(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号

特許第4170267号
(P4170267)

(45)発行日　平成20年10月22日(2008.10.22)　(24)登録日　平成20年8月15日(2008.8.15)

(51)Int.Cl.　　F I
G09C 1/00 (2006.01)　　G09C 1/00 650A
G06F 7/72 (2006.01)　　G06F 7/72

請求項の数 17　(全 16 頁)

(21)出願番号　特願2004-203435(P2004-203435)
(22)出願日　平成16年7月9日(2004.7.9)
(65)公開番号　特開2006-23647(P2006-23647A)
(43)公開日　平成18年1月26日(2006.1.26)
審査請求日　平成17年8月16日(2005.8.16)

(73)特許権者　302062931
ＮＥＣエレクトロニクス株式会社
神奈川県川崎市中原区下沼部１７５３番地
(73)特許権者　899000068
学校法人早稲田大学
東京都新宿区戸塚町１丁目１０４番地
(74)代理人　100123788
弁理士　宮崎　昭夫
(74)代理人　100106138
弁理士　石橋　政幸
(74)代理人　100127454
弁理士　緒方　雅昭

最終頁に続く

(54)【発明の名称】乗算剰余演算器及び情報処理装置

(57)【特許請求の範囲】
【請求項１】
被乗数をＡ、ｕとし、乗数をＢ、Ｎとし、乗算剰余演算結果をＳとしたとき、Ｓ＝Ｓ＋Ａ×Ｂ＋ｕ×Ｎを算出するための乗算剰余演算器であって、
複数のビット数ｑ単位で供給される前記乗数Ｂの値に応じて前記被乗数Ａまたは０の値を切換えて出力し、前記ビット数ｑ単位で供給される前記乗数Ｎの値に応じて前記被乗数ｕまたは０の値を切換えて出力するセレクタと、
前記セレクタから順次出力される値を用いてＡ×Ｂ＋ｕ×Ｎの演算を実行する桁上げ保存加算器と、
前記桁上げ保存加算器から前記ビット数ｑ単位で出力される前記Ａ×Ｂ＋ｕ×Ｎの演算結果と、前記ビット数ｑ単位で供給される過去の該演算結果とを加算し、該加算結果を前記乗算剰余演算結果Ｓとして出力する加算器と、
を有する乗算剰余演算器。
【請求項２】
前記被乗数Ａを保持し、前記セレクタに供給する第１の記憶素子と、
前記被乗数ｕを保持し、前記セレクタに供給する第２の記憶素子と、
前記乗数Ｂを保持し、前記セレクタに前記ビット数ｑ単位で供給する第３の記憶素子と、
前記乗数Ｎを保持し、前記セレクタに前記ビット数ｑ単位で供給する第４の記憶素子と、

前記加算器から出力される前記乗算剰余演算結果Ｓを保持し、前記ビット数ｑ単位で該乗算剰余演算結果Ｓを前記加算器に供給する第５の記憶素子と、
をさらに有する請求項１記載の乗算剰余演算器。
【請求項３】
前記桁上げ保存加算器の動作を制御する制御部をさらに有する請求項１または２記載の乗算剰余演算器。
【請求項４】
前記制御部は、
前記第１の記憶素子に前記被乗数Ａをセットし、
前記第２の記憶素子に前記被乗数ｕをセットし、
前記第３の記憶素子に前記乗数Ｂをセットし、
前記第４の記憶素子に前記乗数Ｎをセットし、
前記セレクタに０を供給する請求項３記載の乗算剰余演算器。
【請求項５】
予め算出された、前記被乗数Ａ、前記乗数Ｂ、前記乗数Ｎ、及び前記乗算剰余演算結果Ｓの値に対する前記被乗数ｕの値の関係が格納されるｕ生成部をさらに有し、
前記制御部は、
前記Ｓ＝Ｓ＋Ａ×Ｂ＋ｕ×Ｎの演算時に前記ｕ生成部を参照することで前記被乗数ｕの値を決定する請求項３または４記載の乗算剰余演算器。
【請求項６】
前記ビット数ｑは２である請求項１乃至５のいずれか１項記載の乗算剰余演算器。
【請求項７】
前記ビット数ｑは４である請求項１乃至５のいずれか１項記載の乗算剰余演算器。
【請求項８】
前記第１の記憶素子及び前記第２の記憶素子はラッチ回路である請求項２乃至７のいずれか１項記載の乗算剰余演算器。
【請求項９】
前記第３の記憶素子、前記第４の記憶素子及び前記第５の記憶素子はシフトレジスタである請求項２乃至８のいずれか１項記載の乗算剰余演算器。
【請求項１０】
請求項１に記載の乗算剰余演算器と、
前記被乗数Ａを保持し、前記セレクタに供給する第１の記憶素子と、
前記被乗数ｕを保持し、前記セレクタに供給する第２の記憶素子と、
前記乗数Ｂを保持し、前記セレクタに前記ビット数ｑ単位で供給する第３の記憶素子と、
前記乗数Ｎを保持し、前記セレクタに前記ビット数ｑ単位で供給する第４の記憶素子と、
前記加算器から出力される前記乗算剰余演算結果Ｓを保持し、前記ビット数ｑ単位で該乗算剰余演算結果Ｓを前記加算器に供給する第５の記憶素子と、
を有する情報処理装置。
【請求項１１】
前記桁上げ保存加算器の動作を制御する制御部をさらに有する請求項１０記載の情報処理装置。
【請求項１２】
前記制御部は、
前記第１の記憶素子に前記被乗数Ａをセットし、
前記第２の記憶素子に前記被乗数ｕをセットし、
前記第３の記憶素子に前記乗数Ｂをセットし、
前記第４の記憶素子に前記乗数Ｎをセットし、
前記セレクタに０を供給する請求項１１記載の情報処理装置。

【請求項１３】
予め算出された、前記被乗数Ａ、前記乗数Ｂ、前記乗数Ｎ、及び前記乗算剰余演算結果Ｓの値に対する前記被乗数ｕの値の関係が格納されるｕ生成部をさらに有し、
前記制御部は、
前記Ｓ＝Ｓ＋Ａ×Ｂ＋ｕ×Ｎの演算時に前記ｕ生成部を参照することで前記被乗数ｕの値を決定する請求項１１または１２記載の情報処理装置。
【請求項１４】
前記ビット数ｑは２である請求項１０乃至１３のいずれか１項記載の情報処理装置。
【請求項１５】
前記ビット数ｑは４である請求項１０乃至１３のいずれか１項記載の情報処理装置。
【請求項１６】
前記第１の記憶素子及び前記第２の記憶素子はラッチ回路である請求項１０乃至１５のいずれか１項記載の情報処理装置。
【請求項１７】
前記第３の記憶素子、前記第４の記憶素子及び前記第５の記憶素子はシフトレジスタである請求項１０乃至１６のいずれか１項記載の情報処理装置。

【発明の詳細な説明】
【技術分野】
【０００１】
本発明はべき乗剰余演算を効率よく処理するための乗算剰余演算器及びそれを備えた情報処理装置に関する。
【背景技術】
【０００２】
近年、パーソナルコンピュータやＰＤＡ（Personal Digital(Data) Assistants）あるいは携帯電話機等の各種情報処理装置の処理能力が飛躍的に向上し、さらに各種記録メディアの大容量化や通信インフラストラクチャーの整備が進んだことで、個人情報や企業情報等がネットワークや無線手段を介して送受信される機会が増大している。そのため、それらの情報を秘匿化し第三者への漏洩を防ぐ技術が益々重要になってきている。
【０００３】
送受信データを秘匿化するための一般的な手法としては、データを送受信する端末装置どうしが共通の鍵を用いて該データの暗号化と復号を行う共通鍵暗号方式がよく知られている。さらに、近年ではＢｔｏＢ、ＢｔｏＣ等の電子商取引の拡大に伴ってＰＫＩ（Public Key Infrastructure）技術が注目されている。
【０００４】
ＰＫＩの基本技術である公開鍵暗号方式は、公開鍵を用いて送信データを暗号化し、該公開鍵とペアとなる公開することのない秘密鍵を用いて受信データを復号する方式である。この公開鍵暗号方式は、送信側と受信側で異なる鍵を用い、かつ秘密鍵を通信相手に通知する必要が無いため、上述した共通鍵暗号方式に比べて秘匿化性能が向上する。
【０００５】
公開鍵暗号方式では、現在、ＲＳＡ（Rivest, Shamir Adleman）暗号が主として用いられている（例えば、非特許文献１参照）。ＲＳＡ暗号は、任意の２つの素数を乗算した値Ｎを素因数分解する際の困難性とＮを法とする数の世界の性質とを利用する暗号化方式であり、暗号化及び復号化のためにべき乗剰余演算（$M^d mod N$）を実行する。
【０００６】
べき乗剰余演算は、通常、以下に示す乗算剰余演算の繰り返し処理に置き換えて実行される。
【０００７】
例えば、ｄ＝１９とするとき、$C=M^d mod N$は、
ｄ＝１９＝１＋２×（１＋２×（０＋２×（０＋２×１））））により、

$C = M^{19} \bmod N$
$= M^{1+2\times(1+2\times(0+2\times(0+2\times1)))} \bmod N$
$= (((((M^1)^2 M^0)^2 M^0)^2 M^1)^2 M^1 \bmod N$
$= (((M^2)^2)^2 M)^2 M \bmod N$

となる。このようにｄを分解すれば、Ｍを単純にｄ回掛けるよりも演算回数を低減できるため、演算時間を短縮できる。なお、ｄの分解方法については様々な方法が知られており、上記はその一例を示している。

【０００８】

しかしながら、このような乗算剰余演算も、乗算によって演算桁数が倍になり、さらにその乗算結果をＮで除算するため、ハードウェアまたはソフトウェアのいずれを利用しても効率よく処理するのが非常に困難な演算である。そのため、乗算剰余演算を効率化するための様々な手法が検討され、代表的な例としてモンゴメリ（Montgomery）法と呼ばれるアルゴリズムを応用した演算方法が知られている（例えば、特許文献１参照）。

【０００９】

モンゴメリ法を応用すると、除算を実質的に行わずに乗算と加減算で上記乗算剰余演算が実現可能であり、乗算剰余演算$P(AB)_N = AB \cdot r^{-n} \bmod N = S$は、例えば、以下の（１）～（８）で示す手順で求めることができる。但し、$0 \leqq N < r^n$、Ｎは奇数（Ｎとｒは互いに素である）、$0 \leqq A < N$、$0 \leqq B < N$、$A = A_{n-1}A_{n-2}\cdots A0$（例えば$A = A_3A_2A_1A_0 = 1234$）である。

（１）$v = -N^{-1} \bmod r$
（２）$S = 0$
（３）for i=0 to n−1 {
（４）　$S = S + A_i \cdot B$
（５）　$u = S \cdot v \bmod r$
（６）　$S = S + u \cdot N$
（７）　$S = S / r$
（８）}

乗算剰余演算は、上記アルゴリズムから$S = S + A_i \times B + u \times N$（ｉ＝０～ｎ－１）の繰り返し演算処理に置き換え可能であり、この処理を実現するための回路である乗算剰余演算器は、例えば図６に示すような構成になる。

【００１０】

図６は従来の乗算剰余演算器の構成を示すブロック図である。

【００１１】

図６に示すように、従来の乗算剰余演算器は、被乗数である上記Ａの値を保持する第１のラッチ回路５１と、被乗数である上記ｕの値を保持する第２のラッチ回路５２と、Ａ＋ｕの値を保持する第３のラッチ回路５３と、１ビット毎に供給される乗数Ｂ、Ｎの値に応じて被乗数Ａ、ｕ、Ａ＋ｕ、または０Ｈ（全ビット０）を選択し出力するセレクタ５７と、セレクタ５７から出力される値を用いてＡ×Ｂ＋ｕ×Ｎの演算を行う周知の桁上げ保存加算器（Carry Save Adder:以下、ＣＳＡと称す）５６と、ＣＳＡ５６から出力される乗算剰余演算結果Ｓと外部で保持された算出済みの乗算剰余演算結果Ｓとを加算し、該加算結果を乗算剰余演算結果Ｓとして出力する加算器５９とを有する構成である。なお、Ａ、ｕ、及びＡ＋ｕの各値は、例えば不図示の制御部により第１のラッチ回路５１～第３のラッチ回路５３に供給され、乗数Ｂ、Ｎ、及び０Ｈの各値は、例えば不図示の制御部によりセレクタ５７に供給される。

【００１２】

図６に示す乗算剰余演算器では、乗算剰余演算器の処理ビット長（例えば、512bit）の乗数Ｂ、Ｎがそれぞれ１ビット単位でセレクタ５７に供給される。また、被乗数Ａ、ｕ、Ａ＋ｕは、ＣＳＡ５６の処理ビット長（図６ではｍビット）に対応して、該ビット長単位でラッチ回路に格納され、ＣＳＡ５６に供給される。したがって、例えば乗算剰余演算器の処理ビット長が512bitであり、ＣＳＡ５６の処理ビット長が128bitの場合、図６に示す

構成では、被乗数Ａ、ｕ、Ａ＋ｕの選択処理を５１２回繰り返すことでＡ（128bit）×Ｂ(512bit)＋ｕ(128bit)×Ｎ(512bit)の演算が完了し、さらにそれを４回繰り返すことでＡ（512bit）×Ｂ(512bit)＋ｕ(512bit)×Ｎ(512bit)の演算処理が完了することになる。

【００１３】

セレクタ５７は、１ビットづつ供給される乗数Ｂ、Ｎの値に応じて、第１のラッチ回路５１～第３のラッチ回路５３から供給される被乗数Ａ、ｕ、Ａ＋ｕ、または０Ｈを選択しＣＳＡ５６に供給する。ＣＳＡ５６は、セレクタ５７から順次供給される被乗数Ａ、ｕ、Ａ＋ｕまたは０Ｈをシフト加算することでＡ×Ｂ＋ｕ×Ｎを算出し、その中間演算結果を保持しつつ乗算剰余演算結果Ｓを１ビット単位で出力する。

【非特許文献１】三谷政昭著、「やり直しのための工業数学」、第５版、ＣＱ出版社、2003年2月1日、ｐ．１１５－１２２

【特許文献１】特表２００１－５２７６７３号公報

【発明の開示】

【発明が解決しようとする課題】

【００１４】

現在、公開鍵暗号方式では、上記べき乗剰余演算のＣ、Ｍ、Ｎ、ｄに１０２４ビットの数値を用いたＲＳＡ暗号が広く利用され、さらにビット数が増えることも予想される。そのため、暗号化及び復号化に膨大な量の乗算剰余演算を実行しなければならない。公開鍵暗号方式は、暗号化及び復号化に要する処理時間が共通鍵暗号方式に比べて長いことが問題であり、乗算剰余演算に要する演算時間の短縮が重要な課題となっている。

【００１５】

なお、市場では、携帯電話機、ＰＤＡ、パーソナルコンピュータやサーバ装置等の情報処理装置の普及に伴い、処理性能が高く、かつ低コストな製品が求められている。したがって、このような要求を満たすためには、乗算剰余演算に要する演算時間を短縮すると共に、回路規模の削減を実現できる乗算剰余演算器が必須となる。

【００１６】

本発明は上記したような従来の技術が有する問題点を解決するためになされたものであり、演算時間をより短縮できる乗算剰余演算器及び情報処理装置を提供することを目的とする。

【００１７】

また、本発明のさらなる目的は、回路規模を増大させることなく演算時間を短縮できる乗算剰余演算器及び情報処理装置を提供することにある。

【課題を解決するための手段】

【００１８】

上記目的を達成するため本発明の乗算剰余演算器は、被乗数をＡ、ｕとし、乗数をＢ、Ｎとし、乗算剰余演算結果をＳとしたとき、Ｓ＝Ｓ＋Ａ×Ｂ＋ｕ×Ｎを算出するための乗算剰余演算器であって、

複数のビット数ｑ単位で供給される前記乗数Ｂの値に応じて前記被乗数Ａまたは０の値を切換えて出力し、前記ビット数ｑ単位で供給される前記乗数Ｎの値に応じて前記被乗数ｕまたは０の値を切換えて出力するセレクタと、

前記セレクタから順次出力される値を用いてＡ×Ｂ＋ｕ×Ｎの演算を実行する桁上げ保存加算器と、

前記桁上げ保存加算器から前記ビット数ｑ単位で出力される前記Ａ×Ｂ＋ｕ×Ｎの演算結果と、前記ビット数ｑ単位で供給される過去の該演算結果とを加算し、該加算結果を前記乗算剰余演算結果Ｓとして出力する加算器と、

を有する構成である。

【００１９】

一方、本発明の情報処理装置は、上記乗算剰余演算器と、

前記被乗数Ａを保持し、前記セレクタに供給する第１の記憶素子と、

前記被乗数ｕを保持し、前記セレクタに供給する第２の記憶素子と、

前記乗数Ｂを保持し、前記セレクタに前記ビット数ｑ単位で供給する第３の記憶素子と、

前記乗数Ｎを保持し、前記セレクタに前記ビット数ｑ単位で供給する第４の記憶素子と、

前記加算器から出力される前記乗算剰余演算結果Ｓを保持し、前記ビット数ｑ単位で該乗算剰余演算結果Ｓを前記加算器に供給する第５の記憶素子と、

を有する構成である。

【００２０】

上記のように構成された乗算剰余演算器及び情報処理装置では、乗数を複数のビット数ｑ単位でセレクタに供給し、セレクタにより該乗数の値に応じて被乗数または０の値を切換えてＣＳＡに供給するため、ＣＳＡの処理ビット長をビット数ｑの値に反比例して短縮できる。

【００２１】

また、本発明の乗算剰余演算器及び情報処理装置は、予め算出された、前記被乗数Ａ、前記乗数Ｂ、前記乗数Ｎ、及び前記乗算剰余演算結果Ｓの値に対する前記被乗数ｕの値の関係が格納されるｕ生成部をさらに有し、

制御部により、前記Ｓ＝Ｓ＋Ａ×Ｂ＋ｕ×Ｎの演算時に前記ｕ生成部を参照することで前記被乗数ｕの値を決定する構成である。

【００２２】

なお、前記ビット数ｑは２または４であることが望ましい。

【００２３】

上記のような乗算剰余演算器は、ビット数ｑを２または４とすることで、ｕ生成部の回路規模の増大を抑制できる。

【発明の効果】

【００２４】

本発明の乗算剰余演算器及び情報処理装置は、ＣＳＡの処理ビット長をビット数ｑの値に反比例して短縮できるため、従来の乗算剰余演算器よりも演算時間を短縮できる。

【００２５】

また、ＣＳＡの処理ビット長を短縮することで、ＣＳＡが備えるフリップフロップ数が低減するため、乗算剰余演算器の回路規模が低減する。特に、ビット数ｑを２または４とすれば、ｕ生成部の回路規模が増大することがないため、回路規模を増大させることなく演算時間を短縮できる。

【発明を実施するための最良の形態】

【００２６】

次に本発明について図面を参照して説明する。

【００２７】

図１は本発明の乗算剰余演算器の一構成例を示すブロック図であり、図２は本発明の情報処理装置の一構成例を示すブロック図である。

【００２８】

図１に示すように、本発明の乗算剰余演算器は、被乗数Ａの値を保持する第１のラッチ回路１と、被乗数ｕの値を保持する第２のラッチ回路２と、乗数Ｂの値を保持する第１のシフトレジスタ４と、乗数Ｎの値を保持する第２のシフトレジスタ５と、第１のシフトレジスタ４から複数ビット（図１では2bit）毎に供給される乗数Ｂの値に応じて被乗数Ａまたは０Ｈを選択しＣＳＡ６に供給する第１のセレクタ（Sel1）7_1及び第２のセレクタ（Sel2）7_2と、第２のシフトレジスタ５から複数ビット（図１では2bit）毎に供給される乗数Ｎの値に応じて被乗数ｕまたは０Ｈを選択しＣＳＡ６に供給する第３のセレクタ（Sel3）7_3及び第４のセレクタ（Sel4）7_4と、第１～第４のセレクタ7_1～7_4から供給される値を用いてＡ×Ｂ＋ｕ×Ｎの演算を行う周知のＣＳＡ６と、ＣＳＡ６から複数ビット（図１では2bit）単位で出力される乗算剰余演算結果Ｓを保持し、複数ビット（図１では2bit）単位で出力する第３のシフトレジスタ８と、ＣＳＡ６から出力されるＡ×Ｂ＋ｕ×Ｎの

演算結果と第３のシフトレジスタ８の出力とを加算し、加算結果を乗算剰余演算結果Ｓとして第３のシフトレジスタ８に再び格納する加算器９と、被乗数ｕの値を生成するためのテーブルが格納されるｕ生成部１０と、被乗数Ａ、ｕの値を第１のラッチ回路１及び第２のラッチ回路２に供給し、乗数Ｂ、Ｎの値を第１のシフトレジスタ４及び第２のシフトレジスタ５に供給し、０Ｈを第１～第４のセレクタ7_1～7_4に供給すると共に、ＣＳＡ６、ｕ生成部１０及び第３のシフトレジスタ８の動作を制御する制御部１１とを有する構成である。

【００２９】

本発明の乗算剰余演算器は、制御部１１による被乗数Ａ、ｕのラッチ回路へのセット、及び乗数Ｂ、Ｎのシフトレジスタへのセットを契機に、外部から供給される所定周波数のクロック（CK）にしたがって動作する回路であり、制御部１１は、例えばプログラムにしたがって動作するＣＰＵ、ＤＳＰあるいは論理回路等によって実現される。

【００３０】

このような構成において、本発明の乗算剰余演算器では、被乗数Ａ、ｕが、ＣＳＡ６の処理ビット長に対応して複数に分割され、制御部１１により該分割単位で第１及び第２のラッチ回路１、２に格納される。一方、乗数Ｂ、Ｎは、制御部１１により、例えば乗算剰余演算器の処理ビット長で第１及び第２のシフトレジスタに格納される。なお、乗数Ｂ、Ｎも、所定のビット長毎に複数に分割され、制御部１１により該分割単位で第１及び第２のシフトレジスタに格納されてもよい。

【００３１】

各セレクタには、第１のラッチ回路１及び第２のラッチ回路２から上記分割単位で被乗数Ａ、ｕが供給され、第１のシフトレジスタ４及び第２のシフトレジスタ５から複数ビット単位で乗数Ｂ、Ｎが供給される。図１では、乗数Ｂ、Ｎがそれぞれ2bit単位で第１～第４のセレクタ7_1～7_4に供給される例を示しているが、乗数Ｂ、Ｎは、4bit単位、あるいはそれ以上のビット数単位でセレクタに供給されてもよい。なお、図１では４つのセレクタを用いて被乗数Ａ、ｕまたは０Ｈを切換える構成を示しているが、乗数Ｂ、Ｎの値に対応する被乗数Ａ、ｕまたは０Ｈを選択できれば、セレクタの数はいくつであってもよい。

【００３２】

第１のセレクタ7_1及び第２のセレクタ7_2は、第１のシフトレジスタ４から複数ビットづつ供給される乗数Ｂの値に応じて、被乗数Ａ（１Ａ、２Ａ）または０Ｈを選択しＣＳＡ６に供給する。同様に、第３のセレクタ７３及び第４のセレクタ７４は、第２のシフトレジスタ５から複数ビットづつ供給される乗数Ｎの値に応じて、被乗数ｕ（１ｕ、２ｕ）または０Ｈを選択しＣＳＡ６に供給する。

【００３３】

ここで、１Ａは被乗数Ａの１倍の値であり、２Ａは被乗数Ａの２倍の値である。また、１ｕは被乗数ｕの１倍の値であり、２ｕは被乗数ｕの２倍の値である。２Ａ、２ｕは、例えば被乗数Ａ、ｕの値を１ビットシフトさせれば得られるため容易に生成できる。図１では第１のセレクタ7_1で２Ａを生成し、第３のセレクタ7_3で２ｕを生成する例を示しているが、２Ａ、２ｕはＣＳＡ６内で生成してもよい。

【００３４】

ＣＳＡ６は、各セレクタから順次供給される被乗数または０Ｈをシフト加算することでＡ×Ｂ、及びｕ×Ｎをそれぞれ算出し、それらの加算結果（乗算剰余演算結果）Ｓを複数ビット単位で出力する。ＣＳＡ６から出力された演算結果は、第３のシフトレジスタ８の出力（過去の乗算剰余演算結果Ｓ）と複数ビット単位で加算され、加算結果は第３のシフトレジスタ８に再び格納される。

【００３５】

なお、図１に示した第１のラッチ回路１、第２のラッチ回路２、第１シフトレジスタ４、第２のシフトレジスタ５、第３のシフトレジスタ８、及びｕ生成部１０は、乗算剰余演算器の内部に備えている必要はなく、乗算剰余演算器を利用する情報処理装置に備えていてもよい。同様に、制御部１１も乗算剰余演算器の内部に備えている必要はなく、乗算剰

余演算器を利用する情報処理装置が備える処理装置（ＣＰＵ）によって実現してもよい。すなわち、乗算剰余演算器は、図１の点線内の構成要素のみを備えていればよい。

【００３６】

また、被乗数Ａ、ｕは、ラッチ回路に格納する必要はなく、例えばシフトレジスタやＲＡＭ等のようにデータを一時的に保持できる記憶素子であればどのようなものを用いてもよい。同様に、乗数Ｂ、Ｎ、及び演算結果Ｓは、シフトレジスタに格納する必要はなく、例えばＲＡＭのように格納されたデータを複数ビット単位で出力できる記憶素子であればどのようなものを用いてもよい。

【００３７】

図２に示すように、本発明の情報処理装置は、例えばパーソナルコンピュータやサーバ装置等のコンピュータシステムであり、プログラムにしたがって所定の処理を実行する処理装置２０と、処理装置２０に対してコマンドや情報等を入力するための入力装置３０と、処理装置２０の処理結果をモニタするための出力装置４０とを有する構成である。

【００３８】

処理装置２０は、ＣＰＵ２１と、ＣＰＵ２１の処理に必要な情報を一時的に記憶する主記憶装置２２と、ＣＰＵ２１に上記制御部１１の処理を実行させるプログラムが記録された記録媒体２３と、処理に必要なデータ等を蓄積するデータ蓄積装置２４と、主記憶装置２２、記録媒体２３、及びデータ蓄積装置２４とのデータ転送を制御するメモリ制御インタフェース部２５と、入力装置３０及び出力装置４０とのインタフェース装置であるＩ／Ｏインタフェース部２６と、図１に示した乗算剰余演算器２７と、ネットワーク等との通信を制御するインタフェースである通信制御装置２８とを備え、それらがバス２９等を介して接続された構成である。なお、処理装置２０には、乗算剰余演算器２７の構成に応じて、被乗数Ａ、ｕを保持するラッチ回路、及び乗数Ｂ、Ｎ、及び演算結果Ｓを保持するシフトレジスタ等を備えていてもよい。

【００３９】

処理装置２０は、記録媒体２３に記録されたプログラムにしたがってＣＰＵ２１により上記制御部１１の処理を実行し、乗算剰余演算器２７を用いて$S = S + A_i \times B + u \times N$の演算を実行する。なお、記録媒体２３は、磁気ディスク、半導体メモリ、光ディスクあるいはその他の記録媒体であってもよい。

【００４０】

次に、本発明の乗算剰余演算器の動作について図面を用いて具体的に説明する。

【００４１】

以下では、Ａ、ｕ、Ｂ、Ｎがそれぞれ512bitであり、処理ビット長が64bitのＣＳＡ６を用い、第１及び第２のシフトレジスタ４、５がそれぞれ2bit単位で各セレクタに乗数Ｂ、Ｎを供給し、第３のシフトレジスタ８が2bit単位で乗算剰余演算結果Ｓを入出力する場合を例にして説明する。また、第１及び第２のシフトレジスタ４、５には乗数Ｂ、Ｎがそれぞれ512bit単位で格納され、第１及び第２のラッチ回路１、２には被乗数Ａ、ｕがＣＳＡ６の処理ビット長に合わせて64bit単位で格納されるものとする。

【００４２】

処理ビット長が64bitのＣＳＡ６を用い、乗数Ｂ、Ｎを2bit単位で出力する場合、Ａ、ｕ、Ｂ、Ｎがそれぞれ512bitの乗算剰余演算（$512\text{bit} \times 512\text{bit} \times 2^{-512} \bmod 512\text{bit}$）は、$64\text{bit} \times 512\text{bit} \times 2^{-64} \bmod 512\text{bit}$（$A \times B \times 2^{-64} \bmod N$）の演算を繰り返し実行すればよい。

【００４３】

本発明の乗算剰余演算器では、モンゴメリ法による乗算剰余演算の特徴である、下位ビットが０になることを利用して（ここでは、下位64bitが０Ｈ）、上記Ｓ、Ａ、Ｂ、Ｎの値に対応するｕを予め算出し、ｕ生成部１０にテーブル形式で格納しておく。

【００４４】

例えば、乗数を2bit単位で出力する場合、ｕの値は以下のように求める（但し、Ｎは奇数）。

【００４５】

N[1:0]=01,(S+AiB)[1:0]=00のとき、
S=S+AiB+uN=00となるuは、u[1:0]=00
N[1:0]=01,(S+AiB)[1:0]=01のとき、
S=S+AiB+uN=00となるuは、u[1:0]=11
N[1:0]=01,(S+AiB)[1:0]=10のとき、
S=S+AiB+uN=00となるuは、u[1:0]=10
N[1:0]=01,(S+AiB)[1:0]=11のとき、
S=S+AiB+uN=00となるuは、u[1:0]=01
N[1:0]=11,(S+AiB)[1:0]=00のとき、
S=S+AiB+uN=00となるuは、u[1:0]=00
N[1:0]=11,(S+AiB)[1:0]=01のとき、
S=S+AiB+uN=00となるuは、u[1:0]=01
N[1:0]=11,(S+AiB)[1:0]=10のとき、
S=S+AiB+uN=00となるuは、u[1:0]=10
N[1:0]=11,(S+AiB)[1:0]=11のとき、
S=S+AiB+uN=00となるuは、u[1:0]=11

以上をまとめると、表１のようになる。

【００４６】

【表１】

N[1]	S+AiB[1:0]	u
0	00	00
0	01	11
0	10	10
0	11	01
1	00	00
1	01	01
1	10	10
1	11	11

【００４７】

ここで、Ａ、Ｂ、Ｎはいずれも既知の値（第１のラッチ回路１、第１のシフトレジスタ４及び第２のシフトレジスタ５に格納する値）であり、Ｓは０Ｈ（演算開始時）または直前の64bit×512bit×2^{-64} mod 512bitの演算結果を用いるため既知である。なお、Ｎは奇数であるため、N[1:0]=01または11で固定である。したがって、Ａ、Ｂ、及びＳの各値を基に算出した被乗数ｕの値をテーブル形式でｕ生成部１０に格納しておき、制御部１１は該テーブルを参照して被乗数ｕの値を決定する。なお、参考として表２に乗数Ｂ、Ｎを4bit単位で出力する場合に用いるｕを生成するためのテーブルを示す。

【００４８】

【表２】

N[3:1], S+AB[3:0]	u	N[3:1], S+AB[3:0]	u	N[3:1], S+AB[3:0]	u	N[3:1], S+AB[3:0]	u
0	0	16	0	32	0	48	0
1	15	17	5	33	3	49	9
2	14	18	10	34	6	50	2
3	13	19	15	35	9	51	11
4	12	20	4	36	12	52	4
5	11	21	9	37	15	53	13
6	10	22	14	38	2	54	6
7	9	23	3	39	5	55	15
8	8	24	8	40	8	56	8
9	7	25	13	41	11	57	1
10	6	26	2	42	14	58	10
11	5	27	7	43	1	59	3
12	4	28	12	44	4	60	12
13	3	29	1	45	7	61	5
14	2	30	6	46	10	62	14
15	1	31	11	47	13	63	7

N[3:1], S+AB[3:0]	u	N[3:1], S+AB[3:0]	u	N[3:1], S+AB[3:0]	u	N[3:1], S+AB[3:0]	u
64	0	80	0	96	0	112	0
65	7	81	13	97	11	113	1
66	14	82	10	98	6	114	2
67	5	83	7	99	1	115	3
68	12	84	4	100	12	116	4
69	3	85	1	101	7	117	5
70	10	86	14	102	2	118	6
71	1	87	11	103	13	119	7
72	8	88	8	104	8	120	8
73	15	89	5	105	3	121	9
74	6	90	2	106	14	122	10
75	13	91	15	107	9	123	11
76	4	92	12	108	4	124	12
77	11	93	9	109	15	125	13
78	2	94	6	110	10	126	14
79	9	95	3	111	5	127	15

【００４９】

本発明の乗算剰余演算器では、まず、制御部１１により、第１のラッチ回路１に被乗数Ａ（512bit）の最下位64bitのデータをセットし、第１のシフトレジスタ４に乗数Ｂ（512bit）のデータをセットし、第２のシフトレジスタ５に乗数Ｎ（512bit）のデータをセットする。

【００５０】

続いて、制御部１１は、64bitの被乗数Ａ、64bitの乗数Ｂ、64bitの乗数ＮからｕをＵ生成部１０に格納されたテーブルを参照してｕ（64bit分）の値を求め、第２のラッチ回路２に格納する。

【００５１】

制御部１１による第１のラッチ回路１、第２のラッチ回路２、第１のシフトレジスタ４及び第２のシフトレジスタ５に対する被乗数または乗数のセットが完了すると、乗算剰余演算器はＳ＝Ｓ＋Ａ×Ｂ＋ｕ×Ｎの演算を開始する。

【００５２】

乗算剰余演算器は、まず、第１のセレクタ７_{1}及び第２のセレクタ７_{2}にて、第１のシフトレジスタ４から出力される2bitの乗数Ｂの値から被乗数Ａ（64bit）または０Ｈを選択

しＣＳＡ６へ供給する。ここでは、第１のセレクタ７_1により０Ｈ／２Ａを切換え、第２のセレクタ７_2により０Ｈ／１Ａを切換える。

【００５３】

同様に、乗算剰余演算器は、第３のセレクタ７_3及び第４のセレクタ７_4にて、第２のシフトレジスタ５から出力される2bitの乗数Ｎの値から被乗数ｕ（64bit）または０Ｈを選択しＣＳＡ６へ供給する。ここでは、第３のセレクタ７_3により０Ｈ／２ｕを切換え、第４のセレクタ７_4により０Ｈ／１ｕを切換える。

【００５４】

ＣＳＡ６は、各セレクタから順次供給される被乗数または０Ｈの値を、桁合わせを実行しつつ加算することでＡ×Ｂ、及びｕ×Ｎを算出し、それらの加算結果（乗算剰余演算結果）Ｓを2bit単位で出力する。ＣＳＡ６から出力された演算結果は、第３のシフトレジスタ８の出力と2bit単位で加算器９にて加算され、加算後の値が第３のシフトレジスタ８に再び格納される。

【００５５】

図３は本発明の乗算剰余演算器が実行する演算処理のうち、Ａ×Ｂの演算処理の手順を示している。ｕ×Ｎの演算処理も図３に示すＡ×Ｂの演算処理と同様である。図３に示すＡ×Ｂの演算結果の下位２ビット（φ１）と、同様にして求めたｕ×Ｎの演算結果の下位２ビットの加算結果が、ＣＳＡ６の出力（2bit）となる。

【００５６】

この処理を第１のシフトレジスタ４及び第２のシフトレジスタ５内の全てのビットデータに対して繰り返し実行することで、$64bit \times 512bit \times 2^{-64} \bmod 512bit$の演算が終了する。但し、この段階ではＣＳＡ６の内部に部分積の演算結果の上位64bitが残っているため、このデータを制御部１１の指示により第３のシフトレジスタ８に格納する。その結果、第３のシフトレジスタ８に$64bit \times 512bit \times 2^{-64} \bmod 512bit$の演算結果Ｓが格納される。

【００５７】

乗算剰余演算器は、$64bit \times 512bit \times 2^{-64} \bmod 512bit$の演算が完了すると、制御部１１により第１のラッチ回路１に被乗数Ａ（512bit）の次の下位64bitのデータ（最下位から65bit目～128bit目のデータ）をセットし、上記と同様にｕ生成部１０のテーブルを参照して被乗数ｕの値を求め、求めた値を第２のラッチ回路２に格納した後、再び$64bit \times 512bit \times 2^{-64} \bmod 512bit$の演算を開始する。

【００５８】

以降、第１のラッチ回路１に格納される被乗数Ａ（512bit）の全てのビットデータに対して同様の処理を繰り返し実行する。すなわち、上記$64bit \times 512bit \times 2^{-64} \bmod 512bit$の演算を８回繰り返す。その結果、本発明の乗算剰余演算器による$512bit \times 512bit \times 2^{-512} \bmod 512bit$の演算が終了する。

【００５９】

次に、本発明の乗算剰余演算器の効果について図面を用いて説明する。

【００６０】

図４は乗数を1bit単位で出力する従来の乗算剰余演算器の回路規模及び乗数を2bitまたは4bit単位で出力する本発明の乗算剰余演算器の回路規模を示すグラフである。また、図５は乗数を1bit単位で出力する従来の乗算剰余演算器の処理クロック数及び乗数を2bitまたは4bit単位で出力する本発明の乗算剰余演算器の処理クロック数を示すグラフである。なお、図５は、乗算剰余演算器の処理ビット長を512bitとしたときの乗算剰余演算に必要な処理クロック数を示している。

【００６１】

図４及び図５に示す1bit構成とは乗数を1bit単位で出力する従来の乗算剰余演算器の構成を示し、2bit構成とは乗数を2bit単位で出力する本発明の乗算剰余演算器の構成を示し、4bit構成とは乗数を4bit単位で出力する本発明の乗算剰余演算器の構成を示している。以降の説明でも1bit構成、2bit構成、あるいは4bit構成と称した場合は同様の構成を示し

ているものとする。

【００６２】

また、図４及び図５に示すグラフの横軸は、乗算剰余演算器の処理ビット長（128bit、256bit、512bit）を示し、以下では、乗算剰余演算器の処理ビット長が同じである場合、乗数の出力ビット単位に反比例してＣＳＡ６の処理ビット長が短縮されるものとする。それらの関係を表３に示す。ここで、表３の各エントリは（ＣＳＡの処理ビット長）＊（出力ビット数）を示している。

【００６３】

【表３】

乗算剰余演算器	4bit 構成	2bit 構成	1bit 構成
128bit	32bit*4bit	64bit*2bit	128bit*1bit
256bit	64bit*4bit	128bit*2bit	256bit*1bit
512bit	128bit*4bit	256bit*2bit	512bit*1bit

【００６４】

図４から分かるように、乗算剰余演算器の処理ビット長が同じである場合、乗数を複数ビット単位で出力する本発明の乗算剰余演算器は、乗数を1bit単位で出力する従来の乗算剰余演算器に比べて回路規模が低減する。

【００６５】

図４を基に1bit構成の従来の乗算剰余演算器と4bit構成の本発明の乗算剰余演算器を比較した場合、本発明の乗算剰余演算器は、従来の約半分の回路規模となる。これは2bit構成とすることでＣＳＡ６の処理ビット長を従来の半分にできるためであり、4bit構成とすることでＣＳＡ６の演算ビット長を従来の１／４にできるためである。

【００６６】

例えば、乗算剰余演算器の処理ビット長を128bitとした場合、従来の乗算剰余演算器では、ＣＳＡでSUM（加算結果）の値とCARRY（桁上げ）の値をそれぞれ128個ずつ保持する必要があるため、２５６個のフリップフロップ（Data-F/F）が必要になる。

【００６７】

それに対して、2bit構成の本発明の乗算剰余演算器が備えるＣＳＡ６では、処理ビット長が従来の半分の64bitで済むため、SUMの値とCARRYの値を保持するフリップフロップも１２８個で済む。すなわち、本発明のように乗数を複数ビット単位で出力すると、ＣＳＡ６が備えるフリップフロップの数を大きく削減できるため、回路規模を低減できる。

【００６８】

一方、図５から分かるように、乗算剰余演算器の処理ビット長が同じである場合、乗数を複数ビット単位で出力する本発明の乗算剰余演算器は、乗数を1bit単位で出力する従来の乗算剰余演算器に比べて処理クロック数が少なくなる。これはＣＳＡ６内に残る演算結果を出力する処理時間の差から生じる結果である。

【００６９】

本発明の乗算剰余演算器では、上述したようにＣＳＡ６の処理ビット長を従来の半分あるいは１／４にできるが、被乗数を分割して処理するため、乗算剰余演算を複数回繰り返すことになる。そのため、本発明の乗算剰余演算器では、従来の乗算剰余演算器よりも繰り返し演算の回数が増え、ＣＳＡ６内に残る部分積の演算結果を出力する回数も増えてしまう。

【００７０】

しかしながら、本発明の乗算剰余演算器では、ＣＳＡ６の処理ビット長を短縮できることから、ＣＳＡ６内に残る演算結果を出力する処理時間も2bit構成の場合で従来の半分となり、4bit構成の場合で従来の１／４となる。そのため、僅かではあるが、１つのＡ、ｕ、Ｂ、Ｎに対する乗算剰余演算の処理時間は従来よりも低減する。

【００７１】
本発明の乗算剰余演算器は、処理時間の大幅な低減は実現できないが、多数の数字の配列に対して大きな値のべき乗剰余演算を行うＲＳＡによる暗号化及び復号に本発明の乗算剰余演算器を用いる場合は、この僅かな処理時間の向上が非常に有益となる。
【００７２】
ところで、被乗数ｕは、乗数Ｂ、Ｎの出力ビット数をｑとすると、上記モンゴメリ法を応用したアルゴリズムの（１）、（５）から以下の式で算出できる。
【００７３】
$v = -N^{-1} \bmod 2^q$
$u = Sv \bmod 2^q$
ここで、ｖは演算開始時に一度だけ計算する値である。なお、ｒに代えて2^qとしているのはｒを２進数で表したためである。
【００７４】
ｑ＝１となる従来の乗算剰余演算器では、Ｎが奇数であることからｖ＝１となるため、ｕ＝Ｓｍｏｄ２＝Ｓ［０］となり、被乗数ｕはＳの下位ビットに等しくなる。したがって、被乗数ｕを実施的に計算する必要はない。
【００７５】
しかしながら、ｑ＞１となる本発明の乗算剰余演算器では、ｕ＝Ｓ［０］が成立しないため、上記２つの演算が必要になる。但し、ｑの値が小さい場合（例えば、ｑ＝２、４）は、ｖ、ｕも2bitまたは4bitであり、その演算に必要なＮ、Ｓも2bitまたは4bitである。そのため、本発明ではＡ、Ｂ、Ｓ、Ｎの値から予めｕの値を算出してテーブルを作成しておき、該テーブルを参照することで第２のラッチ回路２に格納するｕを決定している。このｑの値を大きくすれば、ＣＳＡ６の処理ビット長をさらに短縮できるため、乗算剰余演算の処理時間をさらに短縮することができる。
【００７６】
しかしながら、ｑ＞４の場合、すなわち乗数Ｂ、Ｎを８ビット以上で出力する構成では、被乗数ｕをテーブル内から選択するために必要な、例えばデコーダ等の回路規模が増大するため、記憶素子を含むｕ生成部１０の回路規模が増大し、上述したＣＳＡ６の処理ビット長を短縮することによる乗算剰余演算器の回路規模の低減効果を相殺してしまう。
【００７７】
表４にｑの値に対するｕ生成部１０のレイアウト面積（単位：mm^2）を示し、表５にｑの値に対するＣＳＡとｕ生成部とを含む総レイアウト面積（単位：mm^2）を示す。
【００７８】
【表４】

q=1	q=2	q=4	q=8
0	0.003	0.014	0.937

【００７９】
【表５】

ＣＳＡ+u 生成部	q=1	q=2	q=4	q=8
32bit	0.103	0.169	0.308	1.371
64bit	0.292	0.423	0.592	1.903
128bit	0.580	0.842	1.171	2.988
256bit	1.153	1.691	2.310	5.135

【００８０】
表４及び表５から分かるように、例えばＣＳＡの処理ビット長を256bitとしたとき、ｑ

＝１のときの総レイアウト面積に対して、ＣＳＡの処理ビット長を128bitにできるｑ＝２の場合及びＣＳＡの処理ビット長を64bitにできるｑ＝４の場合の総レイアウト面積は低減する。しかしながら、ｑ＝８にすると総レイアウト面積が増大してしまう。

【００８１】

したがって、本発明の乗算剰余演算器では、ｑの値が２または４であることが回路規模の増大を抑制しつつ演算時間を短縮できるために望ましい。但し、回路規模よりも演算時間の向上を優先する場合は、ｑの値を８以上に設定してもよい。その場合、ｑの値はｕ生成部１０のレイアウト面積の増大を考慮しつつ最適な値を選択すればよい。

【図面の簡単な説明】

【００８２】

【図１】本発明の乗算剰余演算器の一構成例を示すブロック図である。

【図２】本発明の情報処理装置の一構成例を示すブロック図である。

【図３】本発明の乗算剰余演算器が実行する演算処理のうち、Ａ×Ｂの演算処理の手順を示す模式図である。

【図４】本発明の乗算剰余演算器の回路規模を示すグラフである。

【図５】本発明の乗算剰余演算器の処理クロック数を示すグラフである。

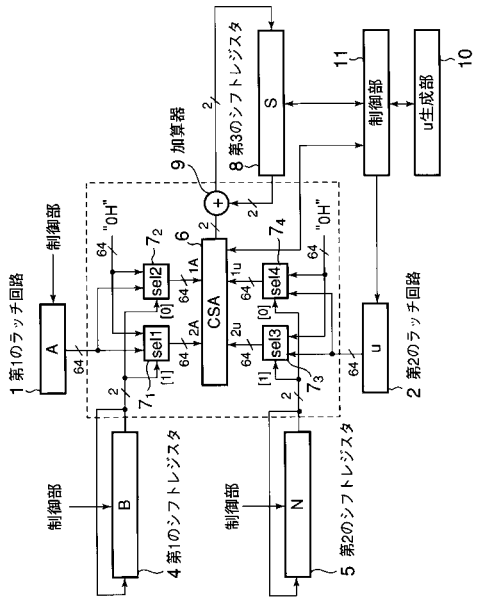
【図６】従来の乗算剰余演算器の構成を示すブロック図である。

【符号の説明】

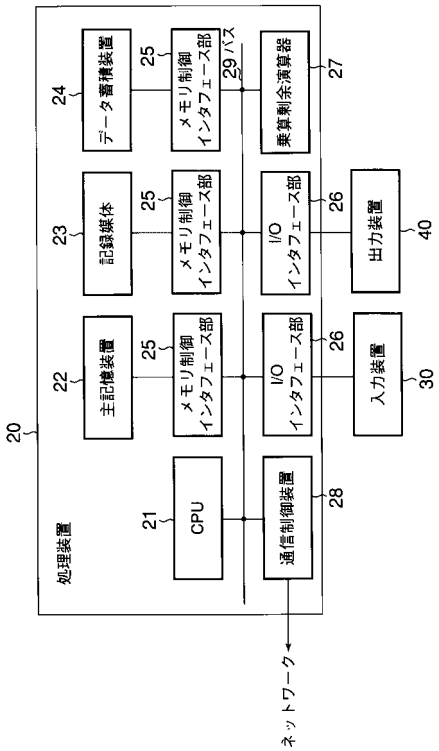
【００８３】

１　第１のラッチ回路
２　第２のラッチ回路
４　第１のシフトレジスタ
５　第２のシフトレジスタ
６　ＣＳＡ
7_1　第１のセレクタ
7_2　第２のセレクタ
7_3　第３のセレクタ
7_4　第４のセレクタ
８　第３のシフトレジスタ
９　加算器
１０　ｕ生成部
１１　制御部
２０　処理装置
２１　ＣＰＵ
２２　主記憶装置
２３　記録媒体
２４　データ蓄積装置
２５　メモリ制御インタフェース部
２６　Ｉ／Ｏインタフェース部
２７　乗算剰余演算器
２８　通信制御装置
２９　バス
３０　入力装置
４０　出力装置

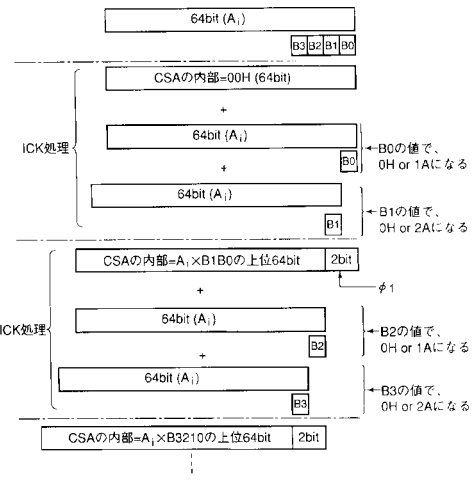
【図１】



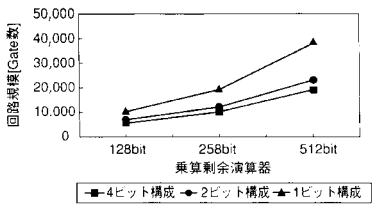
【図２】



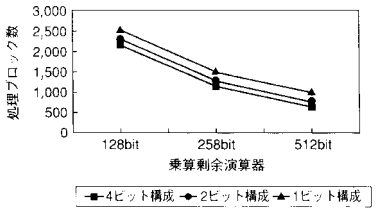
【図３】



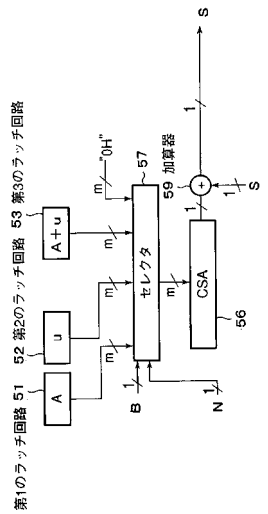
【図４】



【図５】



【図６】



フロントページの続き

(72)発明者 東 邦彦
神奈川県川崎市中原区小杉町１丁目４０３番５３ ＮＥＣマイクロシステム株式会社内
(72)発明者 久門 亨
神奈川県川崎市中原区小杉町１丁目４０３番５３ ＮＥＣマイクロシステム株式会社内
(72)発明者 後藤 敏
東京都新宿区戸塚町一丁目１０４番地 学校法人 早稲田大学内
(72)発明者 池永 剛
東京都新宿区戸塚町一丁目１０４番地 学校法人 早稲田大学内

審査官 速水 雄太

(56)参考文献 特開２００３－２１６０３４（ＪＰ，Ａ）
特表２００１－５２７６７３（ＪＰ，Ａ）
特開２０００－３２２２３９（ＪＰ，Ａ）
鈴木大輔 他，FPGA上で効果的なモンゴメリ乗算回路の一構成法について，２００３年暗号と情報セキュリティシンポジウム（ＳＣＩＳ２００３），２００３年
Alan Daly, William Marnane, Efficient architectures for implementing montgomery modular multiplication and RSA modular exponentiation on reconfigurable logic, Proceedings of the 2002 ACM/SIGDA tenth international symposium on Field-programmable gate arrays, ２００２年, pp. 40-49

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０９Ｃ １／００
Ｇ０６Ｆ ７／７２