システムＬＳＩ設計技術　106－12
(2002.　5.　24)

冗長2進数表現を用いた鍵長可変RSA公開鍵暗号アクセラレータ

谷光　耕平†　　中村　次男††　　笠原　　宏†　　冬爪　成人†

† 東京電機大学情報環境学部　〒270–1200 千葉県印西市武西学園台 2–1200
†† 国際短期大学　〒165–0022 東京都中野区江古田 4–15–1
E-mail: †{tanimitsu,nakamura,kasahara,fuyu}@itl.sie.dendai.ac.jp

あらまし　公開鍵暗号演算用の専用ハードウェアなどに用いられる乗除算器は，高精度な可変精度乗除算器によって構成される事が望ましい．しかし一般に，遅延は演算桁数に比例して増加するので，遅延を演算桁数にかかわらず一定にすることのできる冗長2進数表現を用いた乗除算器によるRSA公開鍵暗号のモジュール化を提案する．本モジュールはチップスライス化を可能にしたアーキテクチャになっているので任意精度に容易に拡張できる点に特徴があり，鍵の長さを更に拡張する場合にも容易に対処可能である．

キーワード　RSA公開鍵暗号，ハードウェアアクセラレータ，準並列形演算器，冗長2進数表現

RSA Public Key Cryptosystem Accelerator Using Redundant Binary Representation

Kouhei TANIMITSU†, Tsugio NAKAMURA††, Hiroshi KASAHARA†, and Narito FUYUTSUME†

† School of Information Environment, Tokyo Denki University　Muzai Gakuendai 2–1200, Inzai-shi, Chiba, 270–1382 Japan
†† Kokusai Junior College　Egota 4–15–1, Nakano-ku, Tokyo, 165–0022 Japan
E-mail: †{tanimitsu,nakamura,kasahara,fuyu}@itl.sie.dendai.ac.jp

Abstract　The multiplier and divider used for specific hardware of public key cryptosystem arithmetic are constructed from many multiplier/divider for number of arbitrary word length. However, with the increase of accuracy, propagation delay problem is usually unavoidable. We propose a modularizing RSA public key cryptosystem with the multiplier/divider using redundant binary representation to cope with the problem.

Key words　RSA Public Key Cryptosystem, Hardware Accelerator, Quasi-Parallel Arithmetic Unit, Redundant Binary Representation

1. まえがき

公開鍵暗号システムはデータの暗号化・復号の他に本人認証などにも使う事ができるため，1,024ビットの鍵長とするRSA公開鍵暗号システムが一般的に用いられている．そのため，高速に処理できる他の暗号方式と組み合わせて使うのが一般的である．実際のデータ暗号化には高速で処理のできる秘密鍵暗号を使い，秘密鍵暗号の鍵を公開鍵暗号で暗号化し，相手に送っている．よりデータの安全性を高めるには全てのデータをRSA公開鍵暗号で暗号化すればよいが，暗号化・復号にかかる時間が秘密鍵暗号と比べ長くなるため，一般的には用いられていない．しかしRSA公開鍵暗号方式はセキュリティが高い反面，演算処理に必要な時間が長いという欠点がある．高速なRSA公開鍵暗号の処理にはハードウェアアクセラレータを用いる方法があるが，従来の方法では，将来セキュリティを高めるために鍵長が増大しハードウェアアクセラレータの処理できる鍵長を超えると処理ができなくなってしまう[1][2][3]．そこで筆者らは，先にビットスライス化を考慮したRSA公開鍵暗号処理専用のハードウェアを開発した[7]．

本研究ではさらなる高速性を追求し，アクセラレータの演算回路に拡張性を取り入れた準並列形演算器[8][9]を用いて，複数のアクセラレータをカスケード接続する事により処理できる鍵長を任意に拡張可能な構造とした．また複数モジュールをカスケード接続すると通常の2進表現による順序回路形演算器では桁上げ・桁借り伝播遅延時間が増大し回路の動作速度が低下する．この問題に対して通常の2進数表現ではなく，冗長2進

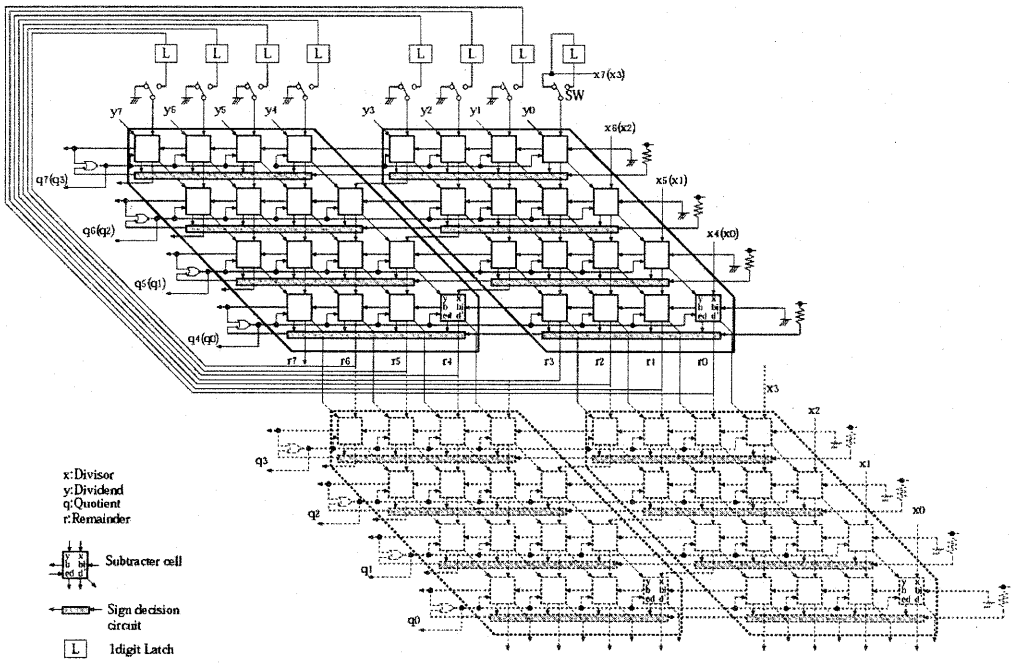

図 1　拡張時の準並列形除算器

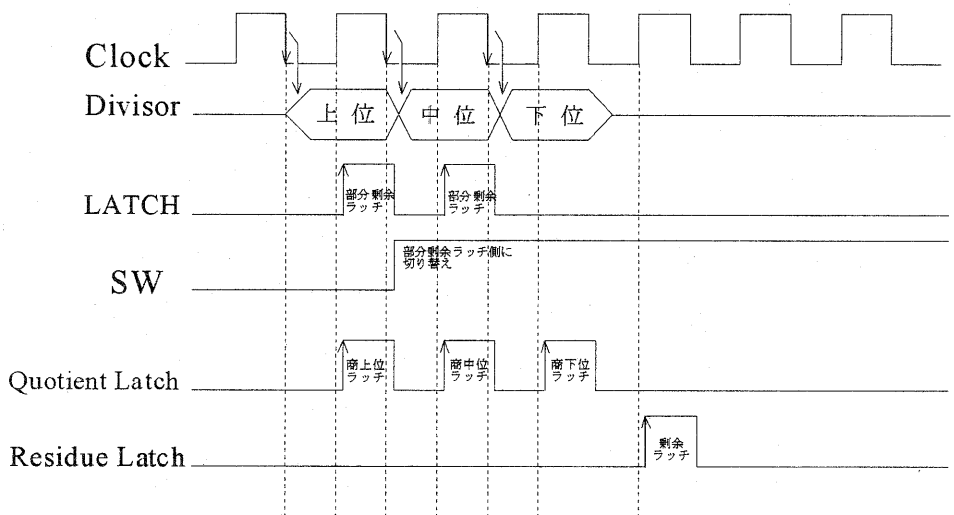

図 2　除算器の 3 モジュールカスケード時のタイミングチャート

数表現を用いることにより桁上げ・桁借り伝播遅延時間を一定に抑えた．そのため，アクセラレータを複数個カスケード接続しても演算回路の遅延時間は一定となり，動作速度の低下を伴わない構造を実現した．

本論文では先ず，準並列形べき乗剰余演算回路と冗長 2 進数表現について説明し，次にそれらを用いて構成した RSA 公開鍵暗号アクセラレータについて述べ，最後に計算機上で RSA 公開鍵暗号処理をした場合との比較検討を行う．

2.　準並列形べき乗剰余演算回路

準並列形演算器 [8] [9] は 1 モジュール精度の n 倍の演算精度に対して n 個のモジュールをカスケード接続し，n-1 回の繰り返し演算で解を得る方式である．繰り返し演算には，モジュール間での演算情報の伝播の他に，n-1 回の部分積・部分剰余のフィードバックや，積，商，剰余の出力法を新たに開発する必要がある．そのためフィードバックされた部分積や部分剰余の

一時的な保持用として演算器にラッチ回路を付加している．

RSA 公開鍵暗号ではべき乗剰余演算が用いられるため積と剰余が必要となる．準並列形乗算器では n 個のカスケード接続の場合，n-1 回だけ部分積と分割された積が発生する．そのため，フィードバックするごとに，分割して出力される積を保持しておかなければならない．準並列形除算器で剰余を求める場合，n 個のカスケード接続では n-1 回のフィードバックによる演算終了後，全剰余が求められる．また準並列形演算器ではカスケード接続の段数を増やした場合でも，フィードバックの回数を制御するだけで演算を実行する事ができる．これにより演算精度の容易な拡張を可能としている．

2.1 準並列形演算回路の動作・拡張性

準並列形演算器の動作を除算器を例にあげて説明する．説明を簡単にする為に 1 モジュールの除算精度が 4 ビットの準並列形除算器を 2 つ接続し，8 ビット除算器に拡張した図を図 1 に示す．除数の上位 4 ビットは上位モジュールの y7 ～ y4 に入力し，下位 4 ビットは下位モジュールの y3 ～ y0 に入力する．被除数は上位 4 ビットと下位 4 ビットを分け，初め x7 ～ x4 に上位 4 ビットを入力して，r0 ～ r6 に部分剰余を得る．この部分剰余 r0 ～ r6 をフィードバックさせ，ラッチ L に保持する．次に SW をラッチ L 側に切り替え x7 ～ x4 に被除数の下位 4 ビットを入力して，r7 ～ r0 に剰余が出力される．

準並列形除算器を 3 モジュールカスケード接続したタイミングチャートを図 2 に示す．

3. 冗長 2 進数表現

冗長 2 進数表現とは，符号付ディジット (SD:Signed Digit) 表現の一種であり，1 桁を-1,0,1 という 3 通りで表す．[4] [5] [6] そのため 1 桁を表すには最低 2 ビットを必要とする．これにより回路規模の増加が懸念されるが，近年の高集積化技術の向上により実現不可能な回路規模では無くなった．冗長 2 進数表現を並列加減算器に用いる事により，演算桁数にかかわらず桁上げ・桁借りを高々一桁に抑える事ができる．

3.1 2 進数表現と冗長 2 進数表現の関係

冗長 2 進数表現は，通常の 2 進数表現から特別な変換回路を必要とせず扱う事ができる．これは，通常の 2 進数表現も冗長 2 進数表現の一部だからである．冗長 2 進数表現から通常の 2 進数表現への変換は，正の桁から負の桁を減算する事により得られる．以下に冗長 2 進数 R の例を示す．

$$\begin{aligned} R &= 10\bar{1}100\bar{1}1_{SD2} \\ &= \langle 0100100100001001 \rangle_2 \\ &= \langle 4909 \rangle_{16} \end{aligned} \tag{1}$$

この冗長 2 進数 R を正のビット列と負のビット列に分割し，正のビット列から負のビット列を減算する

$$\begin{aligned} R &= \langle 10010001 \rangle_2 - \langle 100010 \rangle_2 \\ &= \langle 1101111 \rangle_2 \\ &= \langle 6F \rangle_{16} \end{aligned} \tag{2}$$

このような操作を行う事により，冗長 2 進数表現から通常の 2 進数表現に変換する事ができる．

3.2 冗長 2 進数の加減算

表 1 冗長 2 進数の加算規則

被加数 x_i	加数 y_i	1 桁下位 x_{i-1}, y_{i-1}	桁上げ c_i	中間和 s_i
1	1	*	1	0
0	1	少なくても一方負	0	1
1	0	両方とも非負	1	$\bar{1}$
0	0	*	0	0
1	$\bar{1}$			
$\bar{1}$	1			
$\bar{1}$	0	少なくても一方負	$\bar{1}$	1
0	$\bar{1}$	両方とも非負	0	$\bar{1}$
$\bar{1}$	$\bar{1}$	*	$\bar{1}$	0

*: Don't Care

表 2 冗長 2 進数の減算規則

被減数 x_i	減数 y_i	1 桁下位 x_{i-1}, y_{i-1}	桁借り b_i	中間差 d_i
1	$\bar{1}$	*	1	0
0	$\bar{1}$	$x_{i-1} = 1$ *or* $y_{i-1} = \bar{1}$	1	$\bar{1}$
1	0	それ以外	0	1
0	0	*	0	0
1	1			
$\bar{1}$	$\bar{1}$			
$\bar{1}$	0	$x_{i-1} = 1$ *or* $y_{i-1} = \bar{1}$	0	$\bar{1}$
0	1	それ以外	$\bar{1}$	1
$\bar{1}$	1	*	$\bar{1}$	0

*: Don't Care

冗長 2 進数の加算は次の方法で行う．被加数を X_i，加数を Y_i，中間桁上げを C_i，中間和を S_i とすると，

- X_i+Y_i=2C_i+S_i となるように C_i と S_i を決める．
- 下位桁からの桁上げ C_{i-1} と S_i との和 Z_i を各桁で計算する．

という二つのステップで行う．

S_i とひとつ下位桁の C_{i-1} は同時に $\bar{1}$ または 1 にならないように，一つ下位桁の X_{i-1} と Y_{i-1} も調べて決め，

$$Z_i = S_i + C_{i-1} \tag{3}$$

を各桁で求める．Z_i は S_i と C_{i-1} だけで決まるため，桁上げは一桁しか伝播しない．冗長 2 進数の加算規則を表 1 に示す．

また，冗長 2 進数の減算は次の方法で行う．被減数を X_i，減数を Y_i，中間桁借りを B_i，中間差を D_i とすると，

- X_i-Y_i=2B_i+D_i となるように B_i と D_i を決める．
- 上位桁からの桁借り B_{i-1} と D_i との和 Z_i を各桁で計算する．

という加算と同じ二つのステップで行う．

D_i とひとつ下位桁の B_{i-1} は同時に $\bar{1}$ または 1 にならない

ように，一つ下位桁の X_{i-1} と Y_{i-1} も調べて決め，

$$Z_i = D_i + B_{i-1} \tag{4}$$

を各桁で求める．Z_i は加算と同じく，D_i と B_{i-1} だけで決まり，桁借りは一桁しか伝播しない．冗長 2 進数の減算規則を表 2 に示す．このように冗長 2 進数表現では，加減算の桁数に関係なく一定の時間で演算をする事ができる．

4. RSA 公開鍵暗号アクセラレータ

RSA 公開鍵暗号の暗号化・復号には M を平文，e,n は公開鍵，C を暗号文とし，

$$M^e \equiv C \bmod n \tag{5}$$

を計算する事により暗号文を得る．復号時は d を秘密鍵とし，

$$C^d \equiv M \bmod n \tag{6}$$

を計算する事により復号できる．RSA 公開鍵暗号の処理手順を図 3 に示す．

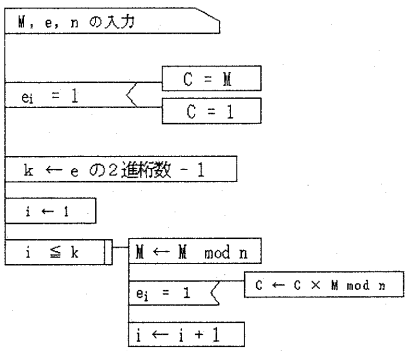

図 3　$M^e \equiv C \bmod n$ の処理

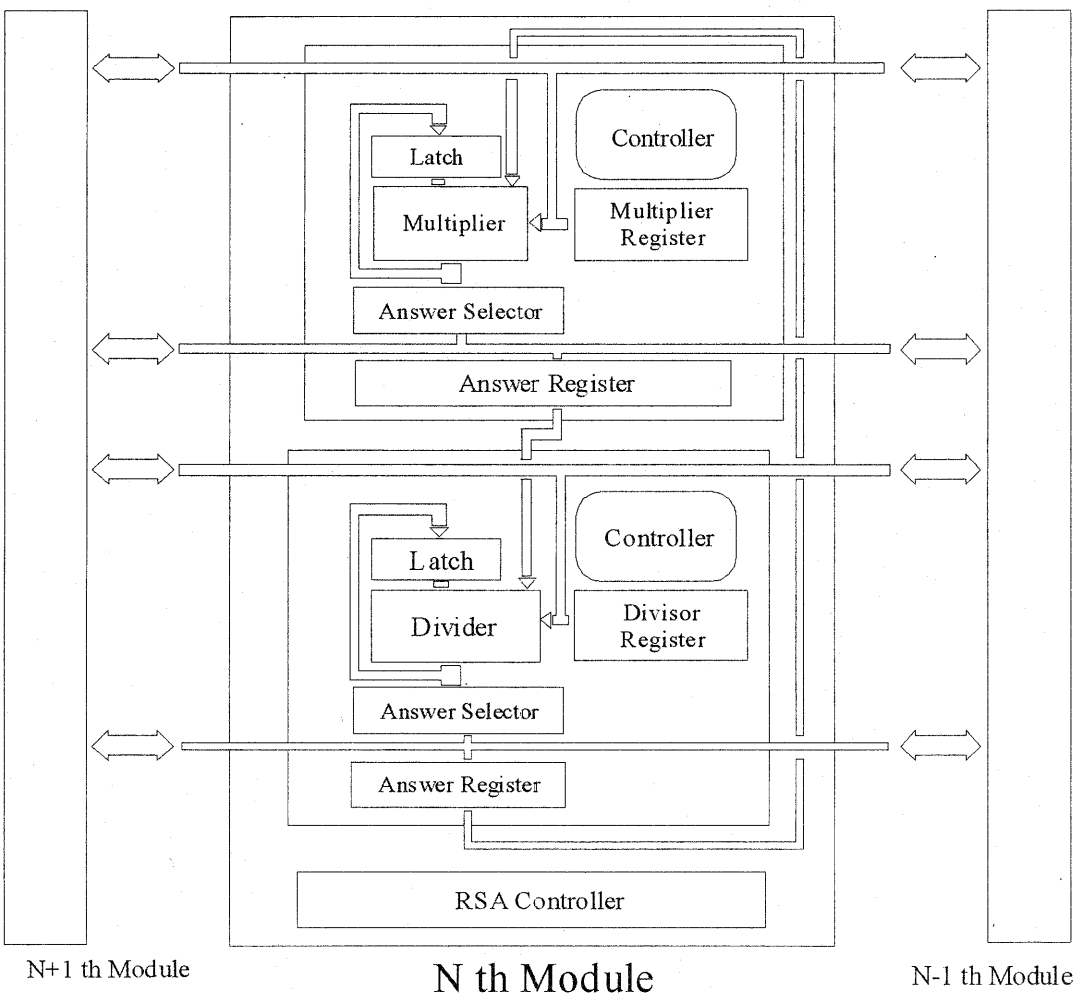

図 4　RSA 公開鍵暗号アクセラレータのブロック図

暗号化と復号は同じべき乗剰余アルゴリズムであるため，RSA 公開鍵暗号システムの高速化には，高速なべき乗剰余演算機構が必要となる．

本アクセラレータは冗長 2 進数表現を用いた準並列形べき乗剰余演算器を使って構成されており，順序回路形演算器を用いたアクセラレータよりも高速に暗号化・復号を行う事ができる．さらに優れた拡張性があり，必要とされる鍵長が 1 モジュールの演算精度を超えた場合でも，同アクセラレータをカスケード接続する事により演算精度を拡張する事ができる．鍵と平文の入力は通常の 2 進数で入力し，内部の演算は全て冗長 2 進数で行う事によりカスケード接続に際しても，演算器の遅延時間は一定に保たれる．暗号化・復号が終了した後，冗長 2 進数から通常の 2 進数に変換して出力する．本アクセラレータのブロック図を図 4 に示す．

4.1 アクセラレータ拡張時の動作例

アクセラレータは初期化時にあらかじめ演算ビット数を精度情報として入力する．それによりアクセラレータ内の制御回路が準並列形演算器のフィードバック回数，フィードバック時に発生する部分積と部分剰余のラッチ，また積，剰余を得た後，その解を保持すべき適切なモジュールに送る制御をする．アクセラレータを複数個接続した場合，鍵と平文をそれぞれ各モジュールに分割して入力するため，それらの分割された演算情報を他のモジュールに受け渡す事が必要となる．準並列形乗算器は最下位モジュールから順に乗数，準並列形除算器は最上位モジュールから順に被除数を最下位モジュールに入力するため，それぞれのモジュールに保持されている除数と被除数をアクセラレータ間のデータバスに適時出力する．この制御は乗算では最下位モジュールから順に乗数を，除算では最上位モジュールから順に被除数を出力し，出力終了後，次段のモジュールに制御信号を出力する．

モジュール数を n とすると，準並列形乗算器では下位モジュールに n-1 回だけ他のモジュールからの乗数入力，また下位モジュールから部分積が発生する．準並列形除算器でも同じく，下位モジュールに n-1 回の被除数の入力，演算終了時の剰余の保持がある．この各モジュールの乗数・被除数出力のタイミングや，部分積・剰余を各モジュールのレジスタに保持する制御は，初めに入力した精度情報より各モジュールの制御回路が独立して行う．

4.2 アクセラレータの試作

1 モジュールの精度を 4 ビットとした RSA 公開鍵暗号アクセラレータを 2 つカスケード接続して鍵長を 8 ビットとし，Altera 社の書き換え可能な CPLD「FLEX10KE」シリーズ (3 万ゲート相当) に書き込み，動作を確認した．

5. 比較・検討

5.1 演算時間

アクセラレータの制御ステップから，一回の処理にかかるクロック数を算出した．1 モジュールの演算精度を N，カスケード数を M とすると，RSA 公開鍵暗号アルゴリズムの制御に必要なクロック数は，

$$(9N-6) \tag{7}$$

また，べき乗剰余演算と制御に必要なクロックは

$$\frac{3}{2}(10M+5)(N-1) \tag{8}$$

となり，全体で必要なクロック数は

$$(9N-6)+\frac{3}{2}(10M+5)(N-1) \tag{9}$$

となる．

1 モジュールの鍵長精度とモジュールのカスケード数との必要クロック数を表 3 ～表 6 に示す．

表 3　1 モジュール 4 ビット

鍵長 (ビット)	クロック数	処理にかかる時間 (ms)		
		12(MHz)	24(MHz)	48(MHz)
256	249,017	20.75	10.38	5.19
512	989,561	82.46	41.23	20.62
1,024	3,945,209	328.77	164.38	82.19

表 4　1 モジュール 8 ビット精度

鍵長 (ビット)	クロック数	処理にかかる時間 (ms)		
		12(MHz)	24(MHz)	48(MHz)
256	126,617	10.55	5.28	2.64
512	499,001	41.58	20.79	10.40
1,024	1,981,049	165.09	82.54	41.27

表 5　1 モジュール 16 ビット精度

鍵長 (ビット)	クロック数	処理にかかる時間 (ms)		
		12(MHz)	24(MHz)	48(MHz)
256	65,417	5.45	2.73	1.36
512	253,721	21.14	10.57	5.29
1,024	998,969	83.25	41.62	20.81

表 6　1 モジュール 32 ビット精度

鍵長 (ビット)	クロック数	処理にかかる時間 (ms)		
		12(MHz)	24(MHz)	48(MHz)
256	34,817	2.90	1.45	0.73
512	131,081	10.92	5.46	2.73
1,024	507,929	42.33	21.16	10.58

また参考として，ソフトウェアで 1,024 ビットの RSA 公開鍵暗号処理にかかる時間を表 7 に示す．測定した環境は FreeBSD 上で RSAREF ライブラリを用い測定した．

表 7　計算機上での RSA 公開鍵暗号の処理時間

CPU	AMD Athlon Processor (701.24 MHz)	
OS	FreeBSD 4.4S	
回数	計算時間 [ms]	一回の計算時間 [ms]
5	1,613.747	322.749
105	33,956.725	323.397
255	82,447.311	323.322
455	147,103.818	323.305

CPU	AMD Athlon Processor (1393.79 MHz)	
OS	FreeBSD 4.4R	
回数	計算時間 [ms]	一回の計算時間 [ms]
5	789,.507	157.901
105	16,571.66	157.825
255	40,245.227	157.824
455	71,806.247	157.815

CPU	Intel Pentium Ⅲ (672.96 MHz)	
OS	FreeBSD 4.2R	
回数	計算時間 [ms]	一回の計算時間 [ms]
5	1,799.368	359.873
105	37,792.304	359.927
255	91,759.524	359.841
455	163,759.838	359.911

5.2 遅延時間

準並列形演算器に冗長 2 進表現を用いている事により，キャリー伝播遅延時間は最大一桁分で抑えられる．そのため，準並列形によるクロック数の減少と演算精度拡張時の遅延時間増加は順序回路形よりも大幅に少なくする事ができる．

6. むすび

情報ネットワーク環境の急速な普及により，電子商取引や，プライバシーの保護といったデータの安全性の確保が重要になっている．データのセキュリティのためには RSA 公開鍵暗号システムといった安全性の高い技術で暗号化するのが望ましい．しかし RSA 公開鍵暗号システムには，1,024 ビットのべき乗剰余演算といった演算量の多い処理が必要であり，ソフトウェアで大量のデータを処理するには時間がかかってしまう．そこで，専用のハードウェアによるアクセラレータを用いた公開鍵暗号システムの高速処理アーキテクチャの実現が望まれている．

本論文では冗長 2 進数による準並列形べき乗剰余演算器を用いた RSA 公開鍵暗号アクセラレータを提案した．準並列形べき乗剰余演算器は順序回路形演算器よりも演算に必要なクロック数が大幅に少なく，同モジュールをカスケード接続する事により容易に精度を拡張できるという特長がある．さらに，カスケード接続による桁上げ・桁借り伝播遅延時間を一定に抑える事のできる冗長 2 進数表現を用いた．本論文で提案する RSA 公開鍵暗号アクセラレータは，ソフトウェアで暗号化・復号を行った場合に比べ数十倍の高速化が見込め，さらに鍵長精度が拡大した時にもアクセラレータのカスケード接続で容易に対応できる．

文　献

[1] 山中 喜義, “情報セキュリティと電子商取引” 電学誌, 117 巻 6 号, pp.365-368(1997-6)

[2] 松井 充, “情報セキュリティ技術” 信学誌, vol.80,No4,pp.364-369(1997-4)

[3] Rovert W. Baldwin, C.Victor Chang, “Locking the e-safe” IEEE SPECTRUM,pp.40-46,FEB 1997

[4] 高木 直史, 安浦 寛人, 矢島 脩三, “冗長 2 進表現を利用した VLSI 向き高速除算器” 信学論 (D),Vol.67-D,No.4,pp.450-457(1984)

[5] 齋藤 正人, 日野杉充希, 恒川 佳隆, 三浦 守, “1 桁 2 ビット/3 ビット混合表現を用いた高速冗長 2 進加減算器の構成法” 信学技報,Vol.CAS99-41, VLD99-41, DSP99-57, pp.450-457(1984)

[6] 阿部 一広, 笠原宏, 中村次男, “冗長 2 進表現を用いた除算器のチップスライス化” 信学技報,VolCPM99-123, ICD99-219, pp.23-29(1999)

[7] 中村 次男, 大石 博朗, 笠原 宏 “, “RSA 公開鍵暗号システム実装におけるビットスライス化の一方式” 電学論 (C),Vol.118-C,No.7/8,July/Aug.,1998

[8] 中村 次男, 笠原 宏, “任意精度向き準並列形高速除算機構” 電学論 (C),Vol.120-C,No.1,Jan.,2000

[9] 中村 次男, 佐藤友威, 鈴川敦之, 阿部一広, 山口芳男, 笠原宏 “超高精度整数乗算器の高速化とモジュール化” 電学論 (C),Vol.121-C,No.7,July,2001