。

20

## 【 0 0 6 6 】

ロード部 7 0 2 に関して、ロードサイズの情報は、制御部 3 0 8 で命令をデコードすることにより得られる。16 ビットロード命令の実行時はサイズビットが “ 0 ”、32 ビットロード命令の実行時はサイズビットが “ 1 ” となる。そして、制御部 3 0 8 によりロードの適切なタイミングに応じて、32 ビットロード命令では上位側ロードイネーブル信号 7 0 3 と下位側ロードイネーブル信号 7 0 4 が “ 1 ”、16 ビットロード命令では下位側ロードイネーブル信号 7 0 4 のみが “ 1 ” となり、データが正しくロードされる。

30

ストア部 7 0 5 に関して、ストア命令の種類とサイズ情報から適正なデータを生成しストアする。

## 【 0 0 6 7 】

32 ビットストア命令でデータのサイズ情報が “ 0 ” のときは、下位データの MSB を符号拡張して生成した 32 ビットデータをストアする必要がある。そこで、ストア用の上位データはサイズビットが “ 0 ” のときは下位データ MSB の符号拡張、サイズビットが “ 1 ” のときは上位データを選択するセレクタ 7 0 6 を用意する。そして、制御部 3 0 8 によりストアの適切なタイミングに応じて、32 ビットストア命令では上位側ストアイネーブル信号 7 0 7 と下位側ストアイネーブル信号 7 0 8 が “ 1 ”、16 ビットストア命令では下位側ストアイネーブル信号 7 0 8 のみが “ 1 ” となり、ストア用に生成されたデータが正しくストアされる。

40

## 【 0 0 6 8 】

以上により、本発明の第 1 の実施の形態として、図 1 の加減算回路 ( ALU ) と、図 4、5、6 のレジスタファイルと、図 7 のロードストア部及びそれらを用いた図 3 に示すマイクロプロセッサとを示した。

## 【 0 0 6 9 】

< 実施の形態 2 >

図 8 に、本発明に係る演算装置の第 2 の実施の形態である加減算回路のブロック図を示す。前述した第 1 の実施の形態の図 1 の加減算回路 ( ALU ) では、下位レジスタを起点とし、MSB を符号拡張し、上位側の加算器で加算するパスがタイミングが最も厳しいクリ

50

ティカルパスである。そこで、そのクリティカルパスを解消するのが、図 8 の回路構成である。図 1 以外は、第 1 の実施の形態と共通である。

#### 【 0 0 7 0 】

図 8 の加減算回路のブロック図を説明する。加減算回路の 2 つの入力データをデータ 1 とデータ 2 とする。図 8 において、参照符号 8 0 1 及び 8 0 2 はデータ 1 を上位側と下位側に 2 分する際の上位レジスタと下位レジスタである。例としてデータ 1 のビット幅を 3 2 ビットとすると、レジスタ 8 0 1 は 1 6 ビットの上位側、レジスタ 8 0 2 は下位側に相当する。データ 1 には、サイズ情報 S 1 を示すサイズビットレジスタ 8 0 3 がある。同様に、8 0 4 及び 8 0 5 はデータ 2 を上位側と下位側に 2 分する際の上位レジスタと下位レジスタであり、8 0 6 はデータ 2 のサイズビットレジスタである。

10

#### 【 0 0 7 1 】

上記レジスタは全てフリップフロップで構成され、クロックの 1 の立ち上がりで動作する。8 0 7 及び 8 0 8 はそれぞれデータの上位側、下位側の加算器である。8 0 9 は制御部であり、演算の減算を示す S U B と並列演算を示す S M D 信号を出力し、演算の制御を行う。また、図 8 では省略してあるが、制御部 8 0 9 は上記レジスタのイネーブル信号も命令単位で出力する。8 1 0 及び 8 1 1 は減算を行う際に使用するデータ 2 と S U B 信号を入力とする E O R ゲートであり、それぞれデータ 2 の上位側、下位側に対応する。8 1 2 はデータ 2 のサイズビット S 2 に従って S U B 信号をマスクするための A N D ゲートである。

#### 【 0 0 7 2 】

また、8 1 3 はキャリー制御部であり、データ 1 とデータ 2 のサイズビットレジスタ 8 0 3 , 8 0 6 のサイズ情報 S 1 , S 2 と、S U B 信号、S M D 信号を入力とし、下位側の加算器 8 0 8 から上位側の加算器 8 0 7 への桁上げ信号の制御を行う。キャリー制御用のため、キャリー保持レジスタ 8 1 4 により以前のキャリーを保持する。サイズ情報が異なるデータの演算を行うときは、サイズを合わせるためにビット長の小さいデータを符号拡張して加算する必要がある。

20

#### 【 0 0 7 3 】

本実施の形態では、高速化を図るため、インクリメンタ、デクリメンタを用いて、上位側を予め計算しておき、下位側から上位側加算器へのキャリーイン C I U により上位側の加算結果を決定するインクリメンタ/デクリメンタ制御部 8 1 5 を使用する。上位結果セレクタ 8 1 6 において、データ 1 とデータ 2 のビット幅が異なる場合はインクリメンタ/デクリメンタ制御部 8 1 5 の出力を選択し、データ 1 とデータ 2 のビット幅が同じときは上位側加算器 8 0 7 の結果データ D 3 H をライン 8 1 7 に出力する。ライン 8 1 8 にデータ下位側加算器 8 0 8 の演算結果データ D 3 L を出力する。演算結果に付随する情報として、ライン 8 1 9 に結果サイズ S 3 を出力する。

30

#### 【 0 0 7 4 】

前述した第 1 の実施の形態との相違点は、第 1 の実施の形態が符号拡張制御により符号拡張データを上位側加算器に入力していたのに対し、本実施の形態では、インクリメンタ、デクリメンタを用いてインクリメンタ/デクリメンタ制御部 8 1 5 にて、上位側加算器 8 0 7 を用いずに高速に計算している点である。インクリメンタ/デクリメンタ制御部 8 1 5 内の “ + 1 ” はインクリメンタであり、入力を D とすると “ D + 1 ” を出力し、“ - 1 ” はデクリメンタであり、入力を D とすると “ D - 1 ” を出力する。インクリメンタ/デクリメンタ制御部 8 1 5 の動作を説明するために、表 2 を用いて説明する。尚、インクリメンタ/デクリメンタ制御部 8 1 5 以外の動作は図 1 と同じであるため、説明を省略する。

40

#### 【 0 0 7 5 】

#### 【表 2】

表 2

| 下位MSB | キャリ-CIU | 上位桁演算結果 |
|-------|---------|---------|
| 0     | 0       | D+0     |
| 0     | 1       | D+1     |
| 1     | 0       | D-1     |
| 1     | 1       | D+0     |

10

インクリメンタ/デクリメンタ制御部 815 は、入力データ 1 とデータ 2 のビット幅が異なるときに使用するので、表 2 の第 1 項の下位 MSB は、ビット幅が “ 0 ” のデータの下位 MSB である。

【 0076 】

第 2 項のキャリ-CIU は下位加算器 808 から上位加算器 807 へのキャリ-であり、第 3 項はビット幅が “ 1 ” の上位側データを D としたときの上位側演算結果である。例としてデータ 1 のサイズを “ 1 ”、データ 2 のサイズを “ 0 ” とすると、下位 MSB はデータ 2 の下位桁の MSB を指し、データ 1 の上位側データが D となる。

20

【 0077 】

下位 MSB が “ 0 ” で、キャリ-CIU が “ 0 ” のときは、上位側の演算結果が D であることからデータ 1 の上位側データ D をスルー出力するように、インクリメンタ/デクリメンタ制御部 815 の出力で選択される。

【 0078 】

下位 MSB が “ 0 ” で、キャリ-CIU が “ 1 ” のときは、上位側の演算結果が D + 1 であるので、データ 1 の上位側データ D のインクリメンタ結果を出力するように、インクリメンタ/デクリメンタ制御部 815 で選択される。

【 0079 】

下位 MSB が “ 1 ” で、キャリ-CIU が 0 のときは、上位側の演算結果が D - 1 であるので、データ 1 の上位側データ D のデクリメンタ結果を出力するように、インクリメンタ/デクリメンタ制御部 815 で選択される。

30

【 0080 】

下位 MSB が “ 1 ” で、キャリ-CIU が “ 1 ” のときは、上位側の演算結果が D であることからデータ 1 の上位側データ D をスルー出力するように、インクリメンタ/デクリメンタ制御部 815 で選択される。

以上のように、インクリメンタ/デクリメンタ制御部 815 を用いて、上位側加算を行う例を示した。

【 0081 】

図 1 の上位側加算器を用いる場合に対し、下位桁からのキャリ-イン CIU から上位側加算結果に至る論理段数が小さいため、高速化が可能である。図 8 では、両データが 16 ビットの演算が連続するとき、インクリメンタ/デクリメンタ制御部 815 の論理が動作してしまうため、図 1 に対し、電力を消費するが、両データのサイズビットが “ 0 ” のときに、セレクタの固定、データの固定を行うことにより、図 1 と同等の消費電力化も可能である。

40

【 0082 】

< 実施の形態 3 >

図 9 は本発明の第 3 の実施の形態を示す固定小数点の加減算回路のブロック図である。これまでの第 1 及び第 2 の実施の形態は、データの形式が整数に限って説明してきた。DS

50

Pなどのデジタル信号処理では、データの形式として固定小数点を使用する。本実施の形態では、データの形式が固定小数点の演算においても回路に変更を加えることで対応できることを示す。

#### 【0083】

加減算回路の2つの入力データをデータ1とデータ2とする。図9において、参照符号901及び902はデータ1を上位側と下位側に2分する際の上位レジスタと下位レジスタである。例としてデータ1のビット幅を32ビットとすると、レジスタ901は16ビットの上位側、レジスタ902は下位側に相当する。データ1には、サイズ情報を示すサイズビットレジスタ903がある。同様に、904及び905は、データ2を上位側と下位側に2分する際の上位レジスタと下位レジスタであり、906はデータ2のサイズビットレジスタである。上記レジスタは全てフリップフロップで構成され、クロック1の立ち上がりで動作する。907及び908はそれぞれデータの上位側、下位側の加算器である。909は制御部であり、演算の減算を示すSUB信号と並列演算を示すSMD信号を出力し、演算の制御を行う。また、図9では省略してあるが、制御部909は上記レジスタのイネーブル信号も命令単位で出力する。

10

#### 【0084】

910及び911は、減算を行う際に使用するデータ2とSUB信号を入力とするEORゲートであり、それぞれデータ2の上位側、下位側に対応する。912は、データ2のサイズビットに従ってSUB信号をマスクするためのANDゲートである。

#### 【0085】

913は下位桁加算器のキャリーイン制御部であり、データ1とデータ2のサイズビットレジスタ903、906を入力とし、下位側の加算器908へのキャリーイン制御を行う。キャリーイン制御部913用のためにキャリー保持レジスタ914により以前のキャリーを保持する。915は上位桁加算器907へのキャリー制御部であり、データ1とデータ2のサイズビットレジスタ903、906と、SUB信号及びSMD信号を入力とし、下位側の加算器908から上位側の加算器907への桁上げ信号の制御を行う。サイズ情報が異なるデータの演算を行うときに、下位桁をスルー出力する制御をスルー制御部916にて行う。出力ライン917及び918に、それぞれデータ上位側及び下位側の演算結果データD3H、D3Lを出力する。演算結果に付随する情報として、ライン919に結果サイズS3を出力する。

20

30

#### 【0086】

図9の加減算回路の動作を説明する前に、図10の固定小数点における演算について説明する。図10は表1の命令のうち4通りの加減算の例を示している。なお、図10において、は小数点位置を表す。

#### 【0087】

(1)データ1、データ2共に32ビットの加算(ADD32)の場合：

データ1とデータ2共に32ビットなので、どちらもサイズビットは“1”である。32ビット加算なので、下位加算器908のキャリーは“0”、上位加算器907のキャリーインCIUは下位加算器からの桁上げ(キャリーアウト)COである。結果は32ビットであり、サイズは“1”となる。

40

#### 【0088】

(2)データ1が32ビット、データ2が16ビットの加算(ADD32)の場合：

データ1は32ビットなので、サイズビットが“1”、データ2は16ビットなので、サイズビットは“0”である。サイズビットが“0”のとき、小数点位置に合わせてデータを配置すると、下位桁のデータはゼロとなる。下位桁に注目すると、データ1の下位桁D1Lと、ゼロとの加算なので、下位桁の結果はデータ1の下位桁D1Lをスルー出力すればよい。そこで、下位桁加算器908の入力として、D1側にD1Lを、D2側に前サイクルデータを、キャリーインCILに前サイクルキャリーCをそれぞれ与える。そしてD1Lのスルー出力は、加算器908とは別の下位結果スルー制御部916を使用する。上位側加算のキャリーインCIUは強制的に“0”を与え、上位側加算器917にて加算す

50



る。結果は32ビットであり、サイズは“1”となる。

【0089】

(3)データ1、データ2が16ビットの加算(ADD16)の場合：

データ1とデータ2は、16ビットなのでサイズビットは“0”である。下位側は演算に無関係なため、ドントケアである。そこで、下位側は前サイクルと同じ結果となるようにする。すなわち、入力データを前サイクルと同じデータにして、下位側加算のキャリーインCILを前サイクルと同じキャリーCとすればよい。上位側加算のキャリーインCIUを“0”とすれば、結果の上位16ビットは正しい演算結果が得られる。結果のサイズは“0”となる。

【0090】

(4)データ1、データ2が16ビットの並列減算(SUBS16)の場合：

データ1とデータ2は、32ビットなのでサイズビットは“1”である。上位側と下位側を並列に演算させるため、データ2の入力を反転し、上位側加算のキャリーインCIU、下位側加算のキャリーインCILをどちらも“1”とする。結果は32ビットであり、サイズは“1”となる。

【0091】

次に、図9の加減算回路の動作、特にキャリー上位制御部915、キャリー下位制御913に関して説明する。

【0092】

(1)データ1、データ2が32ビットのADD32命令、またはSUB32命令の場合：上位桁加算器907と下位桁加算器908を用いる。入力として、データ1のサイズビットレジスタ903に“1”、データ1の上位レジスタ901と下位レジスタ902にそれぞれD1H、D1Lが格納される。データ2も同様にサイズビットレジスタ906に“1”、データ2の上位レジスタ904と下位レジスタ905にそれぞれD2H、D2Lが格納される。

【0093】

制御部909からSUB信号が供給され、SUB32命令のとき“1”、ADD32命令のとき“0”となる。キャリー下位制御部913では、SUB信号が選択されるため、下位加算器908へのキャリーインCILはSUB32命令のとき“1”、ADD32命令のとき“0”となる。上位加算器907へのキャリーインCIUは、下位加算器908からの桁上げCOである。上位加算器907、下位908へのデータ2の入力は、SUB信号との排他的論理和EORであるため、ADD32命令のときはそれぞれD2H、D2Lであり、SUB32命令のときはそれぞれD2H、D2Lである。下位桁、上位桁ともに加算結果を結果とする。

【0094】

(2)データ1、データ2の一方が32ビットで、もう一方が16ビットのADD32命令、またはSUB32命令の場合：上位桁加算器907と下位桁スルー制御部916を用いる。例としてデータ2側を16ビットとすると、入力として、データ1のサイズビットレジスタ903に“1”、データ1の上位レジスタ901と下位レジスタ902にそれぞれD1H、D1Lが格納される。データ2はサイズビットレジスタ906に“0”、データ2の上位レジスタ904にD2Hが格納される。データ2の下位レジスタ905は変化しない。

【0095】

制御部909からSUB信号が供給され、SUB32命令のとき“1”、ADD32命令のとき“0”となる。キャリー下位制御部913では、キャリー保持信号が選択されるため、下位加算器908へのキャリーインCILはCとなる。上位加算器へのキャリーインCIUは、SUB信号が選択されるため、上位加算器907へのキャリーインCIUはSUB32命令のとき“1”、ADD32命令のとき“0”となる。下位桁ではADD32命令、SUB32命令のどちらの場合も、データ1の下位桁スルー出力D1Lが正しい結果となるため、下位結果スルー制御部916にて、データ1の下位桁D1Lを出力するよ

10

20

30

40

50

う制御される。

#### 【0096】

データ2入力は、SUB信号とのEORであるため、ADD32命令のときはD2HとD2Lであり、SUB32命令のときはD2HとD2Lである。データ2側が16ビットのときは、SUB信号による反転がマスクされる。上位桁は加算結果を結果とする。

#### 【0097】

(3)データ1、データ2のどちらも16ビットのADD16命令、またはSUB16命令の場合：

上位桁加算器907と下位桁スルー制御916を用いる。入力として、データ1のサイズビットレジスタ903に“0”、データ1の上位レジスタ901にD1Hが格納される。データ1の下位レジスタ902は変化しない。同様に、データ2はサイズビットレジスタ906に“0”、データ2の上位レジスタ904にD2Hが格納され、データ2の下位レジスタ905は変化しない。

10

#### 【0098】

制御部909からSUB信号が供給され、SUB16命令のときは“1”となり、ADD16命令のときは“0”となる。キャリー下位制御部913では、キャリー保持信号が選択されるため、下位加算器908へのキャリーインCILはCとなる。上位加算器907へのキャリーインCIUは、SUB信号が選択されるため、上位加算器へのキャリーインCIUはSUB16命令のとき“1”となり、ADD16命令のとき“0”となる。下位桁の加算器908は、データ1、データ2の入力、キャリーイン信号CILはいずれも前サイクルの値と同じであるため、16ビット命令が連続するときは、下位桁加算器908の電力を消費しない。

20

#### 【0099】

データ2入力は、SUB信号との排他的論理和EORであるため、ADD16命令のときはD2H、D2Lであり、SUB32命令のときはD2H、D2Lである。データ2側は16ビットなので、SUB信号による反転がマスクされる。下位桁、上位桁は加算結果を結果とする。

#### 【0100】

(4)データ1、データ2のどちらも32ビットの並行命令ADD S16、命令またはSUB S16命令の場合：

30

上位桁加算器907と下位桁加算器908を用いる。入力として、データ1のサイズビットレジスタ903に“1”、データ1の上位レジスタ901と下位レジスタ902にそれぞれD1H、D1Lが格納される。データ2も同様にサイズビットレジスタ906に“1”、データ2の上位レジスタ904と下位レジスタ905にそれぞれD2H、D2Lが格納される。制御部909からSUB信号が供給され、SUB S16命令のときは“1”、ADD S16命令のときは“0”となる。また、SMD信号が供給され“1”である。キャリー下位制御部913では、SUB信号が選択されるため、SUB S16命令のときは“1”となり、ADD S16命令のときは“0”となる。上位加算器907へのキャリーインCIUも同様に、SUB信号が選択されるため、上位加算器へのキャリーインCIUはSUB S16命令のときは“1”となり、ADD S16命令のときは“0”となる。データ2入力は、SUB信号との排他的論理和EORであるため、ADD16命令のときはD2H、D2Lであり、SUB32命令のときはD2H、D2Lである。下位桁、上位桁は加算結果を結果とする。

40

#### 【0101】

以上のように、図9に示す加減算器において、固定小数点データにおける表2の命令の動作を示した。16ビット命令では、下位側の加算器908の2つの入力データD1、D2とキャリーイン信号CILが固定されるため、16ビット命令が連続するときは、下位側加算器908の電力が消費されないという利点を持つ。

#### 【0102】

本実施の形態では、図3のCPU305をDSPに置きかえた場合に、データを固定小数

50

点の形式としたA L U 3 1 1の構成例を図9に示した。固定小数点データ形式のD S Pとするためには、レジスタファイル3 1 2やロードストアユニット3 1 0も変更する必要がある。

#### 【0103】

本実施の形態におけるレジスタファイルのデータチェック部4 0 3の構成例を図11に示す。図5ではデータの上位側が符号拡張であるかどうかを調べたが、図11では下位16ビットがゼロであるかどうかを調べ、ゼロであるときにサイズビットを“0”とする。入力、出力はどちらもサイズビットと、データ上位側と、データ下位側であり、ビット幅はそれぞれ1ビット、16ビット、16ビットである。

#### 【0104】

サイズビットは下位側のA 1 1 “0”検出1 1 0 1あるいはサイズビットの“0”検出を行い、いずれかの条件が成立するときにサイズビットは“0”となる。すなわち、入力が16ビットデータか、入力が32ビットデータで、かつ、下位側がゼロのときにサイズビットが“0”となる。A 1 1 “0”検出1 1 0 1は図5(c)に示したように、それぞれNORツリーにより構成される。

#### 【0105】

図12に、本実施の形態におけるレジスタファイルのデータ保持部4 0 4のうち、レジスタ[i] (0 i 7)の構成図を示す。図5ではデータのサイズが“0”のとき、上位側のラッチの更新を抑止したが、図12では、データのサイズが“0”のとき、下位側のラッチの更新を抑止する。サイズビット1 2 0 1、データ上位側1 2 0 2、データ下位側1 2 0 3は1 (1の反転)のクロックが供給されるラッチで構成される。ビット幅はそれぞれ1ビット、16ビット、16ビットである。サイズビット1 2 0 1、データ上位側のデータ1 2 0 2のラッチは、レジスタ[i]のライトイネーブル信号を表す制御部9 0 9からのライトイネーブル[i]と1のゲーテッドクロックである。データ下位側1 2 0 3はライトイネーブル[i]と1に加え、サイズビット1 2 0 1とのゲーテッドクロックである。

#### 【0106】

すなわち、データ下位側は、サイズビットが“1”で下位側がゼロでなく、イネーブルなデータのときのみライトされる。レジスタファイルはビット幅が大きいので、サイズビットが“0”のときのデータ下位側のライト抑止による電力低減の効果は大きい。データが全て16ビットのプログラムを実行した場合、サイズビットが全て“0”となることから、レジスタファイル部において約50%の電力削減が可能である。

#### 【0107】

図13に、本実施の形態におけるロードストア部の構成を示す。

ロードストア部1 3 0 1はロード部1 3 0 2とストア部1 3 0 6から成り、ロードデータとストアデータはレジスタファイルと接続される。ロードストアバス1 3 1 0はデータの上位ビットと下位ビットとの32ビットであり、CPU内の内部データ信号はサイズビットと、データ上位ビットと、下位ビットとの33ビットのビット幅で接続される。ロードデータ出力とストア入力は、どちらも33ビットである。

#### 【0108】

ロード部1 3 0 2では、ロードサイズの情報制御部9 0 9において命令をデコードすることにより得られ、16ビットロード命令の実行時はサイズビットが“0”となり、32ビットロード命令の実行時はサイズビットが“1”となる。そして、32ビットロード時(ワードロード=“0”)は、バス1 3 1 0上のデータがそのままロードデータとなるが、16ビットロード時(ワードロード=“1”)には、バス上の下位データがロードデータの上位側のデータとなる。

#### 【0109】

そこで、制御部9 0 9によりロードの適切なタイミングに応じて、32ビットロード命令では上位側ロードイネーブル信号1 3 0 3と下位側ロードイネーブル信号1 3 0 4が“1”、かつ、ワードロード信号1 3 0 5が“0”となり、バス上の32ビットデータがその

10

20

30

40

50

ままロードデータとなる。１６ビットロード命令では、下位側ロードイネーブル信号１３０４とワードロード信号１３０５が“１”なので、バス上の下位データが、上位側のロードデータとなり、正しくロードされる。

#### 【０１１０】

ストア部１３０６では、ストア命令の種類とサイズ情報から適正なデータを生成しストアする。３２ビットストア命令でデータのサイズ情報が“１”のときは、上位側のストアデータをバス上の上位側に、下位側のストアデータをバス上の下位側にストアする。１６ビットストア命令は、上位側のストアデータをバス上の下位側にストアし、３２ビットストア命令でデータのサイズ情報が“０”のときは、下位データをゼロにマスクした３２ビットデータをストアする必要がある。そこで、ストア用の下位データは、３２ビットストア時（ワードストア＝“０”）にサイズビットが“０”のときは下位データをゼロにマスクし、１６ビットストア時（ワードストア＝“１”）に上位側データとなるようにワードストア信号１３０７により動作するセレクタを用意する。そして、制御部９０９によりストアの適切なタイミングに応じて、３２ビットストア命令では上位側ストアイネーブル信号１３０８と下位側ストアイネーブル信号１３０９が“１”、１６ビットストア命令では下位側ストアイネーブル信号１３０９のみが“１”となり、ストア用に生成されたデータが正しくストアされる。

10

#### 【０１１１】

図１３のロード部１３０２においては、図７に対しバス上の下位側のデータを上位側のロードデータとするセレクタがある点、ストア部１３０６においては、図７の構成では符号拡張用のセレクタがあるのに対し、図１３の構成では、上位側のストアデータをバス上の下位側のデータとするセレクタがあるという相違点がある。

20

#### 【０１１２】

以上に述べた図１０のＡＬＵと、図４、図１１、図１２のレジスタファイルと、図１３のロードストアユニットを、図３の構成で使用するにより、固定小数点データによるＤＳＰが構成できる。

#### 【０１１３】

以上、本発明の好適な実施の形態についてを述べたが、本発明は前記実施の形態に限定するわけではなく、本発明の精神を逸脱しない範囲において種々の設計変更をなし得ることは勿論である。例えば、実施の形態ではビット幅に関して、全長３２ビットに対し１６ビットずつに２分しているが、全長や分割単位の変更の拡張も容易である。全長１２８ビット、８ビットずつ１６分割してもよい。

30

また、図６と図９では、１クロックのラッチを用いて構成したが、１クロックのフリップフロップを用いたレジスタファイルへの拡張も容易である。

#### 【０１１４】

##### 【発明の効果】

前述した実施の形態より明らかなように、本発明によれば、演算データにサイズ情報を設け、データを上位と下位に分割し、データ単位にサイズビットを設け、サイズビットにより有効／無効を判定して、演算を最小の有効データ幅で動作させる。すなわち、サイズ情報に基づいて常に最小の有効データ幅での演算が可能となるため、常に最大データ幅を使用する従来の演算装置に比べて、電力削減ができる。

40

##### 【図面の簡単な説明】

【図１】本発明の第１の実施の形態を示す加減算回路の構成例を示すブロック図。

【図２】表１に示した命令のうちの４通りの加減算の例を示す説明図。

【図３】第１～３の実施の形態の加減算回路を適用するマイクロプロセッサのブロック図。

【図４】第１～３の実施の形態のレジスタファイルの構成例を示すブロック図。

【図５】第１及び第２の実施の形態のレジスタファイルのデータ保持部の構成例を示すブロック図。

【図６】第１及び第２の実施の形態のレジスタファイルのデータチェック部の構成例を示

50

すブロック図。

【図 7】第 1 及び第 2 の実施の形態のロードストア部の構成例を示すブロック図。

【図 8】第 2 の実施の形態の加減算回路の構成例を示すブロック図。

【図 9】第 3 の実施の形態の加減算回路の構成例を示すブロック図。

【図 10】表 1 に示した命令のうちの 4 通りの固定小数点における演算例を示す説明図。

【図 11】第 3 の実施の形態のレジスタファイルのデータチェック部の構成例を示すブロック図。

【図 12】第 3 の実施の形態のレジスタファイルのデータ保持部の構成例を示すブロック図。

【図 13】第 3 の実施の形態のロードストア部の構成例を示すブロック図。

10

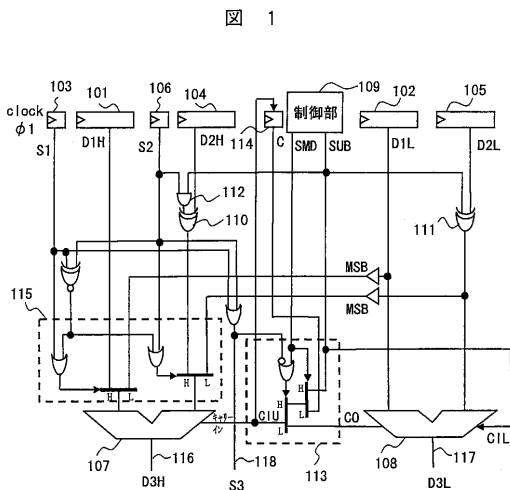
【符号の説明】

1 0 1 , 8 0 1 , 9 0 1 ... データ 1 の上位桁レジスタ、1 0 2 , 8 0 2 , 9 0 2 ... データ 1 の下位桁レジスタ、1 0 3 , 8 0 3 , 9 0 3 ... データ 1 のサイズビットレジスタ、1 0 4 , 8 0 4 , 9 0 4 ... データ 2 の上位桁レジスタ、1 0 5 , 8 0 5 , 9 0 5 ... データ 2 の下位桁レジスタ、1 0 6 , 8 0 6 , 9 0 6 ... データ 2 のサイズビットレジスタ、1 0 7 , 8 0 7 , 9 0 7 ... 上位桁加算器、1 0 8 , 8 0 8 , 9 0 8 ... 下位桁加算器、1 0 9 , 8 0 9 , 9 0 9 ... 制御部、1 1 0 , 1 1 1 , 8 1 0 , 8 1 1 , 9 1 0 , 9 1 1 ... E O R ゲート、1 1 2 , 8 1 2 , 9 1 2 ... A N D ゲート、1 1 3 , 8 1 3 ... キャリー制御部、1 1 4 , 8 1 4 , 9 1 4 ... キャリー保持レジスタ、1 1 5 ... 符号拡張制御部、1 1 6 , 8 1 7 , 9 1 7 ... 結果上位桁の出力ライン、1 1 7 , 8 1 8 , 9 1 8 ... 結果下位桁の出力ライン、1 1 8 , 8 1 9 , 9 1 9 ... 結果サイズ出力ライン、3 0 1 ... 外部メモリ、3 0 2 ... 外部データバス、3 0 3 ... 外部アドレスバス、3 0 4 ... マイクロプロセッサ ( M P U )、3 0 5 ... C P U、3 0 6 ... バスインターフェースユニット、3 0 7 ... ロードストアバス、3 0 8 ... 制御部、3 0 9 ... データバス部、3 1 0 , 7 0 1 , 1 3 0 1 ... ロードストア部、3 1 1 ... A L U、3 1 2 ... レジスタファイル、4 0 2 ... 入力セクタ、4 0 3 ... データチェック部、4 0 4 ... データ保持部、4 0 5 , 6 0 1 , 1 2 0 1 ... サイズレジスタ、4 0 6 , 6 0 2 , 1 2 0 2 ... 上位桁レジスタ、4 0 7 , 6 0 3 , 1 2 0 3 ... 下位桁レジスタ、4 0 8 ... 出力セクタ、5 0 1 ... A 1 1 “ 1 ” 検出、5 0 2 , 1 1 0 1 ... A 1 1 “ 0 ” 検出、7 0 2 , 1 3 0 2 ... ロード部、7 0 3 , 1 3 0 3 ... ロードイネーブル上位、7 0 4 , 1 3 0 4 ... ロードイネーブル下位、7 0 5 , 1 3 0 6 ... ストア部、7 0 6 ... ストア上位桁セクタ、7 0 7 , 1 3 0 8 ... ストアイネーブル上位、7 0 8 , 1 3 0 9 ... ストアイネーブル下位、8 1 5 ... インクリメンタ / デクリメンタ制御部、8 1 6 ... 上位結果セクタ、9 1 3 ... キャリー下位制御部、9 1 5 ... キャリー上位制御部、9 1 6 ... 下位結果スルー制御部、1 3 0 5 ... ワードロード信号、1 3 0 7 ... ワードストア信号、1 3 1 0 ... ロードストアバス、D 1 H ... データ 1 の上位ビット、D 2 H ... データ 2 の上位ビット、D 3 H ... データ上位側の演算結果、D 1 L ... データ 1 の下位ビット、D 2 L ... データ 2 の下位ビット、D 3 L ... データ下位側の演算結果、S M D 信号 ... 並列演算を示す信号、S U B 信号 ... 減算を示す信号、C I U ... 上位側加算器へのキャリーイン、C I L ... 下位側加算器へのキャリーイン、S , S 1 , S 2 , S 3 ... サイズビット。

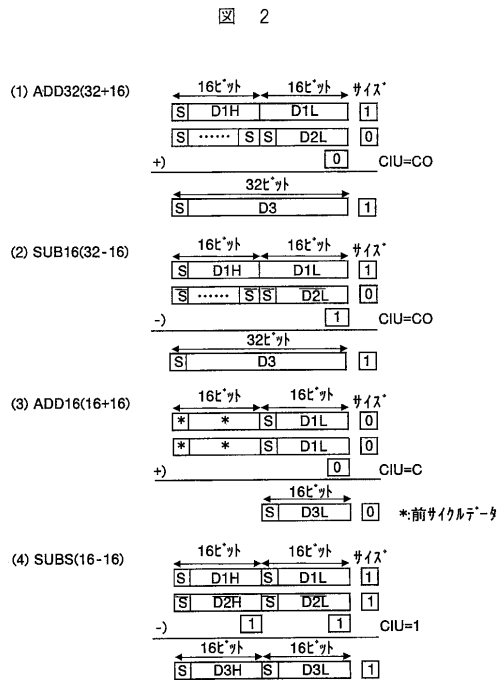
20

30

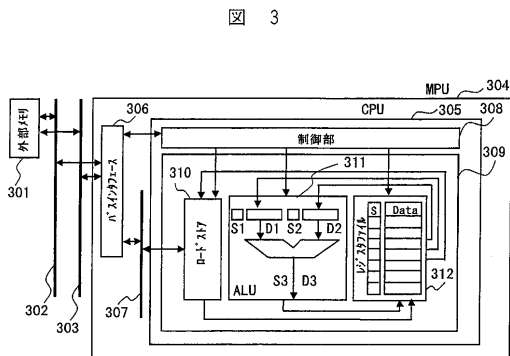
【 図 1 】



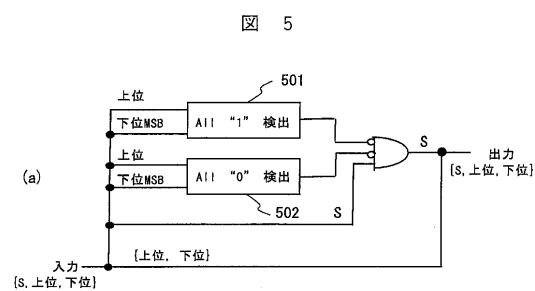
【 図 2 】



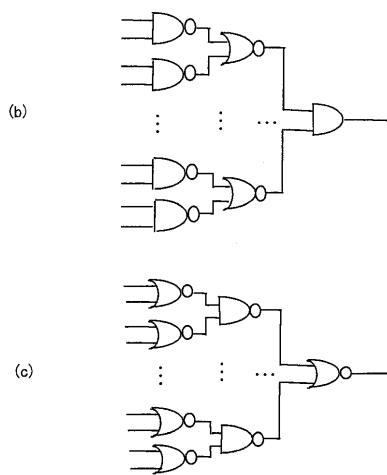
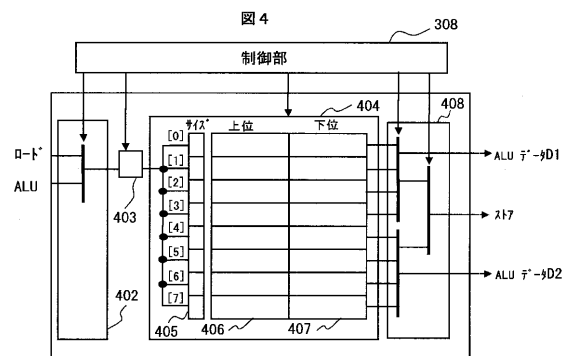
【 図 3 】



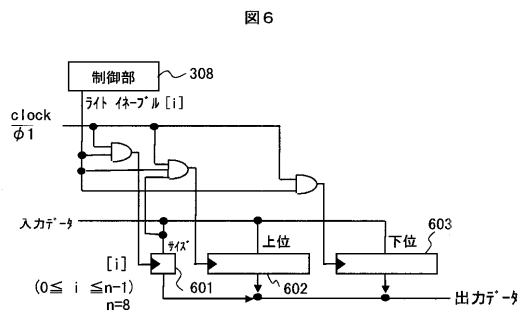
【 図 5 】



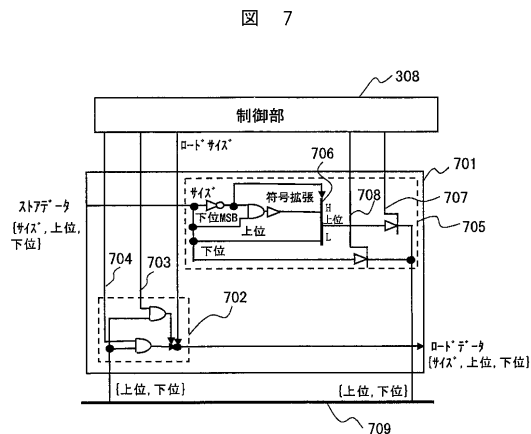
【 図 4 】



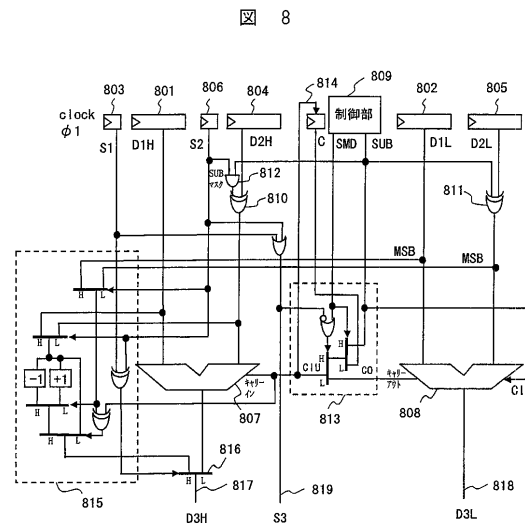
【 図 6 】



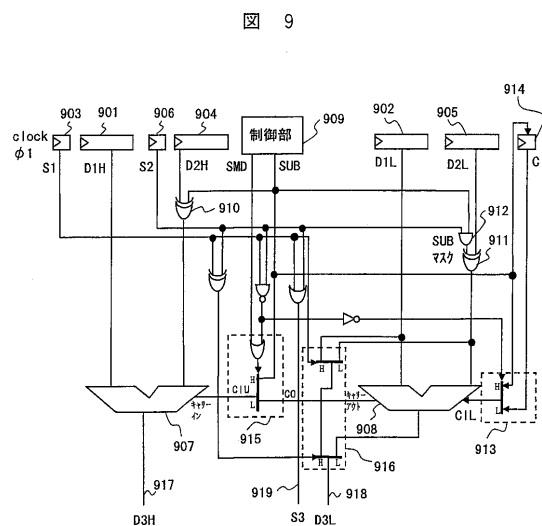
【圖 7】



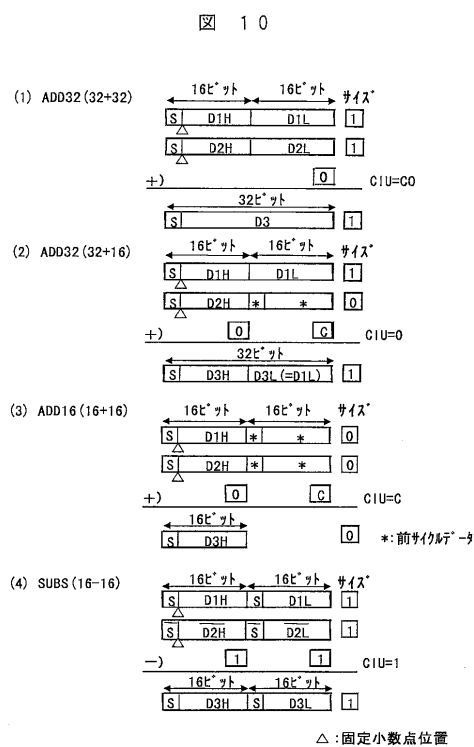
【图 8】



【 図 9 】

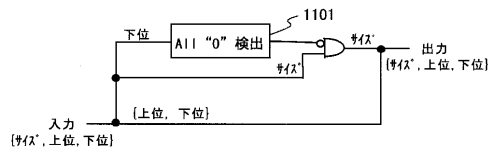


【 図 1 0 】



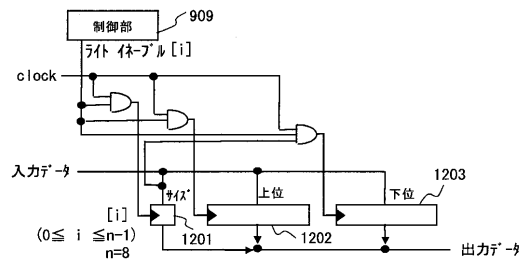
【図 1 1】

図 1 1



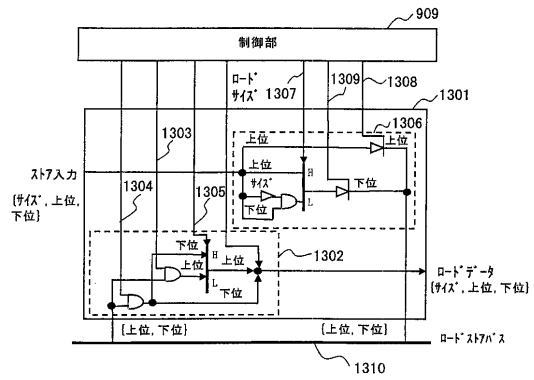
【図 1 2】

図 1 2



【図 1 3】

図 1 3





---

フロントページの続き

- (56)参考文献 特開平07-248895(JP,A)  
特開2001-216136(JP,A)  
特開平07-271554(JP,A)

(58)調査した分野(Int.Cl., DB名)

G06F 7/00  
G06F 9/30  
WPI(DIALOG)