

(19) 日本国特許庁(JP)

(12) 特 許 公 報(B2)

(11) 特許番号
特許第4201980号
(P4201980)

(45) 発行日 平成20年12月24日(2008.12.24)

(24) 登録日 平成20年10月17日(2008.10.17)

(51) Int.Cl.

G09C 1/00 (2006.01)

F I

G09C 1/00 650A

請求項の数 21 (全 37 頁)

(21) 出願番号	特願2000-522511 (P2000-522511)	(73) 特許権者	591225523
(86) (22) 出願日	平成10年11月4日 (1998.11.4)		アトメル・コーポレーション
(65) 公表番号	特表2001-524698 (P2001-524698A)		ATMEL CORPORATION
(43) 公表日	平成13年12月4日 (2001.12.4)		アメリカ合衆国、95131 カリフォル
(86) 国際出願番号	PCT/US1998/023476		ニア州、サン・ノゼ、オーチャード・パー
(87) 国際公開番号	W01999/027437		クウェイ、2325
(87) 国際公開日	平成11年6月3日 (1999.6.3)	(74) 代理人	100077481
審査請求日	平成17年6月17日 (2005.6.17)		弁理士 谷 義一
(31) 優先権主張番号	08/979,803	(74) 代理人	100088915
(32) 優先日	平成9年11月26日 (1997.11.26)		弁理士 阿部 和夫
(33) 優先権主張国	米国 (US)	(72) 発明者	サビン マイケル ジェイ.
			アメリカ合衆国 94087 カリフォル
			ニア州 サニーベイル マンゴ アベニュー
			883

最終頁に続く

(54) 【発明の名称】 マルチプレシジョン整数演算用の装置

(57) 【特許請求の範囲】

【請求項1】

第1のデータ信号yを第2のデータ信号xとともに処理するマルチプレシジョン整数演算用の装置であって、

「n」を複数の大整数ユニット内の大整数ユニットの総数とする複数の大整数ユニット（LIU）、LIU{0}、LIU{1}、LIU{2}、...、LIU{n-1}であって、各大整数ユニットが、

第1の乗算器入力、第2の乗算器入力、上位部分積出力、および下位部分積出力を有する乗算器と、

第1の加算器入力、第2の加算器入力、第3の加算器入力、第4の加算器入力、および加算器出力を有し、前記第3の加算器入力の下位部分積出力を受け取るように乗算器に結合された加算器と、

LIUレジスタ入力およびLIUレジスタ出力を有するLIUレジスタであって、前記LIUレジスタ入力は加算器出力を受け取るように加算器に結合され、前記LIUレジスタはクロックパルスの前に加算器出力の値をラッチしてクロックパルスの後にその値を前記LIUレジスタ出力に転送し、前記LIUレジスタ出力はLIUレジスタ上位出力とLIUレジスタ下位出力に分割され、前記LIUレジスタ上位出力は前記LIUレジスタ出力の上位2ビットを含み、前記LIUレジスタ下位出力は前記LIUレジスタ上位出力のビット以外の前記LIUレジスタ出力の全ビットを含

10

20

み、前記 L I U レジスタ上位出力は第 4 の加算器入力に結合され、前記 L I U レジスタはさらに L I U レジスタクリア入力がアクティブとされたときに前記 L I U レジスタ出力を 0 にクリアするための前記 L I U レジスタクリア入力を有する L I U レジスタと、

L I U レジスタ出力とキャリーイン信号 c_i を受け取るため L I U レジスタに結合された第 1 の論理 OR ゲートであって、前記第 1 の論理 OR ゲートはキャリーアウト信号 c_o を発生し、前記キャリーアウト信号 c_o はアクティブとされている L I U レジスタ出力の任意の 1 ビットにตอบสนองしてアクティブとされ、前記キャリーイン信号 c_i はアクティブとされる第 1 の論理 OR ゲートと、

10

L I U レジスタ上位出力を受け取り、補助キャリーアウト信号 c_a を発生するため L I U レジスタに結合された第 2 の論理 OR ゲートであって、前記補助キャリーアウト信号 c_a は L I U レジスタ上位出力のいずれかのビットがアクティブのときにアクティブとされる第 2 の論理 OR ゲートとを有する複数の大整数ユニットを備え、

前記複数の大整数ユニットのそれぞれの大整数ユニットの第 1 の乗算器入力に第 1 のデータ信号 y が印加されており、

複数の要素 $x_{\{0\}}$ 、 $x_{\{1\}}$ 、 $x_{\{2\}}$ 、 $\dots$ 、 $x_{\{n-1\}}$ を持つ第 2 のデータ信号 x が、各要素を前記複数の大整数ユニットの各大整数ユニットの第 2 の乗算器入力に結合されており、

20

0 よりも大きい i に対し、前記複数の大整数ユニットの各 i 番目の大整数ユニット L I U $\{i\}$ は、 $\{i-1\}$ 番目の大整数ユニット L I U $\{i-1\}$ の上位部分積出力に結合された第 1 の加算器入力を持ち、

0 よりも大きい i に対し、各 i 番目の大整数ユニット L I U $\{i\}$ は、 $\{i-1\}$ 番目の大整数ユニット L I U $\{i-1\}$ の第 2 の加算器入力に結合された L I U レジスタ下位出力を持ち、

減算信号がアクティブのときに第 3 のデータ信号 z の補数をとるための、第 1 の大整数ユニット L I U $\{0\}$ の第 1 の加算器入力に結合された第 1 の相補ゲートと、

第 1 の大整数ユニット L I U $\{0\}$ の L I U レジスタ下位出力に結合され、前記減算信号がアクティブのときに前記 L I U レジスタ下位出力の補数をとるための、第 4 のデータ信号 a を発生する第 2 の相補ゲートと、

30

ラッチレジスタ入力とラッチレジスタ出力を有するラッチレジスタであって、前記ラッチレジスタ入力は前記 n 番目の大整数ユニット L I U $\{n-1\}$ の上位部分積出力に結合され、前記ラッチレジスタ出力は、クロックパルスの前に n 番目の上位部分積の値をラッチし、クロックパルスの後に n 番目の上位部分積の値を前記 n 番目の大整数ユニットの第 2 の加算器入力に供給するための前記 n 番目の大整数ユニット L I U $\{n-1\}$ の第 2 の加算器入力に結合されたラッチレジスタとを備え、

前記第 1 の大整数ユニット L I U $\{0\}$ は論理 0 の値に結合された c_i 入力を持ち、

前記第 2 の大整数ユニット L I U $\{1\}$ は第 1 の大整数ユニット L I U $\{0\}$ の c_a 出力に結合された c_i 入力を持ち、

40

1 よりも大きい i に対し、前記複数の大整数ユニットのそれぞれの i 番目の大整数ユニット L I U $\{i\}$ は $\{i-1\}$ 番目の大整数ユニット L I U $\{i-1\}$ の c_o 出力に結合された c_i 入力を持ち、 n 番目の大整数ユニット L I U $\{n-1\}$ の c_o 出力は第 5 のデータ信号、キャリーフロー信号 c_y であり、

前記ラッチレジスタはラッチレジスタクリア信号を持ち、また各 L I U は L I U レジスタクリア入力を持ち、複数の L I U レジスタクリア入力と前記ラッチレジスタクリア信号と一緒に結合されてデータ信号 $c_l r$ を形成し、

前記ラッチレジスタはクロック入力を持ち、また各 L I U はクロック入力を持ち、複数のクロック入力と一緒に結合されてクロック信号 $c_l k$ を形成することを特徴とする装置。

【請求項 2】

50

第1のデータ信号 y を第2のデータ信号 x とともに処理するマルチプレシジョン整数演算用の装置であって、

「 n 」を複数の大整数ユニット内の大整数ユニットの総数とする複数の大整数ユニット ($L I U$)、 $L I U \{0\}$ 、 $L I U \{1\}$ 、 $L I U \{2\}$ 、 $\dots$ 、 $L I U \{n-1\}$ であって、各大整数ユニットが、

第1の乗算器入力、第2の乗算器入力、上位部分積出力、および下位部分積出力を有する乗算器と、

第1の加算器入力および第2の加算器入力を有し、下位部分積出力を受け取るため乗算器に結合された加算器と、

総和信号出力を持っており、前記加算器と結合された前記 $L I U$ レジスタと

10

を有する複数の大整数ユニットを備え、

前記複数の大整数ユニットのそれぞれの大整数ユニットの第1の乗算器入力に第1のデータ信号 y が印加されており、

複数の要素 $x \{0\}$ 、 $x \{1\}$ 、 $x \{2\}$ 、 $\dots$ 、 $x \{n-1\}$ を持つ第2のデータ信号 x が、各要素を前記複数の大整数ユニットの各大整数ユニットの第2の乗算器入力に結合されており、

0よりも大きい i に対し、前記複数の大整数ユニットの各 i 番目の大整数ユニット $L I U \{i\}$ は、 $\{i-1\}$ 番目の大整数ユニット $L I U \{i-1\}$ の上位部分積出力に結合された第1の加算器入力を持ち、

20

0よりも大きい i に対し、各 i 番目の大整数ユニット $L I U \{i\}$ は、 $\{i-1\}$ 番目の大整数ユニット $L I U \{i-1\}$ の第2の加算器入力に結合された総和信号出力を持ち、減算信号がアクティブのときに第3のデータ信号 z の補数をとるための、第1の大整数ユニット $L I U \{0\}$ の第1の加算器入力に結合された相補ゲートと、

前記 n 番目の大整数ユニット $L I U \{n-1\}$ の上位部分積出力と前記 n 番目の大整数ユニット $L I U \{n-1\}$ の第2の加算器入力とに結合され、クロックパルスの前に n 番目の上位部分積の値をラッチし、クロックパルスの後に n 番目の上位部分積の値を前記 n 番目の大整数ユニットの第2の加算器入力に供給するためのラッチレジスタと

を備えることを特徴とする装置。

【請求項3】

30

第1のデータ信号 y を第2のデータ信号 x とともに処理するマルチプレシジョン整数演算用の装置であって、

コンローラと、該コンローラに結合されて該コンローラから第1のデータ信号 y および第2のデータ信号 x を受け取るように構成された大整数ユニット ($L I U$) とを備えており、

該 $L I U$ が、

第1の乗算器入力、第2の乗算器入力、上位部分積出力、および下位部分積出力を有する乗算器と、

第1の加算器入力および第2の加算器入力を有し、前記下位部分積出力を受け取るため前記乗算器に結合され、第1の総和信号を出力する加算器と、

40

前記加算器に結合され、第1の総和信号を受け取り、第2の総和信号を出力する $L I U$ レジスタと、

第1の乗算器入力に印加される第1のデータ信号 y と、

第2の乗算器入力に印加される第2のデータ信号 x の複数の要素と、

第1の加算器入力に結合され、減算信号がアクティブのときに第3のデータ信号 z の補数をとる相補ゲートと、

前記上位部分積出力と第2の加算器入力とに結合され、クロックパルスの前に上位部分積の値をラッチし、該クロックパルスの後に該上位部分積の値を第2の加算器入力に供給するラッチレジスタと

を備えることを特徴とする マルチプレシジョン整数演算用の装置。

50

【請求項 4】

前記大整数ユニットがさらに、
前記 L I U レジスタの出力に結合されて前記 L I U レジスタからキャリー信号として第 2 の総和信号の上位ビットを受け取る O R ゲートであって、キャリー信号の論理 O R をキャリーフロー信号として出力する第 2 の O R ゲートを備えることを特徴とする請求項 3 に記載の装置。

【請求項 5】

さらに、
「 n 」を複数の大整数ユニット内の大整数ユニットの総数とする複数の大整数ユニット L I U $\{0\}$ 、L I U $\{1\}$ 、L I U $\{2\}$ 、 $\dots$ 、L I U $\{n-1\}$ を備え、
0 よりも大きい i に対し、前記複数の大整数ユニットの各 i 番目の大整数ユニット L I U $\{i\}$ は、 $\{i-1\}$ 番目の大整数ユニット L I U $\{i-1\}$ の上位部分積出力に結合された第 1 の加算器入力を持ち、
0 よりも大きい i に対し、各 i 番目の大整数ユニット L I U $\{i\}$ は、 $\{i-1\}$ 番目の大整数ユニット L I U $\{i-1\}$ の第 2 の加算器入力に結合された総和信号出力を持っていることを特徴とする請求項 3 または 4 に記載の装置。

【請求項 6】

第 1 のデータ信号 x を第 2 のデータ信号 y とともに処理するマルチプレシジョン整数演算用の装置であって、

コンローラと、該コンローラに結合されて該コンローラから第 1 のデータ信号 y および第 2 のデータ信号 x を受け取るように構成された複数の大整数ユニット (L I U) とを備えており、

前記複数の大整数ユニットのそれぞれの大整数ユニットが第 1 の乗算器入力、第 2 の乗算器入力、上位部分積出力、第 1 の加算器入力、第 2 の加算器入力、および総和信号出力を有しており、

前記複数の大整数ユニットのそれぞれの大整数ユニットの第 1 の乗算器入力に第 1 のデータ信号 y が印加され、

複数の要素を持つ第 2 のデータ信号 x の各要素が、前記複数の大整数ユニットのそれぞれの大整数ユニットの第 2 の乗算器入力に結合されており、

初期の大整数ユニット以外の i に対し、前記複数の大整数ユニットの各 i 番目の大整数ユニット L I U $\{i\}$ は、 $\{i-1\}$ 番目の大整数ユニット L I U $\{i-1\}$ の前記上位部分積出力に結合された第 1 の加算器入力を持ち、

前記初期の大整数ユニット以外の i に対し、前記複数の大整数ユニットの各 i 番目の大整数ユニット L I U $\{i\}$ は、 $\{i-1\}$ 番目の大整数ユニット L I U $\{i-1\}$ の第 2 の加算器入力に結合された前記総和信号出力を持つ
ことを特徴とする マルチプレシジョン整数演算用の装置。

【請求項 7】

さらに、
減算信号がアクティブのときに第 3 のデータ信号 z の補数をとるための、第 1 の大整数ユニット L I U $\{0\}$ の第 1 の加算器入力に結合された相補ゲートと、

「 n 」を大整数ユニットの総数としたとき、前記 n 番目の大整数ユニット L I U $\{n-1\}$ の上位部分積出力と前記 n 番目の大整数ユニット L I U $\{n-1\}$ の第 2 の加算器入力とに結合され、クロックパルスの前に n 番目の上位部分積出力の値をラッチし、クロックパルスの後に n 番目の上位部分積出力の値を前記 n 番目の大整数ユニットの第 2 の加算器入力に供給するためのラッチレジスタと
を備えることを特徴とする請求項 6 に記載のマルチプレシジョン整数演算用の装置。

【請求項 8】

さらに、減算信号がアクティブのときに第 3 のデータ信号 z の補数をとるための、第 1 の大整数ユニット L I U $\{0\}$ の第 1 の加算器入力に結合された相補ゲートを備えることを特徴とする請求項 6 に記載のマルチプレシジョン整数演算用の装置。

【請求項 9】

さらに、「 n 」を複数の大整数ユニット内の大整数ユニットの総数としたとき、前記 n 番目の大整数ユニット $L I U \{n-1\}$ の上位部分積出力と前記 n 番目の大整数ユニット $L I U \{n-1\}$ の第 2 の加算器入力とに結合され、クロックパルスの前に n 番目の上位部分積出力の値をラッチし、クロックパルスの後に n 番目の上位部分積出力の値を前記 n 番目の大整数ユニットの第 2 の加算器入力に供給するためのラッチレジスタを備えることを特徴とする請求項 6 または 8 に記載のマルチプレシジョン整数演算用の装置。

【請求項 10】

さらに、
減算信号がアクティブのときに第 3 のデータ信号 z の補数をとるための、第 1 の大整数ユニット $L I U \{0\}$ の第 1 の加算器入力に結合されたゲート手段と、 10

「 n 」を複数の大整数ユニット内の大整数ユニットの総数としたとき、 n 番目の大整数ユニット $L I U \{n-1\}$ の上位部分積出力と前記 n 番目の大整数ユニット $L I U \{n-1\}$ の第 2 の加算器入力とに結合され、クロックパルスの前に n 番目の上位部分積の値をラッチし、クロックパルスの後に n 番目の上位部分積の値を前記 n 番目の大整数ユニットの第 2 の加算器入力に供給するためのレジスタ手段と
を備えることを特徴とする請求項 6 に記載のマルチプレシジョン整数演算用の装置。

【請求項 11】

さらに、減算信号がアクティブのときに第 3 のデータ信号 z の補数をとるための、第 1 の大整数ユニット $L I U \{0\}$ の第 1 の加算器入力に結合されたゲート手段を備えることを 20
特徴とする請求項 6 に記載のマルチプレシジョン整数演算用の装置。

【請求項 12】

さらに、「 n 」を複数の大整数ユニット内の大整数ユニットの総数としたとき、 n 番目の大整数ユニット $L I U \{n-1\}$ の上位部分積出力と前記 n 番目の大整数ユニット $L I U \{n-1\}$ の第 2 の加算器入力とに結合され、クロックパルスの前に n 番目の上位部分積の値をラッチし、クロックパルスの後に n 番目の上位部分積の値を前記 n 番目の大整数ユニットの第 2 の加算器入力に供給するためのレジスタ手段を備えることを特徴とする請求項 6 または 11 に記載のマルチプレシジョン整数演算用の装置。

【請求項 13】

さらに、 30
第 1 の大整数ユニット $L I U \{0\}$ の総和信号出力に結合されて、出力信号を出力する出力ゲートを備えることを特徴とする請求項 2 に記載のマルチプレシジョン整数演算用の装置。

【請求項 14】

さらに、
前記大整数ユニットの総和信号出力に結合されて、出力信号を出力する出力ゲートを備えることを特徴とする請求項 3 に記載のマルチプレシジョン整数演算用の装置。

【請求項 15】

第 1 のデータ信号 x を第 2 のデータ信号 y とともに処理するマルチプレシジョン整数演算用の装置であって、 40

コンローラと、 n 個 ($n > 1$) の大整数ユニット ($L I U$) を持った大整数ユニット ($L I U$) アレイとを備えており、

該 $L I U$ のそれぞれが、該コンローラに結合されて該コンローラから第 1 のデータ信号 y および第 2 のデータ信号 x を受け取るように構成されており、

該 $L I U$ が、

第 1 の乗算器入力、第 2 の乗算器入力、上位部分積出力、および下位部分積出力を有する乗算器と、

第 1 の加算器入力および第 2 の加算器入力を有し、前記乗算器に結合されて前記下位部分積出力を受け取り、第 1 の総和信号を出力する加算器と、

前記加算器に結合され、第 1 の総和信号を受け取り、第 2 の総和信号を出力する $L I U$ 50

レジスタと、

前記 L I U レジスタの出力に結合されて前記 L I U レジスタからキャリー信号として第 2 の総和信号の上位ビットを受け取る O R ゲートであって、前記キャリー信号の論理 O R をキャリーフロー信号として出力する O R ゲートとを備えており、

第 1 の乗算器入力に第 1 のデータ信号 y が印加され、第 2 の乗算器入力に複数の要素を持った第 2 のデータ信号 x が印加される

ことを特徴とするマルチプレシジョン整数演算用の装置。

【請求項 16】

さらに、

前記大整数ユニットの第 1 の加算器入力に結合され、減算信号がアクティブのときに第 3 のデータ信号 z の補数をとる相補ゲートと、

前記大整数ユニットの上位部分積出力と前記大整数ユニットの第 2 の加算器入力とに結合され、クロックパルスの前に上位部分積出力の値をラッチし、クロックパルスの後に上位部分積出力の値を前記大整数ユニットの第 2 の加算器入力に供給するラッチレジスタとを備えることを特徴とする請求項 15 に記載のマルチプレシジョン整数演算用の装置。

【請求項 17】

さらに、前記大整数ユニットの第 1 の加算器入力に結合され、減算信号がアクティブのときに第 3 のデータ信号 z の補数をとる相補ゲートを備えることを特徴とする請求項 15 に記載のマルチプレシジョン整数演算用の装置。

【請求項 18】

さらに、前記大整数ユニットの上位部分積出力と前記大整数ユニットの第 2 の加算器入力とに結合され、クロックパルスの前に上位部分積出力の値をラッチし、クロックパルスの後に上位部分積出力の値を前記大整数ユニットの第 2 の加算器入力に供給するラッチレジスタを備えることを特徴とする請求項 15 または 17 に記載のマルチプレシジョン整数演算用の装置。

【請求項 19】

第 1 のデータ信号 y を第 2 のデータ信号 x とともに処理するマルチプレシジョン整数演算用の装置であって、

「n」を複数の大整数ユニット内の大整数ユニットの総数とする複数の大整数ユニット (L I U)、L I U { 0 }、L I U { 1 }、L I U { 2 }、...、L I U { n - 1 } であって、各大整数ユニットが、

第 1 の乗算器入力、第 2 の乗算器入力、上位部分積出力、および下位部分積出力を有する乗算器と、

第 1 の加算器入力、第 2 の加算器入力、第 3 の加算器入力、第 4 の加算器入力、および加算器出力を有し、前記第 3 の加算器入力の下位部分積出力を受け取るように乗算器に結合された加算器と、

L I U レジスタ入力および L I U レジスタ出力を有する L I U レジスタであって、前記 L I U レジスタ入力は加算器出力を受け取るように加算器に結合され、前記第 1 の L I U レジスタはクロックパルスの前に加算器出力の値をラッチしてクロックパルスの後にその値を前記 L I U レジスタ出力に転送し、前記 L I U レジスタ出力は L I U レジスタ上位出力と L I U レジスタ下位出力に分割され、前記 L I U レジスタ上位出力は前記 L I U レジスタ出力の上位 2 ビットを含み、前記 L I U レジスタ下位出力は前記 L I U レジスタ上位出力のビット以外の前記 L I U レジスタ出力の全ビットを含み、前記 L I U レジスタ上位出力は第 4 の加算器入力に結合され、前記第 1 の L I U レジスタはさらに L I U レジスタクリア入力がアクティブとされたときに前記 L I U レジスタ出力を 0 にクリアするための前記 L I U レジスタクリア入力を有する第 1 の L I U レジスタと、乗算器の上位部分積出力を受け取るための第 2 の L I U レジスタと、L I U レジスタ出力とキャリーイン信号 c i を受け取るため第 1 の L I

10

20

30

40

50

Uレジスタに結合された第1の論理ORゲートであって、前記第1の論理ORゲートはキャリーアウト信号 c_o を発生し、前記キャリーアウト信号 c_o はアクティブとされているL I Uレジスタ出力の任意の1ビットにตอบสนองしてアクティブとされ、前記キャリーイン信号 c_i はアクティブとされる第1の論理ORゲートと、

L I Uレジスタ上位出力を受け取り、補助キャリーアウト信号 c_a を発生するため第1のL I Uレジスタに結合された第2の論理ORゲートであって、前記補助キャリーアウト信号 c_a はL I Uレジスタ上位出力のいずれかのビットがアクティブのときにアクティブとされる第2の論理ORゲートと

10

を有する複数の大整数ユニットを備え、

前記複数の大整数ユニットのそれぞれの大整数ユニットの第1の乗算器入力に第1のデータ信号 y が印加されており、

複数の要素 $x_{\{0\}}$ 、 $x_{\{1\}}$ 、 $x_{\{2\}}$ 、 $\dots$ 、 $x_{\{n-1\}}$ を持つ第2のデータ信号 x が、各要素を前記複数の大整数ユニットの各大整数ユニットの第2の乗算器入力に結合されており、

前記複数の大整数ユニットの各 i 番目の大整数ユニットL I U $\{i\}$ は、各 i 番目の大整数ユニットL I U $\{i\}$ の上位部分積出力に結合された第1の加算器入力を持ち、

0よりも大きい i に対し、各 i 番目の大整数ユニットL I U $\{i\}$ は、 $\{i-1\}$ 番目の大整数ユニットL I U $\{i-1\}$ の第2の加算器入力に結合されたL I Uレジスタ下位出力を持ち、

20

第1の大整数ユニットL I U $\{0\}$ の第1の加算器入力に結合され、減算信号がアクティブのときに第3のデータ信号 z の補数をとるための第1の相補ゲートと、

第1の大整数ユニットL I U $\{0\}$ のL I Uレジスタ下位出力に結合され、前記減算信号がアクティブのときに前記L I Uレジスタ下位出力の補数をとるための、第4のデータ信号 a を発生する第2の相補ゲートとを備え、

前記第1の大整数ユニットL I U $\{0\}$ は論理0の値に結合された c_i 入力を持ち、

前記第2の大整数ユニットL I U $\{1\}$ は第1の大整数ユニットL I U $\{0\}$ の c_a 出力に結合された c_i 入力を持ち、

前記 n 番目の大整数ユニットL I U $\{n-1\}$ は論理0の値に結合された第2の加算器入力を持ち、

30

1よりも大きい i に対し、前記複数の大整数ユニットのそれぞれの i 番目の大整数ユニットL I U $\{i\}$ は $\{i-1\}$ 番目の大整数ユニットL I U $\{i-1\}$ の c_o 出力に結合された c_i 入力を持ち、 n 番目の大整数ユニットL I U $\{n-1\}$ の c_o 出力は第5のデータ信号、キャリーフロー信号 c_y であることを特徴とする装置。

【請求項20】

第1のデータ信号 y を第2のデータ信号 x とともに処理するマルチプレシジョン整数演算用の装置であって、

コンローラと、該コンローラに結合された大整数ユニット(L I U)とを備え、該L I Uが、

40

第1の乗算器入力、第2の乗算器入力、上位部分積出力、および下位部分積出力を有する乗算器と、

第1の加算器入力、第2の加算器入力、第3の加算器入力、第4の加算器入力、および加算器出力を有し、前記第3の加算器入力の下位部分積出力を受け取るように乗算器に結合された加算器と、

L I Uレジスタ入力およびL I Uレジスタ出力を有するL I Uレジスタであって、前記L I Uレジスタ入力は加算器出力を受け取るように加算器に結合され、前記第1のL I Uレジスタはクロックパルスの前に加算器出力の値をラッチしてクロックパルスの後にその値を前記L I Uレジスタ出力に転送し、前記L I Uレジスタ出力はL I Uレジスタ上位出力とL I Uレジスタ下位出力に分割され、前記L I Uレジスタ上位出力は前記L I Uレジ

50

スタ出力の上位2ビットを含み、前記L I Uレジスタ下位出力は前記L I Uレジスタ上位出力のビット以外の前記L I Uレジスタ出力の全ビットを含み、前記L I Uレジスタ上位出力は第4の加算器入力に結合され、前記第1のL I UレジスタはさらにL I Uレジスタクリア入力がアクティブとされたときに前記L I Uレジスタ出力を0にクリアするための前記L I Uレジスタクリア入力を有する第1のL I Uレジスタと、

乗算器の上位部分積出力を受け取るための第2のL I Uレジスタと、

第1のL I Uレジスタに結合されて、L I Uレジスタ上位出力を受け取って、L I Uレジスタ上位出力のいずれかのビットがアクティブのときにアクティブとされるキャリーフロー信号c yを発生する論理ORゲートと、

前記大整数ユニットの第1の乗算器入力に第1のデータ信号yが印加され、

前記大整数ユニットの第2の乗算器入力に、複数の要素を持つ第2のデータ信号xが印加され、

前記大整数ユニットの第1の加算器入力に結合され、減算信号がアクティブのときに第3のデータ信号zの補数をとるための第1の相補ゲートと、

前記大整数ユニットのL I Uレジスタ下位出力に結合され、前記減算信号がアクティブのときに前記L I Uレジスタ下位出力の補数をとるための、第4のデータ信号aを発生する第2の相補ゲートとを備える

ことを特徴とするマルチプレシジョン整数演算用の装置。

【請求項21】

第1のデータ信号yを第2のデータ信号xとともに処理するマルチプレシジョン整数演算用の装置であって、

大整数ユニット(L I U)であって、

第1の乗算器入力、第2の乗算器入力、上位部分積出力、および下位部分積出力を有する乗算器と、

第1の加算器入力、第2の加算器入力、第3の加算器入力、第4の加算器入力、および加算器出力を有し、前記第3の加算器入力の下位部分積出力を受け取るように乗算器に結合された加算器と、

L I Uレジスタ入力およびL I Uレジスタ出力を有するL I Uレジスタであって、前記L I Uレジスタ入力は加算器出力を受け取るように加算器に結合され、前記L I Uレジスタはクロックパルスの前に加算器出力の値をラッチしてクロックパルスの後にその値を前記L I Uレジスタ出力に転送し、前記L I Uレジスタ出力はL I Uレジスタ上位出力とL I Uレジスタ下位出力に分割され、前記L I Uレジスタ上位出力は前記L I Uレジスタ出力の上位2ビットを含み、前記L I Uレジスタ下位出力は前記L I Uレジスタ上位出力のビット以外の前記L I Uレジスタ出力の全ビットを含み、前記L I Uレジスタ上位出力は第4の加算器入力に結合され、前記L I UレジスタはさらにL I Uレジスタクリア入力にアクティブとされたときに前記L I Uレジスタ出力を0にクリアするための前記L I Uレジスタクリア入力を有するL I Uレジスタと、

L I Uレジスタに結合されて、L I Uレジスタ上位出力を受け取って、

L I Uレジスタ上位出力のいずれかのビットがアクティブのときにアクティブとされるキャリーフロー信号c yを発生する論理ORゲートと

を有する大整数ユニットを備え、

前記大整数ユニットの第1の乗算器入力に第1のデータ信号yが印加され、

前記大整数ユニットの第2の乗算器入力に第2のデータ信号xが結合され、

前記大整数ユニットの第1の加算器入力に結合され、減算信号がアクティブのときに第3のデータ信号zの補数をとるための第1の相補ゲートと、

前記大整数ユニットL I U {0}のL I Uレジスタ下位出力に結合され、前記減算信号がアクティブのときに前記L I Uレジスタ下位出力の補数をとるための、第4のデータ信号aを発生する第2の相補ゲートと、

ラッチレジスタ入力とラッチレジスタ出力を有するラッチレジスタであって、前記ラッチレジスタ入力は前記大整数ユニット L I U の上位部分積出力に結合され、前記ラッチレジスタ出力は、クロックパルスの前に上位部分積の値をラッチし、クロックパルスの後に上位部分積の値を前記大整数ユニットの第 2 の加算器入力に供給するための前記大整数ユニットの第 2 の加算器入力に結合されたラッチレジスタとを備え、
 前記ラッチレジスタはラッチレジスタクリア信号を持ち、また前記大整数ユニットは L I U レジスタクリア入力を持ち、前記 L I U レジスタクリア入力と前記ラッチレジスタクリア信号と一緒に結合されてデータ信号 c l r を形成し、
 前記ラッチレジスタはクロック入力を持ち、また前記大整数ユニットはクロック入力を持ち、前記クロック入力と一緒に結合されてクロック信号 c l k を形成することを特徴とする装置。

10

【発明の詳細な説明】

【0001】

(発明の背景)

本発明は、大きな整数を扱う算術計算に関し、さらに詳細には、大きな整数の乗算およびモジュラ簡約 (modular reduction) を実行する手段に関する。

【0002】

(関連技術の説明)

本発明は、RSA Laboratories 著「The Public-Key Cryptography Standards」(RSA Data Security, Inc., Redwood City, CA, 1993 年 11 月) の RSA 方式、W. Diffie および M. Hellman 著「New Directions in Cryptography」(IEEE Transactions on Information Theory, Vol. IT-22, No. 6, 1977 年 6 月, pp. 74~84) の Diffie-Hellman 方式、および「Digital Signature Standard」(FIPS Publication 186, National Institute of Standards and Technology, U. S. Department of Commerce, 1994 年 5 月 18 日) の DSA 方式などの公開鍵暗号法の応用が動機となっている。これらの応用では、モジュラべき乗の計算ステップ、つまり、A、B、および P をマルチプレシジョン整数とした場合に「 $A^B \bmod P$ 」の形の計算を行う必要がある。モジュラべき乗 (modular exponentiation) は、計算量の多い処理手順であるため、特に安価な計算装置で実行する場合

20

30

【0003】

計算装置は通常、固定サイズの基本単位でデータを処理するように設計されている。この基本データ単位のことを「ワード」と呼び、「b」ビットで構成される。b の値は、計算装置の設計時に選択する。b の値としては、8、16、32、または 64 ビットが標準である。ワードは、範囲 $0 \leq x \leq W-1$ (ただし $W = 2^b$) の非負整数 x を表現するために使用される。

【0004】

$W-1$ よりも大きな整数 X を表現するには、マルチプレシジョン表現を使用する。 $0 \leq X \leq W^n - 1$ で、n をある正整数とした場合、X は次式で表すことができる。

40

【0005】

$$X = x_{\{n-1\}} * W^{n-1} + x_{\{n-2\}} * W^{n-2} + \dots + x_{\{1\}} * W + x_{\{0\}},$$
 ただし、各 $x_{\{j\}}$ は、範囲 $0 \leq x_{\{j\}} \leq W-1$ ($j = 0, 1, \dots, n-1$) 内にあるものとする。したがって、整数 X は n 個のワード $x_{\{0\}}, x_{\{1\}}, \dots, x_{\{n-1\}}$ で表され、ワード $x_{\{j\}}$ は W^j の係数である整数を表す。与えられたワードサイズ b と、与えられたワード数 n について、X のマルチプレシジョン表現は一意的である。

【0006】

50

モジュラべき乗は、マルチプレシジョン整数に対するいくつかの演算を必要とする。このような演算の1つに乗算がある。それぞれ n ワードのマルチプレシジョン表現を持つ2つの整数 X および Y が与えられた場合、積 $Z = X * Y$ の計算を計算装置にプログラムする作業は簡単である。この処理手順は、中学校で教わる多桁数の乗算の「積算および加算」法と大差ない。この処理手順の複雑さは2次であり、つまり、計算しなければならない「内積」の数が n^2 である。積 Z は、表現するのに $2n$ 個のワードを必要とする。

【0007】

実際の公開鍵アプリケーションでは、1回のモジュラべき乗の計算にこのような乗算を数百回あるいは数千回実行する必要がある、それぞれの乗算では数十あるいは数百ワード長の整数を扱うのである。単純な計算装置だと、1回のモジュラべき乗の計算時間は数分程度である。電子商取引など多くのアプリケーションでは、これは長すぎて使えない。

10

【0008】

乗算は、モジュラべき乗に関わる唯一の演算ではない。他に、モジュラ簡約、つまり、「 P を法とする(modulo P)」演算を適用することにより積 Z を $2n$ 個のワードから n 個のワードに減らす演算がある。モジュラ簡約の複雑度は、乗算のそれとほとんど同じであり、2つともモジュラべき乗の総計算時間に対する寄与はほぼ同じである。

【0009】

(発明の概要)

本発明では、計算装置に以下の演算を効率よく実行する能力があれば、マルチプレシジョン乗算とモジュラ簡約(modular reduction)を高速に計算できるという観察結果を利用

20

【0010】

$Z \leftarrow Z \pm (X * Y)$

この演算で、 X は m ワードで表される整数であり、 Y と Z はそれぞれ n ワードで表される整数である。つまり、 n はモジュラべき乗の計算で使用する法におけるワードの個数で、 m は n よりも小さな値である。マルチプレシジョン乗算およびモジュラ簡約は、それぞれこの演算を約 n/m 回行うことにより実行でき、モジュラべき乗の計算に対しては合計 $2n/m$ 回の演算となる。

【0011】

本発明は、計算装置に追加することで、計算装置がこの演算を効率よく実行することができるようにするコプロセッサを開示する。計算装置から命令が送られると、コプロセッサは、 X 、 Y 、および Z のワードをメモリから読取り、 Z の新しいワードをメモリに書き込むことにより演算 $Z \leftarrow Z \pm (X * Y)$ を実行する。計算装置は、これらの演算をマルチプレシジョン乗算、モジュラ簡約、そして最終的に、モジュラべき乗演算にアセンブルする。

30

【0012】

このコプロセッサの基本ブロックは大整数ユニット(LIU; large integer unit)である。コプロセッサは、1つまたは複数のLIUを備えており、それぞれ同じものである。各LIUは、乗算器、加算器、レジスタ、およびORゲートを含む。複数のLIU同士を接続して、相補ゲート、ラッチレジスタ、および出力ゲートを備える1つのLIUアレイを構成できる。アレイ内に含まれるLIUの個数が多いほど、べき乗計算は高速化される。

40

【0013】

本発明の追加の目的および利点は、一部は以下の説明で示し、一部は説明から明らかであるか、または本発明を実施することで知ることができる。本発明の目的および利点はまた、添付の特許請求の範囲で特に指摘した手段および組合せを用いて実現し達成することができる。

【0014】

(好ましい実施形態の詳細な説明)

本発明のこの好ましい実施形態をここで詳しく参照するが、その例は添付図面に示されて

50

おり、複数の図面にわたって類似の参照番号は類似の要素を示すものとする。

【0015】

この説明全体を通して、「X～nワード」という表記は、前記の形式によるマルチプレシジョン表現を使って整数Xが表現されていることを意味する。

【0016】

また、いくつかの演算子を導入すると便利である。演算子「 $\div$ 」は、剰余を無視する整数除算を表すのに使用する。演算子「 $\%$ 」は、整数剰余演算、つまり除算の結果求められる剰余を表すのに使用する。この表記を使うと、 $X \div W = x_{\{n-1\}} * W^{n-2} + x_{\{n-2\}} * W^{n-3} \dots + x_{\{2\}} * W + x_{\{1\}}$ 、また $X \% w = x_{\{0\}}$ となる。

【0017】

本発明は、計算装置に追加することで、計算装置がこの演算を効率よく実行することができるようにするコプロセッサを開示する。計算装置11におけるコプロセッサ14の配列を図1に示す。コプロセッサ14は、計算装置がメモリ12との通信に使用するバスを共有する。バスは、計算装置のワードサイズbとマッチする。計算装置から命令が送られると、コプロセッサは、X、Y、およびZのワードをメモリ12から読取り、Zの新しいワードをメモリ12に書き込むことにより演算 $Z \leftarrow Z \pm (X * Y)$ を実行する。計算装置は、これらの演算をマルチプレシジョン乗算、モジュラ簡約(modular reduction)、そして最終的に、モジュラべき乗演算にアセンブルする。

【0018】

図2は大整数ユニット(LIU)21を示す図である。コプロセッサ14はn個のLIUを含んでおり、それぞれは互いに同一である。LIUの個数により、演算 $Z \leftarrow Z \pm (X * Y)$ のオペランド内のワード数Xが決まる。モジュラべき乗の実行時間は、 $1/m$ にだいたい比例する。システム設計者向けに、LIUを増やせばべき乗計算が高速化されるという、複雑さと速度とのトレードオフ関係を考慮したスケーラブルなアーキテクチャを提供している。

【0019】

大整数ユニット(LIU)

図2に示されている大整数ユニット(LIU)21では、1ビット幅の信号を破線で、複数ビット幅の信号を実線で示しており、ビット数が明記されている。ビット数は、他の実施形態では異なる場合があり、本発明はそのような別の実施形態をも含めるものとする。

【0020】

第1の乗算器入力m1と第2の乗算器入力m2は、大整数ユニット21への被乗数入力である。第1の乗算器入力m1と第2の乗算器入力m2はそれぞれ、bビット整数を表すワードである。第1の乗算器入力m1と第2の乗算器入力m2は乗算器22に送られ、積信号 $p = m1 * m2$ が出力されるが、ただし「 $*$ 」は乗算を表すものとする。

【0021】

$0 \leq m1 \leq (W-1)$ および $0 \leq m2 \leq (W-1)$ なので、積信号pの範囲は $0 \leq p \leq (W-1)^2$ である。つまり、積信号pを表現するには2bビット必要だということである。積信号pは、上位部分積信号(high product signal)phと下位部分積信号(low product signal)plに分けられる。上位部分積信号phは、積信号pの上位bビットとして、下位部分積信号plは下位bビットとして構成される。したがって、 $p = ph * W + pl$ ($0 \leq ph \leq (W-1)$ かつ $0 \leq pl \leq (W-1)$) である。上位部分積信号phは大整数ユニット21の出力信号であり、下位部分積信号plは内部で使用される。

【0022】

第1の加算器信号a1と第2の加算器信号a2は、大整数ユニット21への加算器入力である。それぞれbビット整数を表すワードである。第1の加算器信号a1と第2の加算器信号a2は加算器23に送られる。加算器23はさらに、下位部分積信号plと、2ビットの値であるキャリー信号cを受け取る。加算器23は、第1の総和信号qd ($qd = a1 + a2 + pl + c$) を出力する。第1の加算器信号a1、第2の加算器信号a2、また

10

20

30

40

50

は下位部分積信号 p_1 の可能な最大値は $W-1$ である。キャリー信号 c に使用可能な最大の値は 3 である。したがって、第 1 の総和信号 q_d の最大可能値は $3 * (W-1) + 3 = 3W$ であり、総和信号 q_d を表現するには $b+2$ ビットが必要である。

【0023】

加算器 23 の出力の第 1 の総和信号 q_d は、レジスタ 24 にラッチされる。レジスタ 24 は $b+2$ ビット幅である。第 1 の総和信号 q_d の値は、大整数ユニット 21 へのクロック clk 入力 が脈動されるとレジスタ 24 の出力に転送される。

【0024】

レジスタ 24 の出力は第 2 の総和信号 q である。第 2 の総和信号 q の値は次回クロック clk 入力 が脈動されるまで、または大整数ユニット 21 のクリア clr 入力 が脈動されるまで適所にとどまる。

10

【0025】

レジスタ 24 は、大整数ユニット 21 の clr 入力信号を脈動することでクリアすることができる。 clr 入力 が脈動されると、レジスタ 24 の出力ビットは 0 にクリアされる。出力は、 clk 入力 が次回脈動されるまで 0 のままであり、このときに、第 1 の総和信号 q_d の値が出力へ転送される。

【0026】

レジスタ 24 の出力の第 2 の総和信号 q は、キャリー信号 c と総和信号 s に分けられる。キャリー信号 c は、第 2 の総和信号 q の上位 2 ビットとして、総和信号 s は下位 b ビットとして構成される。したがって $q = cW + s$ であり、 $0 \leq s \leq W-1$ 、 $0 \leq c \leq 3$ である。総和信号 s は大整数ユニット 21 の出力信号である。

20

【0027】

キャリー入力信号 ci は大整数ユニット 21 へのキャリー入力であり、キャリー出力信号 co はキャリー出力である。それぞれ 1 ビットの値である。キャリー出力信号 co は、OR ゲート 29 を使用して、キャリー入力信号 ci と第 2 の総和信号 q を構成するビットの論理 OR として計算される。つまり、キャリー出力信号 co は、キャリー入力信号 ci が 0 のときと、第 2 の総和信号 q の各ビットが 0 のときに値が 0 となる。そうでない場合には、キャリー出力信号 co は値 1 を持つ。

【0028】

補助キャリー信号 ca は、OR ゲート 25 を使用し、キャリー信号 c を構成する 2 ビットの論理 OR として計算される。つまり、補助キャリー信号 ca は、キャリー信号 c の各ビットが 0 の場合に値が 0 であり、そうでない場合には、補助キャリー信号 ca は値 1 となる。

30

【0029】

L I U アレイ

説明

図 3 に示されているように、複数の大整数ユニットを相互接続して 1 つの L I U アレイを形成することができる。それとは別に、L I U アレイには L I U を 1 つだけ含めることができる。図 3 では、各信号は 1 ビット幅または b ビット幅のいずれかである。1 ビット幅の信号は破線で示されており、 b ビット幅の信号は実線で示されている。

40

【0030】

図 3 は、4 つの大整数ユニット $L I U \{0\}$ 、 $L I U \{1\}$ 、 $L I U \{2\}$ 、 $L I U \{3\}$ を相互接続して 1 つの L I U アレイ 39 を形成する場合を示している。一般に、大整数ユニットはいくつでも相互接続できる。簡単のため、ここでの説明ではまず図 4 に示されているように 4 つの大整数ユニットの場合について考察し、その後、他の数の大整数ユニットの場合に一般化する。

【0031】

第 1 のデータ信号 y は L I U アレイ 39 への入力であり、それぞれの大整数ユニット $L I U \{0\}$ 、 $L I U \{1\}$ 、 $L I U \{2\}$ 、 $L I U \{3\}$ の第 1 の乗算器入力 m_1 に印加される。

50

【0032】

第2のデータ信号は複数の要素 $x\{0\}$ 、 $x\{1\}$ 、 $x\{2\}$ 、および $x\{3\}$ を持つ。各要素はそれぞれ、L I Uアレイ39の各大整数ユニットの第2の乗算器入力 m_2 に接続される、つまり各要素は、それぞれL I U $\{0\}$ 、L I U $\{1\}$ 、L I U $\{2\}$ 、L I U $\{3\}$ の第2の乗算器入力 m_2 に接続される。

【0033】

第3のデータ信号 z はL I Uアレイ39への入力であり、相補ゲート35の入力に接続されている。この相補ゲート35は排他OR (XOR) ゲートGATE0であり、第3のデータ信号 z の各ビットを減算信号と組み合わせる。この減算信号自体はL I Uアレイ39への入力信号である。減算信号が非アクティブ状態、つまり論理0であれば、第3のデータ信号 z のビットは、相補ゲート35の通過中に影響を受けない。減算信号がアクティブ状態、つまり論理1であれば、第3のデータ信号の各ビットは相補ゲート35を通るときに論理相補ビットで置き換えられる。つまり、1のビットは0のビットで、0のビットは1のビットで置き換えられるということである。相補ゲート35の出力は、第1の大整数ユニットL I U $\{0\}$ の第1の加算器入力 a_1 に接続されている。

10

【0034】

出力信号 a はL I Uアレイ39の出力であり、出力ゲート36の出力から取り出される。出力ゲート36は排他OR (XOR) ゲートGATE1であり、第1の大整数ユニットL I U $\{0\}$ の出力である総和信号 s の各ビットを組み合わせる。減算信号が非アクティブ状態であれば、第1大整数ユニットの総和信号 s のビットは出力ゲート36を通るときに影響を受けない。減算信号がアクティブ状態であれば、第1大整数ユニットの総和信号 s のビットは、その論理相補ビットで置き換えられる。

20

【0035】

ラッチレジスタ34は b ビット幅のレジスタREG0である。ラッチレジスタ34の入力は第4の大整数ユニットL I U $\{3\}$ の上位部分積信号 p_h に接続され、ラッチレジスタ34の出力は第4の大整数ユニットL I U $\{3\}$ の第2の加算器信号 a_2 入力に接続されている。そこで、ラッチレジスタ34は、1クロックパルス前に第4の大整数ユニットから上位部分積信号 p_h の値をラッチし、その値をそのクロックパルスの後に第4の大整数ユニットの第2の加算器信号 a_2 入力に戻す。ラッチレジスタ34は、図2の大整数ユニット21のレジスタ24の clr 入力とまったく同じ動作をするクリア clr 入力を持つ。

30

【0036】

クロック信号 clk はL I Uアレイ39への入力であり、それぞれの大整数ユニットL I U $\{0\}$ 、L I U $\{1\}$ 、L I U $\{2\}$ 、L I U $\{3\}$ の clk 入力とラッチレジスタ34の clk 入力に接続される。クロック入力が脈動されると、それぞれの大整数ユニット内のレジスタの内容とラッチレジスタ34の内容は同時に更新される。

【0037】

クリア信号 clr はL I Uアレイ39への入力であり、それぞれの大整数ユニットL I U $\{0\}$ 、L I U $\{1\}$ 、L I U $\{2\}$ 、L I U $\{3\}$ の clr 入力とラッチレジスタ34の clr 入力に接続される。クリア入力が脈動されると、それぞれの大整数ユニット内のレジスタの内容とラッチレジスタ34の内容は同時に0にクリアされる。

40

【0038】

キャリーフロー信号 cy はL I Uアレイ39の出力であり、第4の大整数ユニットL I U $\{3\}$ の co 出力に接続される。このキャリーフロー信号は、オーバーフローまたはアンダーフローが発生したかどうかを示す。

【0039】

4つよりも多い大整数ユニットを1つのL I Uアレイに相互接続するには、第3の大整数ユニットの相互接続ロジックを複製する。以下の表1に、 m 個 ($m \geq 4$) の大整数ユニットを1つのL I Uアレイに相互接続するロジックの詳細をまとめた。

【0040】

50

【表 1】

 $m \geq 4$

L I U {k}: 信号	k = 0	k = 1	k = 2, 3, ..., m - 2	k = m - 1
m 1	y	y	y	y
m 2	x {0}	x {1}	x {k}	x {m-1}
c i	論理 0	LIU{0}: c a	LIU{k-1}: c o	LIU{m-2}: c o
a 1	GATE0:出力	LIU{0}: p h	LIU{k-1}: p h	LIU{m-2}: p h
a 2	LIU{1}: s	LIU{2}: s	LIU{k+1}: s	REG0: 出力
c a	LIU{1}: c i	N/C	N/C	N/C
c o	N/C	LIU{2}: c i	LIU{k+1}: c i	c y
p h	LIU{1}: a 1	LIU{2}: a 1	LIU{k+1}: a 1	REG0: 入力
s	GATE1:入力	LIU{0}: a 2	LIU{k-1}: a 2	LIU{m-2}: a 2

10

【0041】

20

表1で、「L I U {k} : x」という表記はL I U {k} の信号x、「R E G 0 : x」はR E G 0の信号xなどを意味し、「N/C」は信号が未接続であることを意味している。一番左の欄は k^{ch} の大整数ユニットL I U {k} の信号を示し、後の欄は $k = 0, 1, \dots, m-1$ に対するその信号の相互接続ロジックを示している。図からわかるように、表1の記載事項は $m = 4$ のときの図3に示されている相互接続ロジックとマッチしている。

【0042】

4つよりも少ない大整数ユニットを1つのL I Uアレイに相互接続するには、大整数ユニットのいくつかの相互接続ロジックを集約する。表2、3、および4に、それぞれ $m = 3$ 、 $m = 2$ 、および $m = 1$ とした場合の m 個の大整数ユニットを1つのL I Uアレイに相互

30

【0043】

【表 2】

m = 3

L I U{k} : 信号	k = 0	k = 1	k = 2
m 1	y	y	y
m 2	x { 0 }	x { 1 }	x { 2 }
c i	論理 0	LIU{0} : c a	LIU{1} : c a
a 1	GATE0 : 出力	LIU{0} : p h	LIU{1} : p h
a 2	LIU{1} : s	LIU{2} : s	REG0 : 出力
c a	LIU{1} : c i	N / C	N / C
c o	N / C	LIU{2} : c i	c y
p h	LIU{1} : a 1	LIU{2} : a 1	REG0 : 入力
s	GATE1 : 入力	LIU{0} : a 2	LIU{1} : a 2

10

20

【 0 0 4 4 】

【表 3】

m = 2

L I U{k} 信号	k = 0	k = 1
m 1	y	y
m 2	x { 0 }	x { 1 }
c i	論理 0	LIU{0} : c a
a 1	GATE0 : 出力	LIU{0} : p h
a 2	LIU{1} : s	REG0 : 出力
c a	LIU{1} : c i	N / C
c o	N / C	c y
p h	LIU{1} : a 1	REG0 : 入力
s	GATE1 : 入力	LIU{0} : a 2

30

40

【 0 0 4 5 】

【表 4】

$$m = 1$$

L I U{ <i>k</i> } 信号	$k = 0$
<i>m</i> 1	<i>y</i>
<i>m</i> 2	$x \{0\}$
<i>c</i> <i>i</i>	論理 0
<i>a</i> 1	GATE0 : 出力
<i>a</i> 2	REG0 : 出力
<i>c</i> <i>a</i>	<i>c</i> <i>y</i>
<i>c</i> <i>o</i>	N / C
<i>p</i> <i>h</i>	REG0 : 入力
<i>s</i>	GATE1 : 入力

10

20

【0046】

表2～4に記載されていないL I Uアレイ信号は、*m*の任意の値に対するのと同じ相互接続ロジックを持つ。完全を期すため、表5に信号の詳細を示す。

【0047】

【表5】

信号	接続
<i>c</i> <i>l</i> <i>r</i>	REG0 : <i>clr</i> および LIU{ <i>k</i> } : <i>c</i> <i>l</i> <i>r</i> , $k = 0, 1, \dots, m - 1$
<i>c</i> <i>l</i> <i>k</i>	REG0 : <i>clr</i> および LIU{ <i>k</i> } : <i>c</i> <i>l</i> <i>r</i> , $k = 0, 1, \dots, m - 1$
<i>z</i>	GATE0 : 入力
<i>s</i> <i>u</i> <i>b</i>	GATE0 : 入力, GATE1 : 入力
<i>a</i>	GATE1 : 出力

30

【0048】

演算

L I Uアレイによって実行される一般関数は以下のとおりである。

【0049】

$$A \leftarrow Z \pm (X * Y)$$

ただし、*X*～*m*ワード、*Y*～*n*ワード、*Z*～*n*ワード、そして*A*～*n*ワードである。

【0050】

整数*X*、*Y*、および*Z*は、関数に対する入力オペランドであり、整数*A*は出力オペランドである。*X*のワードはL I Uアレイに同時に、入力 $x \{0\}$ 、 $x \{1\}$ 、 $\dots$ 、 $x \{m - 1\}$ において印加される。*Y*および*Z*のワードは一度に1ワードがL I Uアレイに、それぞれ入力 y および z において印加される。*A*のワードはL I Uアレイから一度に1ワードが、出力*a*において読み出される。

50

【0051】

L I Uアレイへの減算入力は、「±」演算において減算か加算かを選択する。減算入力をアクティブにする、つまり減算入力を論理1に設定することで減算を選択し、減算入力を非アクティブにする、つまり減算入力を論理0にクリアすることで加算を選択する。

【0052】

L I Uアレイのキャリフロー信号 $c y$ は、オーバーフローまたはアンダーフローが発生したかどうかを示す。加算中に計算した値が大きすぎて n ワードで表現できないとオーバーフローが発生する。減算中に計算した値が負になるとアンダーフローが発生する。演算終了時にオーバーフローまたはアンダーフローが発生していれば、キャリフロー信号 $c y$ 出力はアクティブになる。オーバーフローもアンダーフローも発生していなければ、キャリフロー信号 $c y$ 出力は非アクティブである。

10

【0053】

L I Uアレイは、以下の処理手順に従って一般関数を実行する場合に使用できる。

1. $c l r$ を脈動する。
 2. 信号 $s u b$ をアクティブまたは非アクティブにする。
 3. データ入力 $x \{0\}$, $x \{1\}$, . . . , $x \{m-1\}$ を印加する。
 4. $j = 0, 1, . . . , n-1$ について、
 - a. データ入力 y を印加する。
 - b. データ入力 z を印加する。
 - c. $c l k$ を脈動する。
 - d. データ出力 a を読み取る。
- を実行する。

20

【0054】

ステップ1で、L I Uレジスタとラッチレジスタ34 (REG0) をクリアする。前の演算からの累計がクリアされる。

ステップ2で、「±」演算における加算と減算を選択する。減算入力为非アクティブ値であれば加算が選択され、アクティブ値であれば減算が選択される。

ステップ3で、Xオペランドのワードの値が印加される。これらの値は、そのまま全体の処理手順に印加される。

ステップ4で、 n 回実行するループを制御する。それぞれの反復には、インデックス j ($j = 0, 1, . . . , n-1$) で番号が振られる。

30

ステップ4aで、オペランドYのワード j を印加する。

ステップ4bで、オペランドZのワード j を印加する。

ステップ4cで、信号 $c l k$ を脈動する。これによりL I Uレジスタとラッチレジスタ34 (REG0) を新しい値で更新し、出力 a において新しいワードを使用できるようにする。

ステップ4dで、オペランドAのワード j を読み出す。

【0055】

L I Uアレイのクロッキングの分析

L I Uアレイの動作は、単一のサイクル、つまりクロック入力信号の単一の脈動 (pulsing) について分析することができる。例では、4つのL I Uからなるアレイのケースを使用する。この分析をどのように一般化して他の多数のL I Uに適用できるかは明白であろう。

40

【0056】

この説明では、L I Uアレイへの減算入力信号が非アクティブとなっているケースのみを分析する。アクティブな減算信号のケースについては、後ほど分析する。

【0057】

図3と図2を参照すると、整数 Q は値 $Q = q \{0\} + q \{1\} W + q \{2\} W^2 + q \{3\} W^3 + r w^4$ を持つと定義され、ここで、 $q \{j\}$ はL I U $\{j\}$ のレジスタに格納されている値、 r はL I UアレイのレジスタREG0に格納されている値である。それぞれの

50

$q \{j\}$ は $b + 2$ ビットで構成され、 $W = 2^b$ である。

【0058】

クロック信号が脈動する (pulsed) と、 Q の値が変化する。新しい値を Q^+ と呼ぶ。 Q^+ の値は、それぞれの $L I U$ 内の加算器の出力および $L I U \{3\}$ 内の乗算器の出力から導かれる。重ね合わせを使って、次のようにして Q^+ の値を決定することができる。

【0059】

まず、 Q の値は 0、入力信号 y の値も 0 であると仮定する。この場合、 Q^+ に寄与するのは入力信号 z だけである。 z の値は、 $L I U \{0\}$ 内の加算器を伝播して、 $L I U \{0\}$ のレジスタ出力に現れる。 $L I U$ アレイ内の他のレジスタはそれぞれ出力値が 0 である。したがって、この場合、次のようになる。

【0060】

$$Q^+ = z. \quad (i)$$

【0061】

次に、 Q の値は 0、入力信号 z の値は 0 であると仮定する。この場合、 Q^+ への寄与は各 $L I U$ 内の乗算器の出力からだけである。乗算器出力の下位部分は同じ $L I U$ 内のレジスタに寄与する。乗算器出力の上位部分は次に高い $L I U$ 内のレジスタ、すなわち $L I U \{3\}$ の場合には、 $REG 0$ に寄与する。したがって、 Q^+ の値に寄与する場合には、 $L I U \{j\}$ 内の乗算器出力の下位部分は W^j で重み付けされ、上位部分は W^{j+1} で重み付けされる。これは、 Q^+ に寄与する場合には $L I U \{j\}$ 内の乗算器の全出力が W^j で重み付けされることを意味している。 $L I U \{j\}$ 内の乗算器への 2 つの入力はそれぞれ入力信号 y および $x \{j\}$ なので、この場合、次のようになる。

【0062】

$$Q^+ = y x \{0\} + y x \{1\} W + y x \{2\} W^2 + y x \{3\} W^3.$$

(i i)

【0063】

次に、 z の値は 0、 y の値は 0 であると仮定する。この場合、 Q^+ への寄与は Q の現在値からだけである。 $L I U \{j\}$ 内では、 $q \{j\}$ の値は $c \{j\} W + s \{j\}$ と記述でき、ここで、 $c \{j\}$ は $q \{j\}$ の上位 2 ビットから、 $s \{j\}$ は残りのビットからできている。そこで、 Q の値は次のように書き下せる。

【0064】

$$Q = s \{0\} + (c \{0\} + s \{1\}) W + (c \{1\} + s \{2\}) W^2 + (c \{2\} + s \{3\}) W^3 + r W^4.$$

【0065】

それぞれの $c \{j\}$ は $L I U \{j\}$ 内の新しいレジスタ値に寄与し、 Q^+ に寄与するときの W^j の重み付けがされる。 $j = 1, 2$ 、および 3 について、各 $s \{j\}$ は $L I U \{j-1\}$ 内のレジスタに寄与し、 W^{j-1} の重みを付ける。 r の値は $L I U \{3\}$ 内のレジスタに寄与し、 W^3 の重みを付ける。したがって、この場合、次のようになる。

【0066】

$$Q^+ = (c \{0\} + s \{1\}) + (c \{1\} + s \{2\}) W + (c \{2\} + s \{3\}) W^2 + r W^3.$$

【0067】

これは次のように書けることに注意されたい。

【0068】

$$Q^+ = Q / W, \quad (i i i)$$

【0069】

ただし、「 $/$ 」は、剰余を無視する整数除算を表す。

式 (i)、(i i)、および (i i i) は、新しい値 Q^+ へのすべての寄与分を含む。重ね合わせによる寄与分を組み合わせると、次式が得られる。

【0070】

$$Q^+ = z + y (x \{0\} + x \{1\} W + x \{2\} W^2 + x \{3\} W^3) + Q / W.$$

(i v)

【 0 0 7 1 】

クロック脈動 (clock pulse) 後、L I Uアレイの出力信号 a の値は次式で与えられる。

【 0 0 7 2 】

$$a^+ = Q^+ \% W. \quad (v)$$

【 0 0 7 3 】

a^+ はクロック脈動後の a の値を意味し、「%」は整数モジュロ演算を表す。つまり、 a^+ は、 Q^+ を W で除算したときの剰余である。

【 0 0 7 4 】

式 (i v) と (v) は、クロックが脈動したときの L I Uアレイの演算をまとめたものである。 10

【 0 0 7 5 】

一般関数を計算するための手続の分析

このセクションでは、一般関数の計算を行うために使用される処理手続中の L I Uアレイの演算を分析する。この処理手続については前のセクションで説明した。説明からわかるように L I Uアレイは、この手続とともに使用したときに一般関数の計算を正しく行える。

【 0 0 7 6 】

この説明では、L I Uアレイへの減算入力信号が非アクティブとなっているケースのみを分析する。アクティブな減算信号のケースについては、後ほど分析する。 20

【 0 0 7 7 】

一般関数は $A \leftarrow (Z + X * Y) \% W^n$ である。ただし、

$$Z = z \{0\} + z \{1\} W + z \{2\} W^2 + \dots + z \{n-1\} W^{n-1};$$

$$Y = y \{0\} + y \{1\} W + y \{2\} W^2 + \dots + y \{n-1\} W^{n-1};$$

$$X = x \{0\} + x \{1\} W + x \{2\} W^2 + \dots + x \{m-1\} W^{m-1};$$

$$A = a \{0\} + a \{1\} W + a \{2\} W^2 + \dots + a \{n-1\} W^{n-1};$$

【 0 0 7 8 】

ただし、m は、L I Uアレイ内の L I U の個数およびオペランド X におけるワード数であり、n は、オペランド Y、Z および A におけるワード数である。モジュロ演算「% W^n 」は、 $Z + X * Y$ の結果の値が A の n ワードに収まるように切り詰められるという事実を反映している。 $(Z + X * Y) < W^n$ であれば $A = Z + X * Y$ であり、これは、一般関数の「オーバーフローなし」のケースである。 $(Z + X * Y) \geq W^n$ であれば、A は $Z + X * Y$ の下位 n ワードからなり、これは、一般関数の「オーバーフロー」のケースである。 30

【 0 0 7 9 】

一般関数を計算する処理手続については前のセクションで説明した。この処理手続を L I Uアレイの前の分析とを組み合わせることにより、一般関数の計算を次の反復 (recursion) で記述できる。

【 0 0 8 0 】

For $k = 0, 1, \dots, n-1$:

$$Q \{k\} = z \{k\} + y \{k\} X + S \{k-1\};$$

$$a \{k\} = Q \{k\} \% W;$$

$$S \{k\} = Q \{k\} / W;$$

【 0 0 8 1 】

ただし、 $S \{-1\} = 0$ である。「%」は、整数モジュロ演算を表し、「/」は剰余を切り捨てる整数除算を表す。

【 0 0 8 2 】

n ステップ反復が完了したら、以下の式の成り立つことが明らかになる。

【 0 0 8 3 】

$$a \{0\} + a \{1\} W + a \{2\} W^2 + \dots + a \{n-1\} W^{n-1} + S \{n-1\} W^n$$

=

$$z \{0\} + z \{1\} W + z \{2\} W^2. \dots + z \{n-1\} W^{n-1} \\ + X (y \{0\} + y \{1\} W + y \{2\} W^2. \dots + y \{n-1\} W^{n-1}). \\ (v i)$$

【0084】

この式の右辺は $Z + X * Y$ である。式の左辺は $A + S \{n-1\} W^n$ である。「 $\% W^n$ 」演算を式の両辺に適用すると、 $A = (Z + X * Y) \% W^n$ が得られる。したがって、式 (v i) が真であることが明らかにされると、処理手続によって実際に L I U アレイに一般関数を計算させていることが明らかになる。

【0085】

式 (v i) は帰納法によって証明できる。まず、 $n = 1$ について式が成り立つことが示される。次に、 $n = t$ について式が成り立てば、 $n = t + 1$ についても成り立つことが示される。これで、 $n \geq 1$ の任意の値について式が成り立つことがいえる。

10

【0086】

$n = 1$ のケースについて考察する。 $S \{-1\} = 0$ なので、反復から次式が得られる。

【0087】

$$Q \{0\} = z \{0\} + y \{0\} X.$$

【0088】

$S \{k\}$ は商であり、 $a \{k\}$ は $Q \{k\}$ を W で割って得た剰余であるために、任意の k について、 $W S \{k\} + a \{k\} = Q \{k\}$ であることに注意されたい。 $k = 0$ においてこれを評価すると、次式が得られる。

20

【0089】

$$a \{0\} + W S \{0\} = Q \{0\} = z \{0\} + y \{0\} X,$$

【0090】

これは、 $n = 1$ のケースについて式 (v i) を定める。

【0091】

ここで、 $n = t + 1$ のケースについて考察する。 $n = t$ について式 (v i) が成立すると仮定すると、次式が得られる。

【0092】

$$a \{0\} + a \{1\} W + a \{2\} W^2. \dots + a \{t-1\} W^{t-1} + S \{t-1\} W^t \\ = \\ z \{0\} + z \{1\} W + z \{2\} W^2. \dots + z \{t-1\} W^{t-1} \\ + X (y \{0\} + y \{1\} W + y \{2\} W^2. \dots + y \{t-1\} W^{t-1}). \\ (v i i)$$

30

【0093】

$W S \{k\} + a \{k\} = Q \{k\}$ なので、次式が得られる。

【0094】

$$W^{t+1} S \{t\} + W^t a \{t\} = W^t Q \{t\} = \\ W^t (z \{t\} + y \{t\} X + S \{t-1\}). \quad (i i i)$$

【0095】

式 (v i i) と (i i i) を足し合わせると次式が得られる。

40

【0096】

$$a \{0\} + a \{1\} W + a \{2\} W^2. \dots + a \{t\} W^t + S \{t\} W^{t+1} \\ = \\ z \{0\} + z \{1\} W + z \{2\} W^2. \dots + z \{t\} W^t \\ + X (y \{0\} + y \{1\} W + y \{2\} W^2. \dots + y \{t\} W^t).$$

【0097】

これにより、 $n = t + 1$ について式 (v i) が成立し、証明が完成する。

【0098】

アクティブな減算信号の分析

L I U アレイの減算入力信号がアクティブになっている場合、オペランド Z の各ワードは

50

L I Uアレイに印加される前に相補がとられる。数値についてみると、これはZを($W^n - Z - 1$)で置き換えることと等価である。同様に、オペランドAの各ワードはL I Uアレイから出力される前に相補がとられる。これはAを($W^n - A - 1$)で置き換えることと等価である。したがって、減算信号がアクティブのときは、一般関数は次のように計算される。

【0099】

$$Z' = W^n - Z - 1;$$

$$A' = (Z' + X * Y) \% W^n;$$

$$A = W^n - A' - 1.$$

$0 \leq Z' < W^n$ および $0 \leq A' < W^n$ なので、「 $\% W^n$ 」演算をZ'およびA'の式に自由に付加することができる。

10

【0100】

$$Z' = (W^n - Z - 1) \% W^n;$$

$$A' = (Z' + X * Y) \% W^n;$$

$$A = (W^n - A' - 1) \% W^n.$$

この形式では、次のように、「 $\% W^n$ 」演算を無視し、式をマージし、「 $\% W^n$ 」演算を末尾に付加できることが明白である。

【0101】

$$A = (W^n - ((W^n - Z - 1) + X * Y) - 1) \% W^n.$$

この式を簡単になると、次式が得られる。

20

【0102】

$$A = (Z - X * Y) \% W^n.$$

これは、望みどおりの一般関数の減算形式である。

【0103】

シングルRAM装置

L I Uアレイをコントローラおよびランダムアクセスメモリ(RAM)と組み合わせて、上述の一般関数に基づき計算を実行できる装置を構成することができる。このような装置を図4に示した。

【0104】

コントローラ41とL I Uアレイ39の組み合わせでは、計算装置とともに使用することができるコプロセッサを備える。このような場合、ランダムアクセスメモリ(RAM)42は、図1に示されているように、計算装置11と共有することができる。

30

【0105】

コントローラ41は、L I Uアレイ39の信号を処理し、望む演算をアレイに実行させる。コントローラ41は、RAM42から入力オペランドを読み出してL I Uアレイ39と計算装置に転送する。コントローラ41は、L I Uアレイ39から出力オペランドを読み出してRAM42に書き込む。コントローラ41を実施する方法は多数あるが、その一つに有限状態機械(finite state machine)を使用するものがある。RAM42は従来の装置である。このRAMはbビット幅のデータバスを持ち、これはL I Uアレイ39のワードサイズと一致している。コントローラ41は、適切な値をアドレスバスに印加し、読み取り信号をアクティブにし、データバス上でRAM42によって供給されたとおりにワードの値を読み取り、それから読み取り信号を非アクティブにすることにより、RAM42からワードを読み取る。コントローラ41は、適切な値をアドレスバスに印加し、ワードの値をデータバスに印加し、書き込み信号を脈動させることによりRAM42にワードを書き込む。

40

【0106】

ワードを読み取る、またはワードを書き込むのに要する時間のことをRAMサイクルと呼ぶ。たとえば、コントローラ41がnワードからなる整数Xを読み取ったり書き込んだりするときにかかる時間はn RAMサイクルである。

【0107】

50

コントローラ 41 は、装置に、次の処理手順を用いて一般関数 $A \leftarrow Z \pm (X * Y)$ を実行させるが、ここで、 $X \sim m$ ワード、 $Z \sim n$ ワード、および $A \sim n$ ワードである。

【0108】

1. コントローラ 41 は L I U アレイ 39 の clr 入力を脈動する。
2. コントローラは L I U アレイ 減算入力をアクティブまたは非アクティブにする。コントローラは、この処理手順の残りの間、この値を一定に保つ。
3. コントローラは RAM 43 から m 個の値を読み出して、L I U アレイの $x \{0\}$, $x \{1\}$, . . . , $x \{m-1\}$ 入力に印加する。コントローラは、この処理手順の残りの間、これらの値を一定に保つ。
4. $j = 0, 1, \dots, n-1$ について、
 - a. コントローラは、RAM から値を読み取って L I U アレイの y 入力に印加する。
 - b. コントローラは、RAM から値を読み取って L I U アレイの z 入力に印加する。
 - c. コントローラは L I U アレイの clk 入力を脈動する。
 - d. コントローラは、L I U アレイの出力の値を読み取って RAM に書き込む。

10

【0109】

ステップ 3 で、コントローラ 41 は RAM 42 から X オペランドの m ワードを読み取る。これに、 m RAM サイクルを要する。
 ステップ 4 a で、コントローラ 41 は RAM 42 から Y オペランドの n ワードを読み取る。これに、 n RAM サイクルを要する。
 ステップ 4 b で、コントローラ 41 は RAM 42 から Z オペランドの n ワードを読み取る。これに、 n RAM サイクルを要する。
 ステップ 4 d で、コントローラ 41 は RAM 42 に A オペランドの n ワードを書き込む。これに、 n RAM サイクルを要する。

20

【0110】

この処理手順に必要な RAM サイクルの総数は $m + 3n$ である。これは、ステップ 3、4 a、4 b、および 4 d におけるサイクル数を足し合わせることで求められる。

【0111】

読取り—変更—書込みのあるシングル RAM 装置

図 5 に示されているように、一般関数を実行するのに必要な RAM サイクル数は、読取り—変更—書込み動作を実行できる RAM 43 をを使用して減らすことができる。このような動作により、コントローラ 41 は RAM 43 から 1 ワード読み取って、そのワードの値を変更し、新しい値を RAM 43 に書き戻す作業すべてを 1 RAM サイクル内に実行することができる。

30

【0112】

コントローラ 41 と L I U アレイ 39 の組み合わせでは、計算装置とともに使用することができるコプロセッサを備える。このような場合、RAM 43 は、図 1 に示されているように計算装置 11 と共有することができる。

【0113】

コントローラ 41 は次のようにして、RAM 43 に対し読取り—変更—書込み動作を実行する。まず、コントローラは適切な値をアドレスバスに印加し、読取り—変更—書込み信号 rmw をアクティブにする。次に、コントローラは読取り信号をアクティブにし、データバス上に RAM が出した値を読取り、読取り信号を非アクティブにする。さらに、コントローラは変更された値をデータバスに印加し、書込み信号を脈動する。最後に、コントローラは rmw 信号を非アクティブにする。この動作全体を 1 RAM サイクル内に実行する。

40

【0114】

RAM 43 はさらに、すでに説明したように、通常の読み書き演算も実行することができる。コントローラは、 rmw 信号を非アクティブにすることにより、通常の読み書き演算を選択する。この 3 つの演算（読取り、書込み、および読取り—変更—書込み）はそれぞれ、1 RAM サイクルを必要とする。

50

【0115】

読取り－変更－書込み機能を利用する場合には、一般関数は $Z \leftarrow Z \pm (X * Y)$ となり、ここで、 $X \sim m$ ワード、 $Y \sim n$ ワード、および $Z \sim n$ ワードである。したがって、オペランド Z を変更するには、量 $X * Y$ の加算または減算を行う。コントローラ41は、以下の処理手順に従って装置にこの関数（functon）を実行させる。

【0116】

1. コントローラ41はL I Uアレイ39のc l r入力を脈動する。
2. コントローラはL I Uアレイ減算入力をアクティブまたは非アクティブにする。コントローラは、この処理手順の残りの間、この値を一定に保つ。
3. コントローラはRAM42から m 個の値を読み出して、L I Uアレイの $x \{0\}$ 、 $x \{1\}$ 、 $\dots$ 、 $x \{m-1\}$ 入力に印加する。コントローラは、この処理手順の残りの間、これらの値を一定に保つ。
4. $j = 0, 1, \dots, n-1$ について、
 - a. コントローラは、RAMから値を読み取ってL I Uアレイの y 入力に印加する。
 - b. コントローラは、RAMから値を読み取って、L I Uアレイの Z 入力に印加し、L I Uアレイのc l k信号を脈動し、L I Uアレイの a 出力の値を読み取って、RAMに書き込む。

10

【0117】

ステップ3で、コントローラはRAMから X オペランドの m ワードを読み取る。これに、 m RAMサイクルを要する。
 ステップ4 aで、コントローラはRAMから Y オペランドの n ワードを読み取る。これに、 n RAMサイクルを要する。
 ステップ4 bで、コントローラはRAMの Z オペランドの n ワードのそれぞれについて読取り－変更－書込み演算を実行する。これに、 n RAMサイクルを要する。

20

【0118】

この手処理順に必要なRAMサイクルの総数は $m + 2n$ である。これは、ステップ3、4 a、および4 bにおけるサイクル数を足し合わせることで求められる。

【0119】

デュアルRAM装置

一般関数を実行するのに必要なRAMサイクル数は、第2のRAMを図6に示されているように装置に追加することによりさらに減らすことができる。「Z-RAM」と呼ばれる第1のRAM44は、 Z オペランドを格納する。Z-RAM44は読取り－変更－書込み機能を備える。「Y-RAM」と呼ばれる第2のRAM45は、 Y オペランドを格納する。Y-RAM45は読取り－変更－書込み機能を備えている必要はない。デュアルRAMを使用すると、コントローラ41は Y オペランドの1ワードの読取りと Z オペランドの1ワードの読取り－変更－書込みを同時に実行できる。この同時動作は1 RAMサイクルタイムで実行される。

30

【0120】

コントローラ41、L I Uアレイ39、およびZ-RAM44の組み合わせは、図1の計算装置とともに使用することができるコプロセッサを備える。このような場合、Y-RAM45は、図1に示されているように計算装置11と共有することができる。

40

【0121】

デュアルRAM装置は、次の処理手順に従って一般関数 $Z \leftarrow Z \pm (X * Y)$ を実行し、ここで、 $X \sim m$ ワード、 $Y \sim n$ ワード、および $Z \sim n$ ワードである。

【0122】

1. コントローラ41はL I Uアレイ39のc l r入力を脈動する。
2. コントローラはL I Uアレイ減算入力をアクティブまたは非アクティブにする。コントローラは、この処理手順の残りの間、この値を一定に保つ。
3. コントローラはY-RAM45またはZ-RAM44から m 個の値を読み出して、L I Uアレイの $x \{0\}$ 、 $x \{1\}$ 、 $\dots$ 、 $x \{m-1\}$ 入力に印加する。コントローラ

50

は、この処理手順の残りの間、これらの値を一定に保つ。

4. $j = 0, 1, \dots, n-1$ について、

a. コントローラは、Y-RAMから値を読み取ってL I UアレイのY入力に印加し、Z-RAMから値を読み取ってL I UアレイのZ入力に印加し、L I Uアレイのclock信号を脈動、L I Uアレイのa出力の値を読み取ってZ-RAMに書き込む。

【0123】

ステップ3で、コントローラはRAMからXオペランドのmワードを読み取る。これに、m RAMサイクルを要する。

ステップ4 aで、コントローラはRAMからYオペランドのnワードを読取り、RAMからZオペランドのnワードのそれぞれについて読取り－変更－書込み演算を実行する。これに、n RAMサイクルを要する。

10

【0124】

この処理手順に必要なRAMサイクルの総数は $m+n$ である。これは、ステップ3および4 aにおけるサイクル数を足し合わせることで求められる。

【0125】

パイプライン化大整数ユニット

L I U内のクリティカルパスは、乗算－累算演算のときにアレイ内の第1の $n-1$ 個のL I Uにおいて発生する。ここで、乗算器の部分積信号phはa1の次の高い順位のL I Uの加算器に入る。このパスは加算器を通り、総和信号sとしてそのL I Uを抜けて、第2の加算器信号a2を通して元のL I Uに戻る。

20

【0126】

ロジックをパイプライン化すると、L I Uの動作周波数を上げることができる。パイプライン化L I Uを図7に示す。他のレジスタREG1を導入することにより、クリティカルパスに沿った遅延が減少する。新しいレジスタは、乗算器の出力の上位部分積信号上に置かれる。これにより、クリティカルパスは、乗算器の遅延と加算の遅延を足したものからちょうど乗算器遅延に減少される。

【0127】

このレジスタの結果として、アレイの相互接続が変わる。パイプライン化L I Uを備えたL I Uアレイを図8に示す。各L I Uの上位部分積信号phは、a1において自身にフィードバックすることに注意されたい。信号phは、第1の加算器信号a1で1サイクルだけ遅れる。

30

【0128】

当業者であれば、乗算器も同様にパイプライン化できることを理解するであろう。それぞれのパイプライン段により待ち時間のサイクルが加わるが、動作周波数を高めるだけではこれを補正することはできない。実際、これらのレジスタを最適に配置する電子設計自動化ソフトウェアが入手可能である。コントローラを調整して $X*Y$ の計算を進め、 $A \leftarrow Z \pm (X*Y)$ が正しく計算されるようにできる。

【0129】

デュアルポートRAM装置

乗算－累算(multiply-accumulate)演算以外の設計におけるクリティカルパスは、読取り－変更－書込み演算である。このパスは、シングルポートRAMに代えてデュアルポートRAMを使用することで低減できる。これは、完全デュアルポートRAMではなく、むしろ単一書込みポート単一読取りポートRAMであることに注意されたい。このタイプのRAMによって、1つのアドレスから読み取ることができると同時に、一方で異なるアドレスに書き込むことができる。図9は、改良した装置の図である。

40

【0130】

デュアルポートRAMを使用することにより、読取り－変更－書込み演算を3つの部分に分けられる。図10にこれを示した。第1の部分は、読取り演算($Z\{i\}$)、第2の部分は変更演算であり、 $Z\{i-1\} = Z\{i-1\} + / - X\{i-1\} * Y\{i-1\}$ が形成される。第3の部分は、 $A\{i-2\}$ のRAMへの書き戻しである。読取りアドレス

50

は決して書込みアドレスと等しくならないので、それらの間の差は常にちょうど2である。このため、aを書き込んでも、zで読み取っているデータに影響しない。このようにして、3つの部分はそれぞれ、完了するサイクル全体を持ち、最高動作周波数をかなり改善する。

【0131】

実現可能な別の手段として、変更および書込み演算を単一のサイクルにまとめるという方法がある。これにより、待ち時間は1サイクルだけ短縮し、いくつかの種類のRAMについては、RAMに書き込むデータがそのサイクルの最後の部分についてのみ通常有効でなければならない程度に高速化することができる。ここでもやはり、読取りアドレスと書込みアドレスが異なるという要件が満たされている。

10

【0132】

特定の算術演算

前のセクションでは、YとZをnワードのオペランド、Xをmワードのオペランド、そしてmをLIUアレイ内のLIUの個数以下とした場合に、一般関数 $Z \leftarrow Z \pm (X * Y)$ を実行するために装置を使用する方法について説明した。この関数は、マルチプレシジョン演算およびモジュラ簡約で使用する算術演算を実行するように特殊化させることができる。特殊化は、コントローラの、LIUアレイの制御信号およびデータ信号処理によって行われる。

【0133】

以下の説明では、このような算術演算の例をいくつか取りあげる。これらの例は、実行可能な演算のうちのごくわずかに過ぎず、可能なすべての算術演算を代表しているわけではない。

20

乗算：この装置を使用して、次の計算を実行することができる。

【0134】

$Z \leftarrow X * Y$ 、

【0135】

ただし、 $X \sim m$ ワード、 $Y \sim k$ ワード、 $Z \sim n$ ワード、 $n \geq k$ である。

【0136】

$X * Y$ の値が値が大きすぎてnワード表現に収まらない場合にオーバーフローが発生する。 $n \geq k + m$ であれば、 $X * Y$ は常に収まる。 $n < k + m$ であれば、XおよびYの値に応じて $X * Y$ の値は収まる場合も収まらない場合もある。計算の終わりににおけるLIUアレイのcy出力の値は、オーバーフローが発生したかどうかを示す。

30

【0137】

シングルRAM装置を使用する処理手順を以下に示す。

【0138】

1. コントローラはLIUアレイのclr入力を脈動する。
2. コントローラはLIUアレイの減算入力を非アクティブにし、LIUアレイのz入力に0を印加する。コントローラは、この処理手順の残りの間、これらの値を一定に保つ。
3. コントローラはRAMからm個の値を読み出して、LIUアレイの $x\{0\}$, $x\{1\}$, ..., $x\{m-1\}$ 入力に印加する。コントローラは、LIUアレイの他のすべての $x\{ \}$ 入力に0を印加する。コントローラは、この処理手順の残りの間、これらの値を一定に保つ。
4. $j = 0, 1, \dots, k-1$ について、
 - a. コントローラは、RAMから値を読み取って、LIUアレイのy入力にその値を印加する。
 - b. コントローラはLIUアレイのclk入力を脈動する。
 - c. コントローラは、LIUアレイの出力の値を読み取って、RAMに書き込む。
5. コントローラは、LIUアレイのy入力に0を印加する。コントローラは、この処理手順の残りの間、この値を一定に保つ。
6. $j = k, k+1, \dots, n-1$ について、

40

50

- a. コントローラはL I Uアレイのc l k入力を脈動する。
 b. コントローラは、L I Uアレイの出力の値を読み取って、R A Mに書き込む。

【0139】

ステップ4 cおよび6 bにおいてR A Mに書き込んだ値は、望む結果のzのワードである。

【0140】

この処理手順によって正しい答えが得られることを調べるために、それが一般関数 $Z \leftarrow Z \pm (X * Y')$ を計算することと等価であることを観察する。ここで、Zの初期値は0のnワード表現、Y'はYのnワード表現である。

【0141】

読取り－変更－書込み装置を持つシングルR A M装置の処理手順は同一である。読取り－変更－書込み機能は、Zオペランドが書込みのみであって読取りを行わないため、利点がない。

【0142】

デュアルポートR A M装置の処理手順は、ステップ4 aがステップ4 cと同時に実行される点を除いて類似している。

【0143】

各装置に必要なR A Mサイクルの数を表6にまとめた。

【0144】

【表6】

	シングルR A M	シングルR A M w / r m w	デュアルR A M
S t e p 3	m	m	m
S t e p 4	2 k	2 k	k
S t e p 6	n - k	n - k	n - k
合計	m + n + k	m + n + k	m + n

【0145】

乗算－累算：この装置を使用して、次の計算を実行することができる。

【0146】

$Z \leftarrow Z + (X * Y)$ 、

ここで、 $X \sim m$ ワード、 $Y \sim k$ ワード、 $Z \sim n$ ワード、 $n \geq k$ である。

【0147】

結果の値が大きすぎてnワードに収まらない場合にオーバーフローが発生する。オーバーフローは、計算が終了したときのL I Uアレイのc y出力上のアクティブな値で示される。

【0148】

シングルR A M装置を使用する処理手順を以下に示す。

【0149】

1. コントローラはL I Uアレイのc l r入力を脈動する。
2. コントローラはL I Uアレイ減算入力を非アクティブにする。コントローラは、この処理手順の残りの間、この値を一定に保つ。
3. コントローラはR A Mからm個の値を読み出して、L I Uアレイの $x \{0\}$ 、 $x \{1\}$ 、 $\dots$ 、 $x \{m-1\}$ 入力に印加する。コントローラは、L I Uアレイの他のすべての $x \{ \}$ 入力に0を印加する。コントローラは、この処理手順の残りの間、これらの値を一定に保つ。

4. $j = 0, 1, \dots, k-1$ について、
- a. コントローラは、RAMから値を読み取ってLIUアレイのy入力に印加する。
 - b. コントローラは、RAMから値を読み取ってLIUアレイのz入力に印加する。
 - c. コントローラはLIUアレイのclk入力を脈動する。
 - d. コントローラは、LIUアレイの出力の値を読み取ってRAMに書き込む。
5. コントローラは、LIUアレイのy入力に0を印加する。コントローラは、この処理手順の残りの間、この値を一定に保つ。
6. $j = k, k+1, \dots, n-1$ について、
- a. コントローラは、LIUアレイのcy出力の値を読み取る。値が非アクティブであれば、コントローラは処理手順を終了する。
 - b. コントローラは、RAMから値を読み取ってLIUアレイのz入力に印加する。
 - c. コントローラはLIUアレイのCLK入力を脈動する。
 - d. コントローラは、LIUアレイの出力の値を読み取ってRAMに書き込む。

10

【0150】

ステップ4 cおよび6 dにおいてRAMに書き込んだ値は、望む結果のzのワードである。

【0151】

この処理手順によって正しい答えが得られることを調べるために、一般関数 $Z \leftarrow Z + (X * Y')$ を計算することと同等であることを観察する。ここで、 Y' はYのnワード表現である。

20

【0152】

Zのnワードすべてが新しい値で置き換えられる前にこの処理手順がステップ6 aで終了する場合がある。これは、置換後の値が置換前の値と同一であると処理手順によって認識された場合に発生する。処理手順を終了すると、効果のないサイクルを排除することによりRAMサイクル総数が低減される。

【0153】

読取り－変更－書込み装置を持ったシングルRAMの処理手順は、ステップ4 bと4 dが単一の読取り－変更－書込みサイクルで実行され、ステップ6 bと6 dが単一の読取り－変更－書込みサイクルで実行されることを除いて類似している。

【0154】

デュアルRAM装置の処理手順は、ステップ4 aがステップ4 bおよび4 dの読取り－変更－書込みサイクルと同時に実行されるという点を除いて、読取り－変更－書込み装置を持つシングルRAMと似ている。

30

【0155】

各装置に必要なRAMサイクルの数を表7にまとめた。

【0156】

【表7】

	シングルRAM	シングルRAM w / r m w	デュアルRAM
Step 3	m	m	m
Step 4	3 k	2 k	k
Step 6	$\leq 2(n - k)$	$\leq n - k$	$\leq n - k$
合計	$\leq m + 2n + k$	$\leq m + n + k$	$\leq m + n$

40

【0157】

50

「 $\leq$ 」記号は、ステップ6 aで処理手順が早期に終了する可能性を反映している。

【0158】

乗算－否定－累算：この装置を使用して、次の計算を実行することができる。

【0159】

$Z \leftarrow Z - (X * Y)$ 、

【0160】

ここで、 $X \sim m$ ワード、 $Y \sim k$ ワード、 $Z \sim n$ ワード、 $n \geq k$ である。

【0161】

結果の値が負になるとアンダーフローが発生する。アンダーフローは、計算が終了したときのL I Uアレイのc y出力におけるアクティブな値で示される。

10

【0162】

この処理手順は、減算入力ステップ2でアクティブにされ、非アクティブにはされないという点を除き、乗算－累算 (multiply-accumulate) 演算の処理手順と同一である。この処理手順は、Zのワードについての置換後の値が置換前の値と同じであると認識された場合に早期に終了する。RAMサイクルの数は、乗算－累算演算と同じである。

【0163】

モジュラべき乗への応用

A、B、およびPをそれぞれnワードの整数とする。モジュラべき乗は、 $A^B \% P$ を計算する問題である。モジュラべき乗を実行するよく知られ得ている手法に、モンゴメリの方法がある。P. Montgomery著「Modular Multiplication without Trial Division」Mathematics of Computation, 44 (1985), pp. 519～521。この方法は、L I Uアレイに基づく本装置に適している。

20

【0164】

本特許は、モンゴメリの方法のチュートリアルであることを意図していない。本発明を使用してこの方法を実施する方法を示し、RAMサイクルに関して実施速度を分析することのみを目的としている。

【0165】

モンゴメリの方法は、「モンゴメリ積」と呼ばれる関数に基づいている。Pを法とする2つの整数XとYのモンゴメリ積は、次式で定義される。

30

【0166】

$f(X, Y, P) = ((X * Y * G) \% P) + kP$,

ただし、Gは $(G * W^n) \% P = 1$ を満たす整数で、kは小さな整数である。このような整数Gは、Pが奇数の場合に存在し、Pが偶数の場合には存在しない。したがって、モンゴメリの方法はPが奇数の場合にのみ使用される。

【0167】

「 kP 」という項は、 $f()$ の値が0から $P-1$ の範囲になければならないという要件を反映するために含まれている。これは、後で明確になる理由から望ましいものである。

【0168】

モンゴメリの方法を使用してモジュラべき乗を実行する処理手順は以下のとおりである。

40

【0169】

1. $T \leftarrow W^{2n} \% P$
2. $A \leftarrow f(T, A, P)$
3. $T \leftarrow f(T, 1, P)$
4. For each bit in B, most significant first, do:
 - a. $T \leftarrow f(T, T, P)$
 - b. if the bit in B is 1, $T \leftarrow f(T, A, P)$
5. $T \leftarrow f(T, 1, P)$
6. $T \leftarrow T \% P$

Tはnワードの整数である。この処理手順が終了すると、Tは $A^B \% P$ の値を含む。

50

【0170】

ステップ4におけるループは指数Bのビット毎に反復する。Bがnワードを持ち、各ワードはbビットを持つので、ループはnb回反復する。ステップ4aにおけるモンゴメリ積は、ループの反復毎に計算される。ステップ4bにおけるモンゴメリ積は、指数の対応するビットが1ビットの場合にのみ計算される。指数のビットの半分が1ビットであれば、ステップ4bにおけるモンゴメリ積はループ反復の半分について計算される。この仮定を置くと、ループ中で実行されるモンゴメリ積の総数は $1.5 * nb$ となる。

【0171】

公開鍵暗号法では、指数のビット数が大きく、数百あるいは数千ビットになるのがふつうである。これは、べき乗を計算するにはモンゴメリ積計算を数百あるいは数千回実行する必要があるということを意味する。そのため、べき乗を高速に計算する鍵は、モンゴメリ積を高速で計算することである。

10

【0172】

L I Uアレイに基づく本発明装置は、モンゴメリ積を効率よく計算することができる。次のセクションでは、この実行方法について説明する。

【0173】

ステップ1、2、および3は「計算前」ステップであり、ステップ5および6は「計算後」ステップである。これらのステップは、ステップ4に比べて計算量が非常に少なく済み、無視できるくらいである。

【0174】

この処理手順を管理する責任は、コントローラおよび、装置がコプロセッサとして取り付けられている計算装置に分けることができる。システム設計者は、この責任をどのように分けるかということに関して裁量権がある。極端な場合、コントローラを処理手順のすべてのステップを実行できる十分な情報処理機能を受け持つように設計し、計算装置は単に処理手順を開始し、コントローラにオペランドの記憶場所を提供するだけということも考えられる。また、コントローラは一般関数 $Z \leftarrow Z + (X * Y)$ のみを取り扱い、計算装置は、モンゴメリ積を構築し、上記の処理手順によりモジュラべき乗に組み立てる責を負うように設計することもできる。

20

【0175】

マルチプレジジョン乗算

モンゴメリ積 $f(X, Y, P)$ を計算する第1のステップは、XとYの積を計算することである。この手続は、中学校で教わる通常の「乗算および加算」処理手順である。得られる積は表すのに $2n$ ワードを必要とし、ここで、nはX、Y、およびPのそれぞれのワード数である。

30

【0176】

表記は言い回しを簡単にするためだということ同意できるであろう。Xをnワードの整数とする。表記 $X = (x\{j\}, 0 \leq j < n)$ は、Xの値が $x\{0\} + x\{1\}W + x\{2\}W^2 + \dots + x\{n-1\}W^{n-1}$ であることを意味する。Xの「部分整数(subinteger)」つまり、Xの連続するワードのサブセットから成り立つXの整数部分の演算が行われる。たとえば、整数 $X' = \{x\{j\}, 2 \leq j < 5\}$ は値 $x\{2\} + x\{3\}W + x\{4\}W^2$ を持つXの3ワードの部分整数である。演算 $X' \leftarrow X' + 1$ は $x\{2\}$ を $x\{2\} + 1$ で置き換える($x\{2\}$ は小さく、オーバーフローを起こさないと仮定する)。これは、 $x\{2\}$ がXにおいて W^2 によって重み付けされているためXを W^2 だけ増やす効果を持つ。

40

【0177】

mをL I Uアレイ内のL I Uの個数とする。nはmの倍数であると仮定する。整数の長さは、0ワードを上位側に付加してmの倍数にすることで常に増やすことができるため、一般性を失うことなくこのように仮定することができる。整数Z、 Z_i 、および X_i は次のように定義できる。

【0178】

50

$Z = \{z \{j\}, 0 \leq j \leq 2n\}$;
 $Z_i = \{z \{j\}, i \leq j < i + n + m\}$;
 $X_i = \{x \{j\}, i \leq j < i + m\}$.

【0179】

Zは、計算された積を保持するために使用する $2n + 1$ ワードの整数である。Z整数は、次のセクションで明らかになる理由から、必要な長さよりも1ワードだけ長くなっている。Z_iは、z {i} で始まる $n + m$ ワードを持つZの部分整数である。X_iは、x {i} で始まる m ワードを持つXの部分整数である。

【0180】

XとYの積は、次の処理手順で計算することができる。

10

【0181】

1. $Z \leftarrow 0$
 2. $i \leftarrow 0$
 3. While ($i < n$) do:
 a. $Z_i \leftarrow Z_i + (X_i * Y)$
 b. $i = i + m$

【0182】

ステップ1で、Zを0にクリアする。次にステップ3におけるループで、「乗算および加算」を実行する。ループの反復毎に、Xオペランドの m ワードを処理する。これらのワードはX_iを形成し、これにオペランドYを掛け、累算してZ_iを求める。Z_iは、X_iがX内でオフセットされているワードの数と同じ数だけZ内でオフセットされる。最後の反復の後、Zは望んでいた積 $X * Y$ を含む。

20

【0183】

演算 $Z_i \leftarrow Z_i + (X_i * Y)$ の後のZの値について考察する。 $S_i = \{x \{j\}, 0 \leq j \leq i + m\}$ とすると、Xの $i + m$ ワードが処理されているため、Zの値は $S_i * Y$ である。 $S_i * Y$ は $n + m + i$ ワードに収まる。Z_iは i ワードだけZにオフセットされ、長さは $n + m$ なので、演算 $Z_i \leftarrow Z_i + (X_i * Y)$ はオーバーフローすることがない。

【0184】

ループは、全部で n / m 回反復する。表8は、ループの反復毎にステップ3 aで必要なRAMサイクルの数と、必要なRAMサイクルの総数をまとめたものである。

30

【0185】

【表8】

	シングルRAM	シングルRAM w / r mw	デュアルRAM
Step 3 a	$3m + 3n$	$2m + 2n$	$2m + n$
合計	$3n + 3n^2 / m$	$2n + 2n^2 / m$	$2n + n^2 / m$

40

【0186】

ステップ3 aは、表に示されているよりも少ないサイクル数で済ませることができる。これは、LIUアレイのcy出力が早期終了が受け付けられることを示している場合に発生する。実際には、この効果によるサイクル数の減少は、必要なサイクルの総数のごく数分の1に過ぎない。

【0187】

モンゴメリ簡約 (Montgomery Reduction)

$2n$ ワードの積がZに入ると、モンゴメリ積の第2のステップは、Pを法としてそれを n ワードの値に減らすことである。モンゴメリ簡約は右から左へ作用し、Zの一番右のワード（つまり、最下位ワード）が無効になるようにPの適切な倍数がZに加算される。

50

【0188】

この簡約は、次式を満たす整数 Q を必要とする。

【0189】

$$(Q * P) \% W^m = W^m - 1.$$

【0190】

Q は m ワードの整数である。 P が奇数であれば、このような Q は常に存在する。 Q の計算は実に簡単である。 Q はべき乗計算の最初に一度計算され、その後、べき乗の簡約ステップ毎に使用される。 Q は、一度しか計算されないで、計算量はそれほどでもない。

【0191】

部分整数 Z_i は、乗算ステップで定義したのとは異なる方法で定義することもできる。

【0192】

$$Z_i = (z_{\{j\}}, i \leq j \leq 2n),$$

【0193】

つまり、 Z_i は $z_{\{i\}}$ で始まる Z の部分整数であり、 $z_{\{i\}}$ の左にある Z のすべてのワードを含む。 Z_i は $2n + 1 - i$ ワードからなる。

【0194】

さらに、 m ワードの整数 V を定義する必要もある。 V はスクラッチ変数である。

【0195】

簡約ステップは、次の処理手順を用いて実行することができる。

【0196】

1. $i \leftarrow 0$
2. While ($i < n$) do:
 - a. $V \leftarrow Q * Z_i$
 - b. $Z_i \leftarrow Z_i + (V * P)$
 - c. $i = i + m$

【0197】

ステップ 2 におけるループの反復毎に、ステップ 2 b で Z の m ワードをクリアする。つまり、ワード $z_{\{i\}}$, $z_{\{i+1\}}$, ..., $z_{\{i+m-1\}}$ は 0 の値で終わる。このクリア動作は次の通りである。

【0198】

ステップ 2 a において、 Q に Z_i を掛けて、その結果を V に置く。 Q と V はそれぞれ m ワードの整数であるため、 Z_i の最下位 m ワード（つまり、 $\{z_{\{j\}}, i \leq j < i + m\}$ ）だけが問題であり、 Z_i の他のワードは、いったん m ワードに切り詰められて V に収まるようになると、その結果に影響しない。したがって、ステップ 2 a は、「半乗算」として実行され、 m ワードの 2 つの整数は乗算されるが、積の下位 m ワードのみが計算される。

【0199】

ステップ 2 b において、 V に法 P を掛けて、累算して Z_i とする。ステップ 2 a において計算された V の値では、 Z_i の最下位 m ワードがそれぞれ 0 の値を持つ結果になる。 Z_i への累算はオーバーフローしないことが保証されており、高い確率で、 L I U アレイの $c y$ 出力が非アクティブになるために累算は早期に終了する。終了する正確な時間はデータによって異なるが、 V は m ワードを、 P は n ワードを持つので、終了は通常は Z_i の $n + m$ ワードが累算された後になる。

【0200】

ループは、全部で n / m 回反復する。表 9 は、ループの反復毎にステップ 2 a と 2 b で必要な RAM サイクルの数と、必要な RAM サイクルの総数をまとめたものである。

【0201】

【表 9】

10

20

30

40

	シングルRAM	シングルRAM w / r m w	デュアルRAM
Step 2 a	3 m	3 m	2 m
Step 2 b	3 m + 3 n	2 m + 2 n	2 m + n
合計	6 n + 3 n ² / m	5 n + 2 n ² / m	4 n + n ² / m

10

【0202】

ループの終わりにおいてZ（つまり、 $\{z\{j\}, 0 \leq j < n\}$ ）の下位nワードは値0をとり、Z（つまり、 $z\{2n\}$ ）の最上位ワードは0また1の値をとる。最上位ワードが1であれば、Zの上位ワードから次のようにPを引いてクリアしなければならない。

【0203】

$$Z_n \leftarrow Z_n - (1 * P)$$

【0204】

ここで、本発明装置は、オペランドの1つを1に設定して「乗算—不定—累算」を実行するためにも使用される。この演算は、値0を持つZの最上位ワードを生じることが保証されている。演算が必要かどうか（演算前に $Z\{2n\} = 1$ かどうか）はデータによる。それとは関係なく、関係するRAMサイクルの数は乗算および簡約のループに必要なサイクル数と比べて無視できる。

20

【0205】

この時点で、Zは2nワードの整数であり、最下位nワードは値0を持つ。モンゴメリ積を完了するには、以下を実行する。

【0206】

$$Z \leftarrow Z_n / W^n$$

【0207】

これは、Zの上位nワードを下位nワードに単に右シフトするだけのものである。得られる値はnワードの整数であり、得られる値は所望されたモンゴメリ積である。

30

【0208】

この簡約処理手順ではZがnワード内に収まることを保証しているが、 $Z < P$ であることは保証していない。ステップ4におけるべき乗処理手順のループを目的とするため、Zをnワードに収まるために必要なもの以上にZを簡約することは必要ではない。べき乗処理手順は計算後のステップ6を含み、最終結果をPよりも小さい値に簡約する。

【0209】

総RAMサイクル

乗算および簡約ステップに必要なRAMサイクル数に関する前のセクションの結果を表10と組み合わせると、モンゴメリ積に対する単一の結果が得られる。この値に $1.5nb$ を掛けると、モジュラべき乗を計算するために必要なRAMサイクル総数が求められる。

40

【0210】

【表10】

	シングルRAM	シングルRAM w / r mw	デュアルRAM
モンゴメリ積	$6n(1.5 + n/m)$	$4n(1.75 + n/m)$	$2n(3 + n/m)$
べき乗	$9n^2b(1.5 + n/m)$	$6n^2b(1.75 + n/m)$	$3n^2b(3 + n/m)$

【0211】

10

べき乗の公式は、計算前および計算後のステップの間に、つまり、べき乗処理手順におけるステップ1、2、3、および5に必要なRAMサイクルを無視している。これらのステップはステップ4に比べてRAMサイクル数が非常に少なく済み、無視することができる。

【0212】

オペランドがLIUの個数に比べて大きい場合、 n/m は大きい。これは、実用的な公開鍵暗号法の応用のケースである場合が多い。このような場合、RAMサイクル数は約 Kbn^3/m であり、Kは3つのタイプの装置についてそれぞれ9、6、または3である。したがって、RAMサイクルの総数は $1/m$ に略比例する。たとえば、LIUの個数を倍にすることで、RAMサイクル数は半分程度に減少する。デュアルRAM装置は、シングルRAM装置が必要とするサイクル数のおおよそ $1/3$ を必要とし、デュアルRAM装置は読取り—変更—書込み装置を持つシングルRAMで必要なサイクル数の約半分を必要とする。

20

【0213】

本発明の範囲または精神から逸脱することなく本発明のマルチプレシジョン整数演算用の計算方法と装置にさまざまな変更を印加することができることは当業者であれば明白であり、特許請求の範囲およびその等価物の範囲内にあるとすれば本発明はマルチプレシジョン整数演算用の計算方法と装置の変更および変種をもその対象とすることを意図している。

【図面の簡単な説明】

30

添付の図面は、本明細書に含まれかつその一部をなしており、本発明の好ましい実施形態を示すものであり、説明とともに本発明の原理を説明するのに役立つ。

【図1】 計算装置を有するコプロセッサの図である。

【図2】 大整数ユニット（LIU）を示す図である。

【図3】 相互接続されてLIUアレイを形成する4つのLIUを示すブロック図である。

。

【図4】 コントローラおよびランダムアクセスメモリ（RAM）と組み合わされたLIUアレイを示す図である。

【図5】 コントローラおよび読取り—変更—書込み機能を有するコントローラおよびRAMと組み合わされたLIUアレイを示す図である。

40

【図6】 コントローラ、第1のRAM、および第2のRAMと組み合わされたLIUアレイを示す図である。

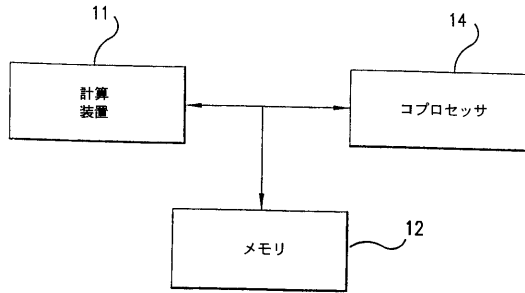
【図7】 パイプライン化大整数ユニット（LIU）を示す図である。

【図8】 相互接続されてLIUアレイを形成する4つのパイプライン化LIUを示すブロック図である。

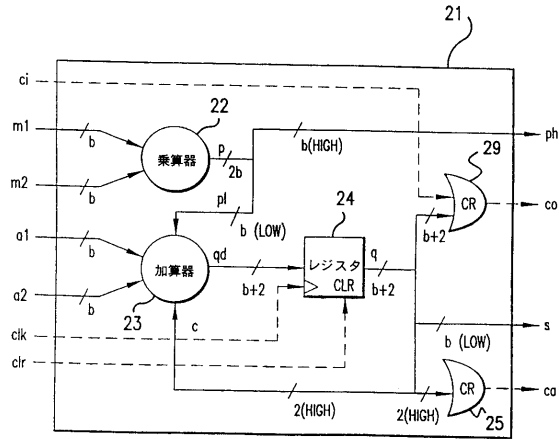
【図9】 デュアルポートRAM装置を示す図である。

【図10】 デュアルポートRAMとともに使用するための読取り—変更—書込み装置を示すブロック図である。

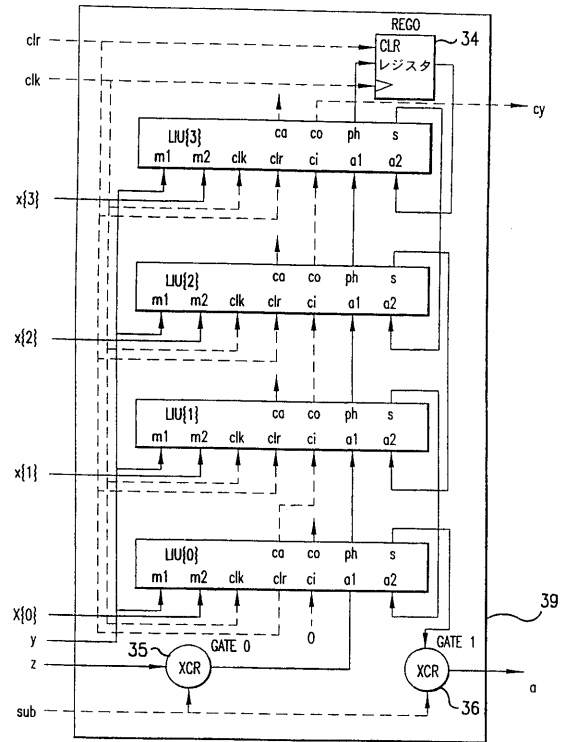
【図 1】



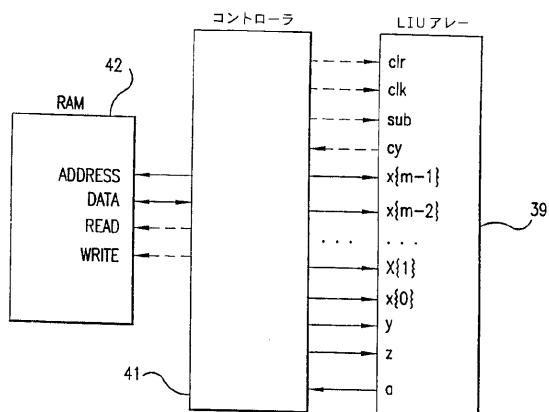
【図 2】



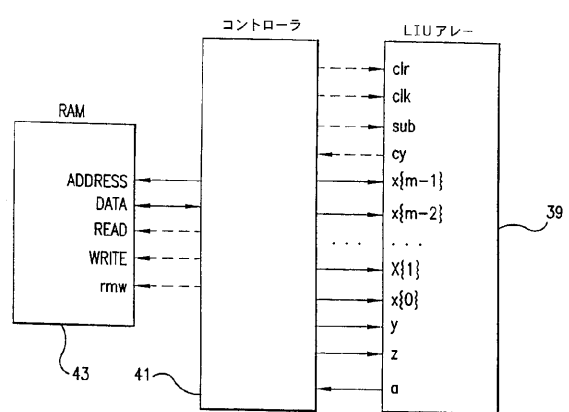
【図 3】



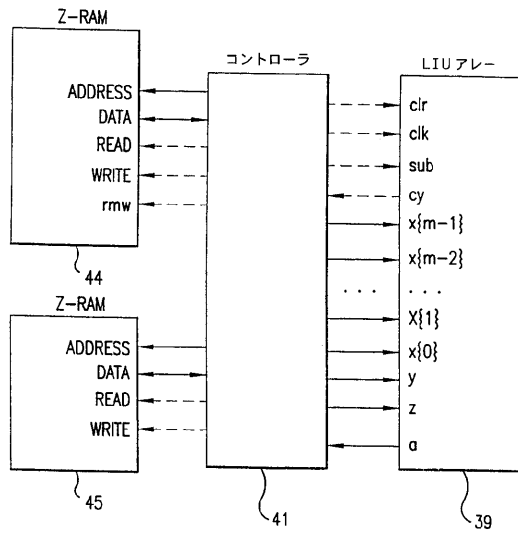
【図 4】



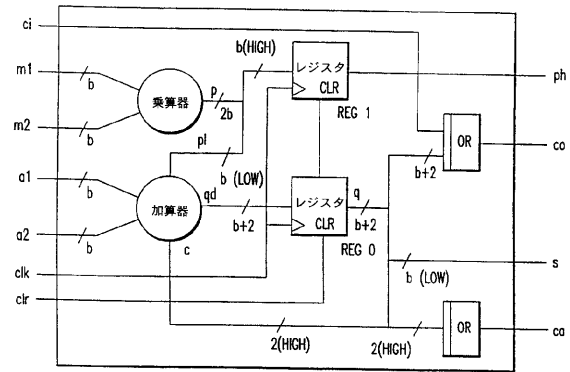
【図 5】



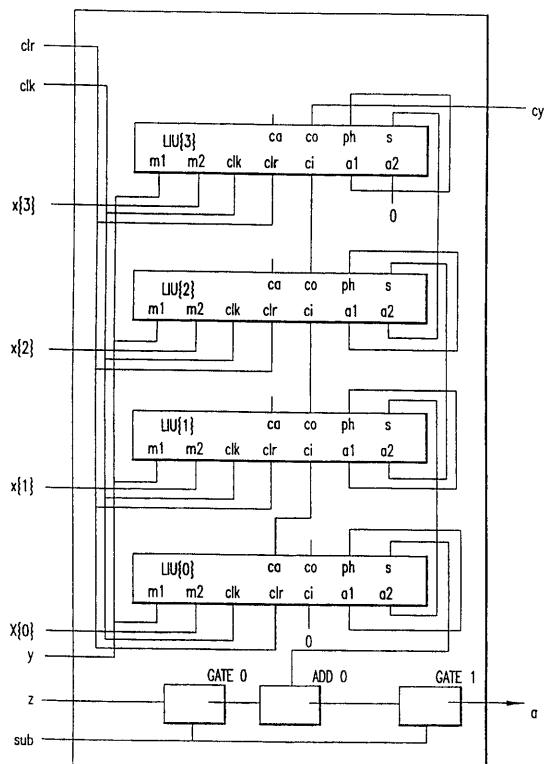
【図 6】



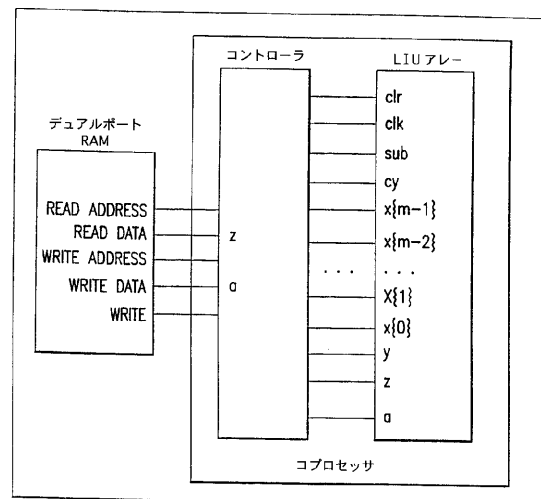
【図 7】



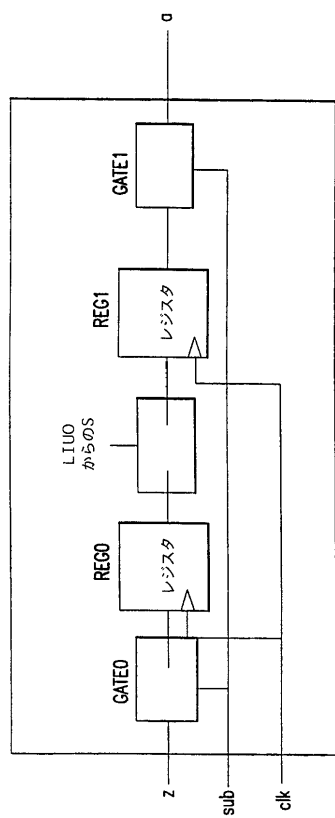
【図 8】



【図 9】



【図10】



フロントページの続き

(72)発明者 ヘイシン マーク ダブリュー.

アメリカ合衆国 94301 カリフォルニア州 パロ アルト イースト クレセント ドライ
ブ 557

審査官 青木 重徳

(56)参考文献 特表2000-503146 (JP, A)

特開平06-012229 (JP, A)

特開昭62-097060 (JP, A)

(58)調査した分野(Int.Cl., DB名)

G09C 1/00