算術演算回路と暗号処理ハードウエアへの応用

葛 毅, 櫻井隆雄, 坂井修一, 田中英彦

1 はじめに

高速な算術演算回路に基づく暗号処理ハードウエアに関する研究を行う. 2 章で, 算術演算回路の中の除算回路に関するこれまで研究, 3 章で, 2 章の成果を土台とした, 現在の研究テーマである暗号処理の中の公開鍵暗号で用いられる乗算剰余演算のアルゴリズムと回路化に関する研究について述べる.

2 算術演算と除算回路

プロセッサの中心に位置する算術演算回路は高速である必要があり, 様々なアルゴリズムと回路構成法が研究されている. 本研究では, 除算回路の中で有効なアルゴリズムである SRT 除算 [1] を改良している.

SRT 除算の一般的な構成法は, 商の桁の選択を表を引く事で行うテーブルモデル [2] である. これに対して, 本研究では, 高基数 SRT 除算で商の桁を選択するのにテーブルではなく算術演算を用いる回路構成を提案し, 誤差計算を行い, 評価している [3].

以下に示す各々数種類の回路を設計し, 論理合成ツールで合成して得られる速度と面積の値を用いて比較している. 設計には VDEC で製作された CMOS 0.6μm のセルライブラリを用いている. 商を選択する回路以外は, 手作業で設計している.

- rX_restore : 基数 X の回復法.
- r2_srt: 基数 2 の SRT 除算.
- rXqY_arith: 基数 X, q_{max}=Y の算術モデル.
- rXqY_table: 基数 X, q_{max}=Y のテーブルモデル.
- r4x4q2_overlap: 基数 4, $q_{max} = 2$ のテーブルモデルを二つオーバーラップさせた基数 16[4].

図 1 は各手法の 16, 32, 54, 114 ビットでの比較結果である. 縦横の軸は面積と時間であり, 原点に近いほど性能が良い. 図より 32 ビット以下では高基数回復法, 54 ビット以上では算術モデルが有効であることが分かる. 本手法は, 回路面積を抑えた高速化に成功している.

3 公開鍵暗号と乗算剰余演算回路

近年, インターネットの広い普及により, 暗号処理が必要になってきており, 装置が高速, 小型であることが要求されている. 特に, 公開鍵暗号は長いビットの剰余演算を必要とし, 処理が重いため, 各種暗号アルゴリズムのハードウエア実装に関する研究が盛んになされている.

暗号アルゴリズムは, 秘密鍵暗号と公開鍵暗号に大別される. 本研究では, 除算回路の研究を基礎として, 算術演算を必要とする公開鍵暗号に着目する.

公開鍵暗号で一般的な RSA 暗号の高速化には, 非常にビットの長い冪剰余演算の高速化が重要であり, この演算は乗算剰余演算に分解できる. この乗算剰余演算 [5] は, 一般的に 1024 ビット程度の大きな数 N による剰余をとる必要がある. これには次の回路構成手法がある.

1. 乗算と除算で行う方法 [6].
2. モンゴメリのアルゴリズムを用いる方法 [7].

1. の方法については, 除算を用いた剰余演算の高速化が問題になる. これには 1024 ビットという長ビットの除算が必要になり時間がかかる. SRT 除算を用いることにより加算での桁上げをなくすことで遅延時間を小さくできる.

2. の方法は, モンゴメリにより提案された乗算剰余演算の高速化アルゴリズムである [7]. 近年は, このモンゴメリ乗算に基づく回路構成法に関する論文が多い [8][9]. これは剰余演算で必要な除算をなくし, 乗算のみで行うことにより高速化する. x, N を整数, N を n ビットの奇数, $R = 2^n$ とし, $0 \leq x < RN$ とする. このとき, $x \bmod N$ を計算したいとすると, x

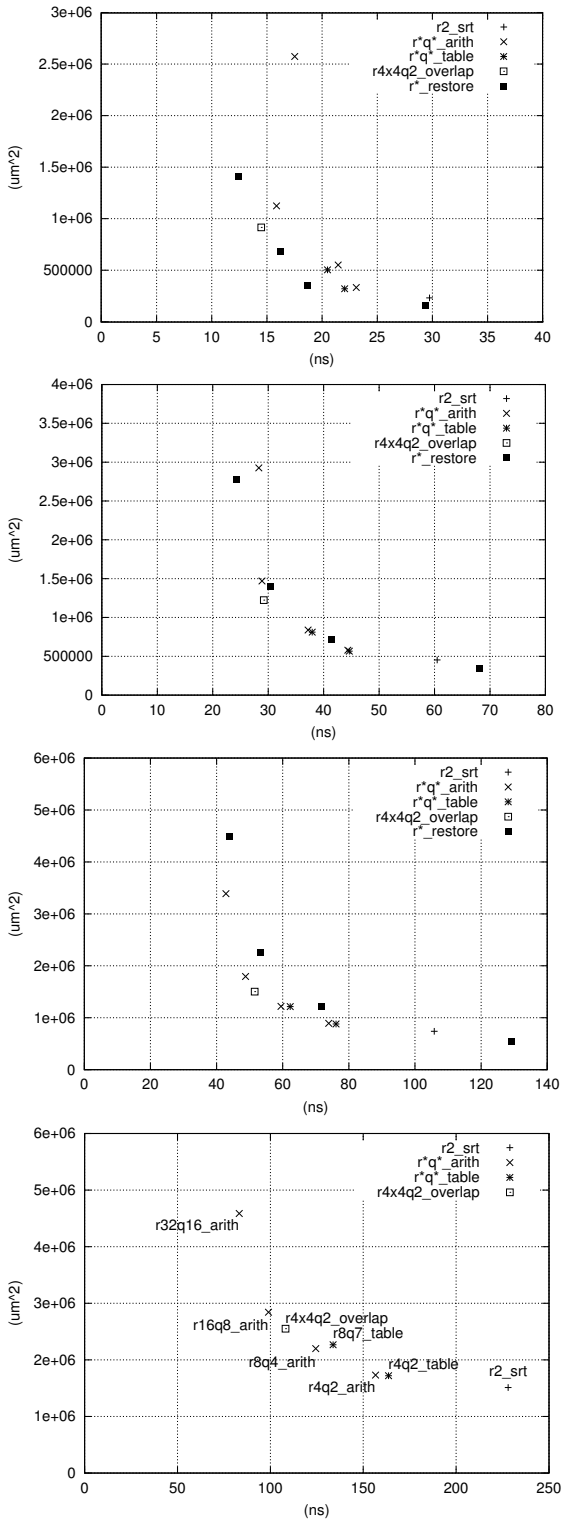

図 1: 除算回路の各手法の比較: 上から 16, 32, 54, 114 ビットの場合.

に N の整数倍 aN を足した数 $x + aN$ の N による剰余 $(x + aN) \bmod N$ は, $x \bmod N$ と等価である. ここで a に適切な数を選ぶと, $x + aN$ は R で割りきれるようにできる. $x + aN$ を n ビット右シフトして簡単な補正をすれば, それは $xR^{-1} \bmod N$ と同じになる. 以上を行う手続きが次の $\mathrm{REDC}(x)$ である. x を与えると $xR^{-1} \bmod N$ が得られる. なお, N' は $0 < R^{-1} < N$, $0 < N' < R$, $RR^{-1} - NN' = 1$ を満たす整数である.

function $\mathrm{REDC}(x)$

$a := (x \bmod R)N' \bmod R$;

$X := (x + aN)/R$;

if $X \geq N$ then return $X - N$ else return X;

ここで, $z = x \cdot y \bmod N$ を計算したい場合は, 予め, $X = \mathrm{REDC}(x(R^2 \bmod N)) = xR \bmod N$ と $Y = \mathrm{REDC}(y(R^2 \bmod N))$ を計算しておき, $Z = \mathrm{REDC}(XY) = XYR^{-1} \bmod N$ を計算してから, $\mathrm{REDC}(Z) = ZR^{-1} \bmod N = z$ を計算する.

現在は, 上記 1. 2. の各々に基づく乗算剰余演算回路の典型的な回路の性能を面積と速度に基づき概算している. 今後は, 各々の手法を比較して, 有効な手法を決定し, その手法に基づく新しい回路構成法を検討する.

参考文献

[1] J. E. Robertson: "A New Class of Digital Division Methods," *IRE Trans. Electronic Computers,* vol. EC-7, no. 9, pp. 218-222, Sept. 1958.

[2] N. Burgess and T. Williams: "Choices of Operand Truncation in the SRT Division Algorithm," *IEEE Trans. Computers*, vol.44, no.7, pp.933–938, July 1995.

[3] 葛 毅, 阿部公輝, 浜田穂積: "高基数 SRT 除算の論理回路実現に基づく回路構成と評価," 情報処理学会論文誌, vol.43, no.8, pp.2665–2673, Aug. 2002.

[4] G. S. Taylor: "Radix 16 SRT Dividers With Overlapped Quotient Selection Stages," *Proc. 7th IEEE Symp. Computer Arithmetic,* pp. 64-71, 1985.

[5] 高木直史: "初等関数計算回路のアルゴリズム," 情報処理, vol. 37, no. 4, pp. 362-368, Apr. 1996.

[6] N. Takagi: "A Radix-4 Modular Multiplication Hardware Algorithm for Modular Exponentiation," *IEEE Trans. Computers*, vol. 41, no. 8, pp. 949–956, Aug. 1992.

[7] P. L. Montgomery: "Modular Multiplication Widthout Trial Division," *Mathematics of computation,* vol. 44, no. 170, pp. 519–521, Apr. 1985.

[8] S. E. Eldridge, and C. D. Walter: "Hardware Implementation of Montgomery's Modular Multiplication Algorithm," *IEEE Trans. Computers*, vol. 42, no. 6, pp. 693–699, June 1993.

[9] 新保淳, 野崎華恵, 川村信一: "高速 RSA 暗号 LSI," 東芝レビュー, vol. 55, no. 7, 2001.