C-036

モンゴメリ逆元算のFPGA化設計

グェン・デゥク・クワン† 岩井 啓輔† 黒川 恭一†
†防衛大学校 情報工学科

1 はじめに

公開鍵暗号方式は，鍵交換（鍵配送）のための安全経路を別途設ける必要がない，電子署名が可能である等の特徴があり，その一つとして，楕円曲線暗号（ECC - Elliptic Curve Cryptography）が注目されている．

楕円曲線暗号を始めとする公開鍵暗号方式の計算量は，秘密鍵暗号方式と比べて大きく，その高速化は重要な課題である．楕円曲線暗号方式では，暗号化や復号に曲線上の点のスカラー倍を求める必要があり，そのために異なる二点の加算と二倍算を繰り返す．これらの演算は，体上の加算，減算，乗算，平方，逆元算等に基づいており，有限体上では特に逆元算の計算量が大きくなる．そのため，高速なモジュラー逆元算が必要となっている．

本研究では ECC への使用を念頭に入れて，GF(p) 上のモンゴメリ逆元算回路を FPGA に実装し，その処理速度や回路規模の面からの性能評価・検討を行う．

2 モンゴメリ逆元算の概要

2.1 モンゴメリ逆元算

ECC の演算では，最初に整数をモンゴメリドメインに変換する．全てのモジュラー演算はこのモンゴメリドメインで行われ，その演算結果は整数値に再変換される．このモジュラー演算の中にモンゴメリ逆元算が含まれている．

2.2 モンゴメリ逆元算アルゴリズム

整数 a のモンゴメリモジューロ逆元

$$X := \mathrm{MonInv}(a) = a^{-1}2^{n} \pmod{p} \qquad (2.1)$$

を求めるアルゴリズムとして，Kaliski は文献 [2]で以下に示す方式を提案した．ただし，a は $[1, p-1]$ に含まれるものとする．また n は p のビット数である．このアルゴリズムは２つの段階で構成される．第１段階は AlmMonInv を呼ばれ，入力 a と p に対して，

$$r := \mathrm{AlmMonInv}(a) = a^{-1}2^{k} \pmod{p} \qquad (2.2)$$

$$n \le k \le 2n$$

となる r と k を計算する．その計算過程を表１に示す．続く第２段階は，その入力として第１段階の出力を受け取り，最終の結果 $a^{-1}2^{n} \pmod{p}$ を出力する．その計算過程を表２に示す．

3 ハードウェア構成

3.1 アーキテクチャ

AlmMonInv アルゴリズムの第１段階では，基本的な関数を繰り返すループ構造を持っている．本研究では，このようなアルゴリズムに対して，ループ・アーキテクチャによる実現を図った．そのループ回数は表 1 に示した v の値によるため，この構成では可変となる様に設計した．

表１ アルゴリズムの第１段階（AlmMonInv）

```
AlmMonInv ( a, p )
Input : a ∈ [1, p - 1] , p
Output : r & k,  r = a^-1 2^k (mod p), n ≤ k ≤ 2n
u := p, v := a, r := 0, s := 1
k := 0
while (v > 0){
if u is even then u := u/2, s := 2s
else if v is even then v := v/2, r := 2r
else if u > v then u := (u - v)/2, r := r + s, s := 2s
else if v ≥ u then v := (v - u)/2, s := s + r, r := 2r
k := k + 1}
if r ≥ p then r := r - p
return r := p – r
```

表２ MonInv アルゴリズムの第２段階

```
MonInv ( r, p, k )
Input : r,  p  and k from AlmMonInv
Output : x, where x := MonInv ( a ) =   a-12n (mod p)
for i = 1 to ( k - p ) {
if r is even then r := r/2
else r := (r + p)/2}
return x := r
```

図１に AlmMonInv アルゴリズムをループ・アーキテクチャによって実現したブロック図を示す．まず，処理の始めにレジスタ a と p の値を入力する．これらの値は，レスタ u, v, r, s の値と共に，comp. ブロックに入力され,comp. ブロックにおいて，表１の while 文中の各処理を行った後，各レジスタの値が更新される．一回のループが終わる都度，kの値は１加えられる．

各レジスタに格納される変数のうち u, v, r, s 等はループ処理の進行に伴い，ビット長が順次短くなってゆく．それらの初期値は全て n であり，順次 $n/2$, $n/4$, ... となってゆく．そこで，本研究では n ビット長のハードウェアを用意する図２の様な統合法と，w ビット に分割して処理してゆく図

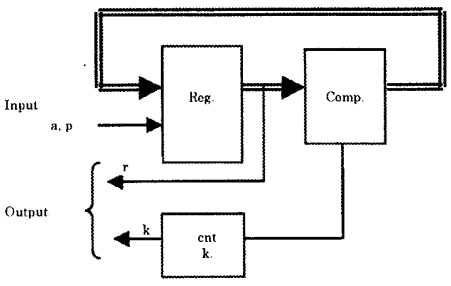

図１：AlmMonInv のブロック図

Implementation of Montgomery Inverter on an FPGA
†Nguyen Duc Quang, Keisuke IWAI and Takakazu KUROKAWA
Department of Computer Science, National Defense Academy

３の様な分割法の二つの方法を検討した.

図２に示した統合法の中で, すべての処理はｎビット単位で行われる. 計算部はすべてのクロックサイクルにおいてこの n bit をまとめて計算する. 本報告では, 統合法の実装結果を主に示してゆく.

一方の図３に示した分割法の場合, 計算部の処理はｗビット単位に順次行われる. 従って k 回目のループで処理する変数のビット長は $n/2^{k-1}$ となり, 各ループで ｗ ビット単位に処理する回数は $\lceil n / 2^{(k-1)w} \rceil$ となる.

3.2 統合法の実現

統合法の構成を図４に示す. 図中左上にある sub により２つの減算 $(u\text{-}v$ と$(v\text{-}u)$が並列に実行される. これらは $(p\text{-}r)$ を計算する時にも再利用される. rs1 は$(u - v)/2$ と $(v - u)/2$ を実行するために用いられる１ビット右シフトする回路であるが, 実際には配線論理により実現している. $(r+s)$ を計算するために、図中左下の add が用意されている. この様にして更新された u, v, r, s の新しい値はレジスタに保持される.

3.3 実装・評価環境

今回の実装では, Xilinx 社製の Virtex-II XC2V6000 を用いた KAC-02A という大規模 FPGA ボードを利用した. また, 回路設計には Verilog HDL を用い, システムデザイン用のツールに ISE 4.1.03i を利用した.

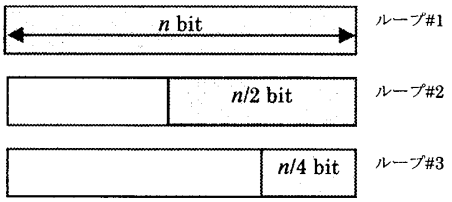

図２：統合法

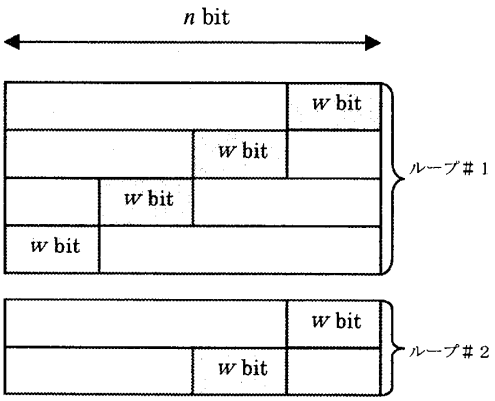

図３：分割法

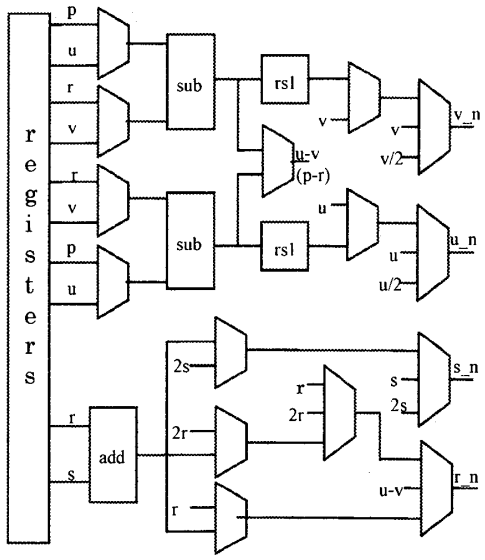

図４：統合法の構成

4 実装結果

表３に今回の実装結果をまとめる. 全体の計算時間は, 第１段階に要するクロックサイクルの数 $(k+2)$ と第２段階に要するクロックサイクル数 $(k - n)$ の和となる.

表３ 実装結果

Bit length	256
Gate counts	43,735
Slices	2,065
Frequency (MHz)	20.28

5 おわりに

本稿では, モンゴメリ逆元算回路を FPGA 実装した. 今後の課題としては,分割法の実装や, 両者の性能向上としてキャリーチェーンの改良法の検討が上げられる.

参考文献

[1] Montgomery, P.L., "Modular Multiplication Without Trail Division", *Mathematics of Computation*, 44(170):519-521, April 1985.
[2] Kaliski, "The Montgomery inverse and its applications", *IEEE Transactions on Computers*, Volume 44 , Issue 8, pp.1064 – 1065, Aug., 1995.
[3] Savas, and Koç, "The Montgomery Modular Inverse Revisited", *IEEE Transactions on Computers*, 49(7):763-766, July 2000.
[4] Kobayashi, and Morita, "Fast Modular Inversion Algorithm to Match Any Operation Unit", IEICE *Transactions Fundamentals*, E82-A(5):733-740, May 1999.
[5] Adnan Abdul-Aziz Gutub, Alexandre Ferreira Tenca, and Çetin Kaya Koç, "Scalable VLSI Architecture for GF(p) Montgomery Modular Inverse Computation", *Proceedings of the IEEE Computer Society Annual Symposium on VLSI* pp.53-58, April 2002.