高速モンゴメリ乗算回路に関する研究

吉原 智明 (110128)

北陸先端科学技術大学院大学 情報科学研究科

2003 年 2 月 14 日

キーワード: 公開鍵暗号, RSA, 乗算剰余演算, モンゴメリ乗算, FPGA.

近年, 通信ネットワークを介して重要な情報が流通するようになり, その内容を外部から盗聴されないようにするための暗号化技術が必要になってきている.

暗号方式として最初に出た秘密鍵暗号方式は, 送信者と受信者が共通の鍵を用いて暗号化と復号化を行うので, 鍵を秘密に配送する必要があった. しかし, 公開鍵暗号方式は, 送信者が暗号化に使う鍵と受信者が復号化に使用する鍵が異なるので, 鍵の配送と管理が容易になる. また, 複数の送信者が共通の暗号鍵を共有することもできる. よって, インターネットのような不特定多数間での暗号通信を可能とする方式として広く研究が行われている.

公開鍵暗号方式で重要なのは, 暗号鍵を公開してもそれを用いて復号鍵を求めることが容易ではない, つまり計算量的に不可能である点である. 代表的な公開鍵暗号方式である RSA 暗号方式, 楕円暗号方式においては, 暗号鍵からの復号鍵算出に前者では素因数分解問題, 後者では楕円離散対数問題が使われる. これらを解くにはスーパーコンピュータを用いても数百年以上という非現実的な時間がかかると言われており, 両暗号方式の解読に対する安全性はそれに依存している.

公開鍵暗号方式の多くは, それまで知られてきた秘密鍵暗号方式よりも計算量が多く, 暗号化復号化に膨大な計算時間を必要とする. 近年の CPU の高速化により, ソフトウェア実装でも実用的な速度が得られるようになってきているが, スマートカードを代表とするローエンド向けの 8 ビットや 16 ビットの組み込み CPU では処理が遅く実用に耐えない. また SSL サーバーなど暗号処理が集中する用途において, ソフトウェア処理では十分なパフォーマンスを得ることができない. そのためローエンド, ハイエンド双方において暗号処理専用ハードウェアが必要となる.

本研究では, 主な公開鍵暗号方式で大きな整数に対する剰余演算が繰り返し行われていることに着目する. その演算部分を高速化することができれば, それによって, 公開鍵暗号用のハードウェア全体を高速化することができると考える. そして, 剰余演算を行うアルゴリズムには, いろいろなものが知られているが, 本研究では, モンゴメリ乗算アルゴリズムを取り上げる. このアルゴリズムは, 除算処理を利用せずに剰余付き乗算を計算する

手法である. 通常, 多倍長整数での四則演算においてもっとも処理時間が掛かるのが除算であるため, もっとも有効なアルゴリズムのひとつである.

具体的にここで取り上げた演算は, 法 N とそれより小さな数 A と B を入力として, $A \cdot B \cdot R^{-1} \bmod N$ を出力する. ここで R は $2^{r \cdot m-1} \leq N < 2^{r \cdot m} = R$ とし, 被乗数 A を基数 2^r として m 個のブロックに分けて部分積剰余をもちいた演算である. これは $A \bmod N$ などにも対応可能であり汎用といえる. このアルゴリズムは R で除算するところを $r \cdot m$ ビットの右シフトに置き換えたりすることで除算を省いているので, ハードウェアで実装しやすいというメリットがある.

本研究ではこのモンゴメリ乗算アルゴリズムを FPGA を用いてハードウェア化を行い高速化を図る. ハードウェアとして, 18bit の高速な乗算器が搭載された Xilinx 社の XC2V3000 という 300 万ゲートの FPGA を用いた. 記述には, ハードウェア記述言語 Verilog HDL を使用し, Xilinx 社の論理合成ツールである ISE ロジックデザインツールを用いてコンパイルをおこなう. そのツールを使うことにより, RTL（回路の構成要素であるレジスタやカウンタなどと, それらの間でのデータの転送状態（接続状態）をクロックの概念をいれて表現しているレベル）で書かれたソースコードを, FPGA に直接ダウンロードできる形であるネットリストに合成することができる. また実際にハードウェアとしてシミュレーションを行い, 実際の演算にかかる時間を計測することができる

今回実際に構成した演算回路は法 N, A, B をそれぞれ, 1024bit とする $A \cdot B \cdot R^{-1} \bmod N$ の演算回路である. FPGA に 18bit の高速な乗算器が搭載されたものを使用したので, 一回の乗算は 16bit 単位で区切り $r = 16$ とした.

本研究の解析には, ISE ロジックデザインツールを使用したが, 実際の演算時間としたものは, レジスタに入力値を入れる時間を除いた計算のみの時間で, それは最大遅延時間 $\times$ クロックサイクル数として導いた. また一般的なソフトウェア上のモンゴメリ乗算の時間を 3 台の異なるパソコン環境で時間を測定することにより, 比較を行った.

結果として, 本研究では, ソフトウェアに対して約 5 〜20 倍速くモンゴメリ乗算を行えるハードウェアを実現することができた.