コンピュータセキュリティ　14－16
（2001. 7. 25）

スマートカード向けAESハードウェアの試作

岡田 壮一[†]　鳥居 直哉[†]　長谷部 高行[‡]

†‡株式会社 富士通研究所
†〒674-8555 明石市大久保町西脇 64
‡〒211-8588 川崎市中原区上小田中 4-1-1
E-mail: {sokada, torii, hasb}@flab.fujitsu.co.jp

あらまし　次世代共通鍵ブロック暗号 AES に採用された Rijndael アルゴリズムのハードウェア実装について、スマートカードに搭載可能なハードウェア構成を示し、その ASIC による性能評価、および FPGA 試作を行った。本ハードウェアは、3 種の鍵長(128 ビット、192 ビット、及び 256 ビット)での暗号化／復号処理が可能であり、更に CBC モードの処理もサポートしている。富士通 0.35 μm CMOS ASIC でのシミュレーションによる性能評価の結果、回路規模は 17.6K ゲートで、鍵長 128 ビットの時のスループットは 73Mbps である。また、FPGA は ALTERA 社の EP1K100QC208-3 を用い、最大 32MHz で動作し、鍵長 128 ビットの時のスループットは、44Mbps である。
キーワード　共通鍵ブロック暗号, AES, Rijndael, FPGA

AES hardware implementation for smart cards

Souichi Okada[†]　Naoya Torii[†]　Takayuki Hasebe[‡]

†‡FUJITSU LABORATORIES LTD.
†64, Nishiwaki, Ohkubo-cho, Akashi 674-8555, Japan
‡4-1-1, Kamikodanaka, Nakahara-ku, Kawasaki 211-8588, Japan
E-mail: {sokada, torii, hasb}@flab.fujitsu.co.jp

Abstract　We describe a hardware implementation of AES(Rijndael) for smart cards. We show the hardware configuration, the result of the FPGA implementation and the performance evaluation of the ASIC implementation using 0.35 μm CMOS ASIC. The hardware encrypts/decrypts the data with 128 bit, 192 bit, and 256 bit key. And it supports CBC mode of operation. In the FPGA implementation by ALTERA EP1K100QC208-3, the throughput is 44 Mbps at 32 MHz when key length is 128 bit. In the ASIC implementation, the hardware size is 17.6 Kgates, and the throughput is 73 Mbps when key length is 128 bit.
Key Words　Block cipher, AES, Rijndael, FPGA

1 はじめに

米商務省技術標準局(NIST : National Institute of Standards and Technology)は、DES(Data Encryption Standard)に代わる次世代の共通鍵ブロック暗号 AES(Advanced Encryption Standard)の公募選定を行い、2000 年 10 月に Rijndael が選ばれた。

本稿では、Rijndael アルゴリズムについて、スマートカードに暗号コプロセッサとして搭載するためのハードウェアについて述べる。Rijndael アルゴリズムのハードウェア実装については、様々な実装結果が報告されている[1][3][4][5][6][12]。これらは、FPGA や ASIC を用いて Rijndael の高速実装を行ったものである。また、小型実装として、Rijndael アルゴリズムの 1 段分のラウンド処理を実装評価した結果も報告されている[2][11][13]。

本稿では、今回の試作にあたり想定したスマートカード実装のための目標諸元を示し、それに基づいたコプロセッサのハードウェア構成、及び試作結果について述べる。コプロセッサは、128 ビット、192 ビット、及び 256 ビットの 3 種の鍵長に対し、データの暗号化／復号処理を行う。更に、ECB モードと共に CBC モードにも対応している。本コプロセッサを、当社 0.35μm CMOS ASIC[10] を用いて実装評価したところ、回路規模 17.6K ゲート(CBC モード処理含む)であり、最大処理遅延で評価した処理速度は、鍵長 128 ビットの時 73Mbps となった。尚、本コプロセッサは、FPGA で試作し、動作チェックを完了している。

2 目標諸元

設計にあたり、スマートカード実装のために想定した目標諸元について説明する。

スマートカードには、大別して接触型、非接触型の 2 種のスマートカードが存在する。今回の実装にあたって、処理速度は、スマートカードの動作クロックでインタフェース速度以上の処理速度があることを目標とした。接触型では 3.57MHz で最大 56Kbps、非接触型では近接型(ISO 14443)を想定し、13.56MHz において 848Kbps である。よって、目標処理速度は、より高速処理が必要な 848Kbps(@13.56MHz)以上とする。

回路規模に関しては、スマートカードに搭載可能なチップ面積は 25mm² であり、このうち暗号コプロセッサとしては、小規模である程良いが、一応の目安として 30K ゲート以下とする。

表 1 目標諸元

プロセス	0.35μm CMOS ASIC
処理速度	848Kbps(@13.56MHz)以上
回路規模	30K ゲート以下
鍵長	128bit、192bit、256bit
モード処理	ECB、CBC

鍵長は、将来的な使用を可能とするため、3 種の鍵(128 ビット、192 ビット、及び 256 ビット)に対応する事とする。

また、ファイルの暗号化が容易な様に ECB モードに加え、CBC モードにも対応する事とする。

実装プロセスは、現在、スマートカード向けに使用されている 0.35μm CMOS ASIC とする。

以上、目標諸元を表 1 にまとめる。

3 ラウンド処理部

Rijndael アルゴリズム[7][8]は、ラウンド処理を規定回数繰り返し行うアルゴリズムである。全アルゴリズムをハードウェア実装する方法もあるが、回路規模が約 600K ゲートという評価結果[1]もあり、基本単位(ラウンド処理)をハードウェア実装し、それを繰り返し使用するループ処理を採用する。ラウンド処理には、ByteSub(InvByteSub)変換、ShiftRow(InvShiftRow)変換、MixColumn(InvMixColumn)変換、Round Key 加算の処理が必要であり、図 1 に本試作ハードウェアで用いた基本構成ブロック図を示す。ShiftRow 変換部の出力を ByteSub 変換部あるいは Round Key 加算部 2 あるいはセレクタ 2 にフィードバックした 3 つのループで構成する。128 ビット入力データに対して、セレクタ 1 で n ビット単位(n<128 の 8 の倍数)に取り出し、内部を n ビット処理とすることにより、回路規模の削減を図る(5 章で詳述するが、本ハードウェアでは、32 ビット処理を採用した)。本章でのゲート数は、当社 0.35μm CMOS ASIC[10]での評価値である。

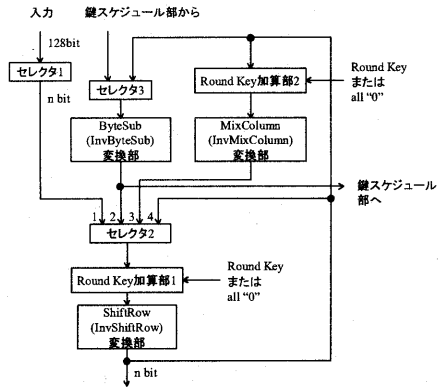

図1　ラウンド処理部基本構成ブロック図

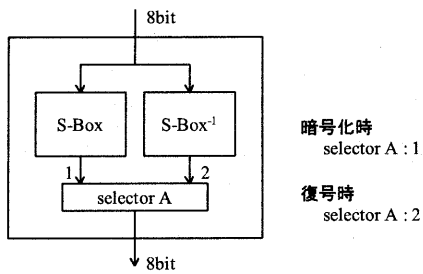

図2　S-box/S-box^{-1}によるByteSub(InvByteSub)変換部基本構成ブロック図

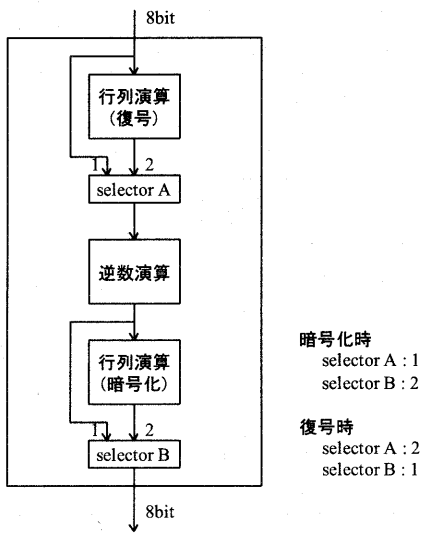

図3　本ハードウェアでのByteSub(InvByteSub)変換部基本構成ブロック図

3.1　ByteSub(InvByteSub)変換部

ByteSub(InvByteSub)変換部では、暗号化の時は、逆数演算、暗号化行列演算の順番で処理を行う。復号の時は、暗号化の逆なので、復号行列演算、逆数演算の順番で処理を行う。AES FIPS 仕様書(draft)[8]では、変換テーブルでの実現方式(図2)も記載しているが、この方式では、暗号化用と復号用の 8 ビットテーブル(S-box、S-box^{-1})が 2 個必要となる。回路規模の削減を図るために、本ハードウェアでは、逆数演算のみ変換テーブルで実装し、アフィン変換(行列演算)は、論理回路で実装する(図 3)。

また、ラウンド処理部の処理ビット n が、8 ビットの時、ByteSub(InvByteSub)変換部は、図 3 の構成となるが、16 ビット、32 ビットの時は、それぞれ、2 並列、4 並列の構成となる。

3.1.1　逆数演算部

本ブロックは、GF(2^8)上の逆数演算(既約多項式 $M(x) = x^8 + x^4 + x^3 + x + 1$)を、逆数変換テーブルを用いて実現する。実装方式としては、コンパイルド ROM を用いて ROM で構成する方式と、論理合成回路を用いてランダムロジックで最適化する方式がある。各方式での回路規模は、以下のとおりであり、本ハードウェアでは、より回路規模の小さい論理合成回路を採用する。

コンパイルド ROM　1800 ゲート
論理合成回路　834 ゲート

3.1.2　アフィン変換(行列演算)部

入力($x_7, x_6, x_5, \cdots, x_1, x_0$)に対して、以下の行列式に基づいて、出力($y_7, y_6, y_5, \cdots, y_1, y_0$)の行列演算を行う。暗号化および復号の時の行列式は、以下のとおりである。

(1)暗号化

$$\begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{pmatrix} = \begin{pmatrix} 1&0&0&0&1&1&1&1 \\ 1&1&0&0&0&1&1&1 \\ 1&1&1&0&0&0&1&1 \\ 1&1&1&1&0&0&0&1 \\ 1&1&1&1&1&0&0&0 \\ 0&1&1&1&1&1&0&0 \\ 0&0&1&1&1&1&1&0 \\ 0&0&0&1&1&1&1&1 \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

表2　ByteSub(InvByteSub)変換部回路規模

本構成(図3)	アフィン(70) + 逆数変換テーブル(834) = 904 ゲート
2個の S-box 構成(図2)	S-box(620) + S-box^{-1}(607) = 1227 ゲート[2]

上記暗号化行列式は、16 個の XOR と 4 個の NOT で構成でき、本ハードウェアでは 1 サイクルで演算している。

(2)復号

$$\begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{pmatrix} = \begin{pmatrix} 00100101 \\ 10010010 \\ 01001001 \\ 10100100 \\ 01010010 \\ 00101001 \\ 10010100 \\ 01001010 \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

上記復号行列式は、13 個の XOR と 2 個の NOT で構成でき、本ハードウェアでは 1 サイクルで演算している。

3.1.3　回路規模比較

本構成で実装した場合(図 3)と 2 個のテーブル(S-box、S-box^{-1})で実装した場合[2](図 2)の回路規模の比較を表 2 に示す。表 2 より、本構成は、図 2 の構成よりも 323 ゲート小さく、32 ビット処理の ByteSub(InvByteSub)変換部全体では、1292 ゲート小さくなる。

3.2　ShiftRow(InvShiftRow)変換部

Rijndael アルゴリズムでは、ShiftRow 変換と InvShiftRow 変換のシフト処理の向きが逆なので、暗号化と復号用の回路が必要になる。しかし、暗号化と復号でデータの処理の順番を逆にすることで、シフト処理が共通化できることに着目し、1 つの回路で ShiftRow(InvShiftRow)変換できる構成を提案する。基本回路構成を図 4 に示す。図 4 では、入力 0-3 および出力 0-3 は各 8 ビットデータであり、FF00-33 は、8 ビットのフリップフロップである。図 4 のような構成とすることにより、ByteSub 変換後あるいは InvMixColumn 変換後のデータ格納用レジスタとしても兼用できる。

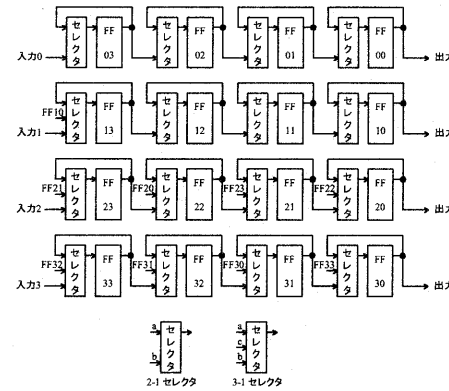

図 4　ShiftRow(InvShiftRow)変換部基本回路構成

また、暗号化と復号で別々の回路を持つ場合と比較すると、本構成では、2 行目と 4 行目のセレクタ(8 ビットの 2-1 セレクタ 8 個)が削減でき、200 ゲート程度の回路規模の削減となる。

(1)データ処理順とレジスタ機能

Rijnadael アルゴリズムでは、128 ビットデータを 8 ビット単位に 16 分割し、下のような 4×4 配列として処理を行う。

暗号化開始の時、Round Key 加算、ByteSub 変換後に、ShiftRow 変換を行う。復号開始の時、Round Key 加算後に、InvShiftRow 変換を行う(ただし、2 ラウンド以降は InvMixColumn 変換後に、InvShiftRow 変換を行う)。

暗号化 →　　←復号

$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$
$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$
$a_{2,0}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$
$a_{3,0}$	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$

本ハードウェアでは、暗号化の時、Column データ列($a_{0,0}$, $a_{1,0}$, $a_{2,0}$, $a_{3,0}$)から行い、復号の時は、逆に、Column データ列($a_{0,3}$, $a_{1,3}$, $a_{2,3}$, $a_{3,3}$)から行う。尚、処理順を逆にしても、ByteSub (InvByteSub) 変換および MixColumn (InvMixColumn)変換に影響を及ぼさないのは明白である。

ByteSub 変換もしくは InvMixColumn 変換結果のデータを、図 4 の入力 0-3 に各 8 ビット単位で入力し、4 サイクルで 128 ビットのデータ入力

が完了する。この時、データを8ビット単位でシフトしながら格納する。従って、1サイクル目に入力された4個の8ビットデータは、データ入力完了時には、FF00、FF10、FF20、FF30に、それぞれラッチされている。

(2)ShiftRow機能

(2-1)暗号化(ShiftRow変換)

変換前 ⇒ 変換後

$a_{0,3}$	$a_{0,2}$	$a_{0,1}$	$a_{0,0}$
$a_{1,3}$	$a_{1,2}$	$a_{1,1}$	$a_{1,0}$
$a_{2,3}$	$a_{2,2}$	$a_{2,1}$	$a_{2,0}$
$a_{3,3}$	$a_{3,2}$	$a_{3,1}$	$a_{3,0}$

左3Byte
左2Byte
左1Byte

$a_{0,3}$	$a_{0,2}$	$a_{0,1}$	$a_{0,0}$
$a_{1,0}$	$a_{1,3}$	$a_{1,2}$	$a_{1,1}$
$a_{2,1}$	$a_{2,0}$	$a_{2,0}$	$a_{2,2}$
$a_{3,2}$	$a_{3,1}$	$a_{3,0}$	$a_{3,3}$

(1)で述べた処理順で、ByteSub変換後のデータは4サイクルかけて本ブロックに入力され、上記のような配列で各FFにラッチされている。

次に、ShiftRow変換時には、各セレクタを切り換えることにより、以下のシフト処理を1サイクルで行う。

1行目(FF00-03)については、シフトしない。2行目(FF10-13)については、左3Byteシフトする。3行目(FF20-23)については、左2Byteシフトする。4行目(FF30-33)については、左1Byteのシフトする。この結果、ShiftRow変換後は、上記のような配列で各FFにラッチされている。

(2-2)復号(InvShiftRow変換)

変換前 ⇒ 変換後

$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$
$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$
$a_{2,0}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$
$a_{3,0}$	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$

左3Byte
左2Byte
左1Byet

$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$
$a_{1,3}$	$a_{1,0}$	$a_{1,1}$	$a_{1,2}$
$a_{2,2}$	$a_{2,3}$	$a_{2,0}$	$a_{2,1}$
$a_{3,1}$	$a_{3,2}$	$a_{3,3}$	$a_{3,0}$

(1)で述べた処理順で、InvMixColumn変換後のデータは4サイクルかけて本ブロックに入力され、上記のような配列で各FFにラッチされている。

InvShiftRow変換時には、ShiftRow変換と同じシフト処理を1サイクルで行う。変換後は、上記のような配列で各FFにラッチされている。

3.3 MixColumn(InvMixColumn)変換部

本ブロックは、行列演算を行う。要素どうしの乗算はGF(2⁸)上の乗算である。(既約多項式 $M(x) = x^8 + x^4 + x^3 + x + 1$)

× c(x) ⇒

$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$
$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$
$a_{2,0}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$
$a_{3,0}$	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$

$b_{0,0}$	$b_{0,1}$	$b_{0,2}$	$b_{0,3}$
$b_{1,0}$	$b_{1,1}$	$b_{1,2}$	$b_{1,3}$
$b_{2,0}$	$b_{2,1}$	$b_{2,2}$	$b_{2,3}$
$b_{3,0}$	$b_{3,1}$	$b_{3,2}$	$b_{3,3}$

(1)暗号化(MixColumn変換)

$c(x) = '03'x^3 + '01'x^2 + '01'x + '02'$

$$\begin{pmatrix} b_{0,j} \\ b_{1,j} \\ b_{2,j} \\ b_{3,j} \end{pmatrix} = \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \begin{pmatrix} a_{0,j} \\ a_{1,j} \\ a_{2,j} \\ a_{3,j} \end{pmatrix}$$

(2)復号(InvMixColumn変換)

$c(x) = '0B'x^3 + '0D'x^2 + '09'x + '0E'$

$$\begin{pmatrix} b_{0,j} \\ b_{1,j} \\ b_{2,j} \\ b_{3,j} \end{pmatrix} = \begin{pmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{pmatrix} \begin{pmatrix} a_{0,j} \\ a_{1,j} \\ a_{2,j} \\ a_{3,j} \end{pmatrix}$$

暗号化／復号ともに、c(x)の係数はすべて4ビットなので、行列式の要素どうしの乗算は、8ビット×4ビット乗算器で演算可能である。乗算結果の加算はbitwise XORである。

MixColumn(InvMixColumn)変換部の基本構成ブロック図を図5に示す。本ブロックは、乗算器1-4と各乗算器の出力の排他的論理和を演算するXORで構成される演算器が、4つで構成される。ラウンド処理部の処理ビットnが32ビットの時、本ブロックは、図5の構成となるが、8ビット、16ビットの時は、それぞれ、乗算器が1個あるいは2個の構成となる。

次に乗算器について述べる。8ビット×4ビットの乗算器(図6)を用いて、暗号化および復号の時の乗数c(4ビット)を設定して、乗算を行うことが可能である。この乗算器は、我々が提案した2の拡大体上でのbit-parallel乗算器[14]である。図中、＋はXOR、×はANDである。

ここで、暗号化時の乗算処理と復号時の乗算処理に対して、2つの乗数を組み合わせることにより、暗号化と復号で回路の共通化が可能となる。

前記行列式の要素$b_{0,j}$に注目すると、暗号化は、

$b_{0,j} = 02*a_{0,j} + 03*a_{1,j} + 01*a_{2,j} + 01*a_{3,j}$

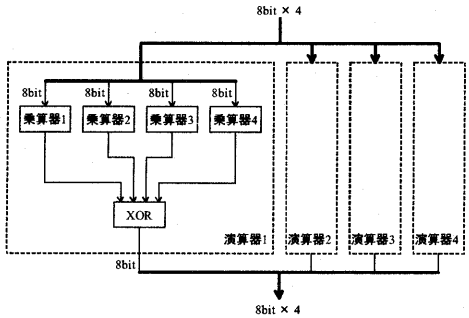

図5　MixColumn(InvMixColumn)変換部基本構成ブロック

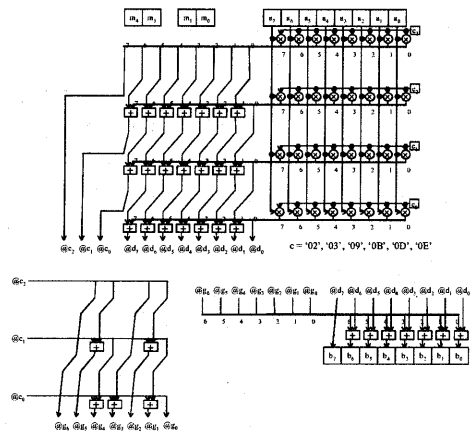

図6　8ビット×4ビット乗算器

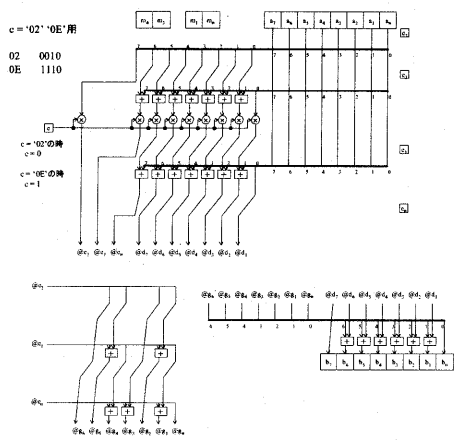

図7　乗算器1(2倍とE倍の共通回路)

であり、復号は、

$b_{0j} = 0E*a_{0j} + 0B*a_{1j} + 0D*a_{2j} + 09*a_{3j}$

である。

例えば、乗算器1を乗数2とE、乗算器2を乗数3とB、乗算器3を乗数1とD、乗算器4を乗数1と9の乗算を行う回路とすることにより、b_{1j}, b_{2j}, b_{3j},を演算する他の演算器についても同じ4つの乗算器で構成でき、かつ、暗号化と復号の演算が共通の回路で可能となる。乗算器1の回路図を図7に示す。2とEの2進表示は0010と1110である。下位2ビット(10)が共通かつ固定なので、XORやANDの削減が可能となる。また、上位2ビットが11と00なので、上位2ビットの部分積を下位2ビットの部分積とXORするか否かを切り換える構成とすることにより、さらにXORやANDの削減が可能である。他の乗算器2-4についても同様に回路規模の削減が可能となり、図6を用いた構成と比べて、演算器1個あたり300ゲートの削減となり、MixColumn(InvMixColumn)変換部全体では、1200ゲートの削減となる。

4　鍵スケジュール部

Rijndaelアルゴリズムでは、鍵長により、拡大鍵生成アルゴリズムが異なる[7][8]。また、暗号化と復号の処理では、使用するRound Keyの順番が逆になるので、拡大鍵の使用順序も逆になる。よって、鍵長毎に、暗号化用と復号用の鍵スケジュール回路が必要になる。

ここでは、回路規模削減を図るために、1つの回路で3種の鍵長に対応した拡大鍵生成が可能となる鍵スケジュール部基本回路構成を図8に示す。図中、FF0-7はそれぞれ32ビットのフリップフロップであり、拡大鍵レジスタとして使用する。また、鍵の32ビットワード数Nkは表3のとおり、鍵長によって異なる。ラウンド数Nrも表3のとおり、鍵長によって異なる。

鍵は、32ビット毎に分割(W_0〜W_{Nk-1})されてFF7に入力され、鍵長に関わらず、8サイクルかけて、FF0-7にセットされる。

表3　各鍵長でのNkとNrの値

	鍵		
	128bit	192bit	256bit
Nk	4	6	8
Nr	10	12	14

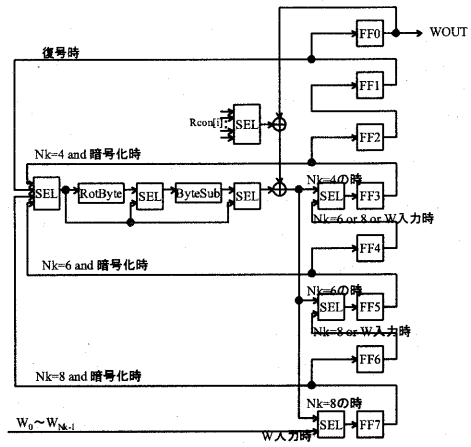

図8　鍵スケジュール部基本回路構成

鍵長に応じて、各セレクタが制御され、拡大鍵生成アルゴリズムに従って生成された 32 ビット拡大鍵は、WOUT から出力され、ラウンド処理部の Round Key 加算部 1 あるいは 2(図 1)で使用される。

また、復号の時は、図 8 の回路で一旦、暗号化時の最終ラウンドで使用する Round Key まで生成し、その鍵を元に、逆に戻して生成する。暗号化と復号での鍵スケジュールを同じ回路で実現することにより、回路規模を削減する。さらに、図 8 の ByteSub を図 1 のラウンド処理部内の ByteSub 変換部と共用することにより、回路規模を削減する。

図 8 では、鍵長 256 ビット対応であるが、128 ビット専用にすると、FF4-7 と一部セレクタが削減できる。

次に、鍵のセットアップについて述べる。暗号化の時は、鍵長に関わらず 8 サイクル要する。復号の時は、一旦、暗号化の最終段の Round Key を生成してから、拡大鍵レジスタにセットするため、8 + 4×(Nr+1) +8 サイクル要する。以上、鍵セットアップに要するサイクル数を表 4 に示す。

表 4　鍵セットアップ

		サイクル数
暗号化		8
復号	鍵長 128bit	8 + 4×11 + 8 = 60
	鍵長 128bit	8 + 4×13 + 8 = 68
	鍵長 128bit	8 + 4×15 + 8 = 76

5　検討

5.1　試作結果

目標諸元で示したモード処理・鍵長に対応し、3、4 章で述べた回路の設計、評価を行った。表 5 に、当社 0.35 μm CMOS ASIC[10]で評価した回路規模と処理速度を示す。これらの値は、表 1 の目標諸元をクリアしている。尚、処理速度は、クリティカルパスディレイ(11.27nsec)をワーストケースで評価した時の値である。また、本ハードウェアを FPGA で試作し動作確認を行った。使用した FPGA は、ALTERA 社の EP1K100QC208-3[9]であり、最大 32MHz で動作可能である。

本ハードウェアでの 1 ラウンド処理のサイクル数は 9 サイクルである。つまり、ByteSub (InvByteSub) 変換、および MixColumn (InvMixColumn)変換に必要なサイクル数はそれぞれ 4 サイクルであり、ShiftRow 変換は 128 ビットのデータに対して 1 サイクルでシフト処理を行う。したがって、最初の Round Key 加算に 4 サイクルかかるから、各鍵長に対して、4 + 9× Nr サイクルで Rijndael の暗号化もしくは復号が完了する。鍵長 128 ビット、192 ビット、256 ビットでの処理サイクル数は、94、112、130 となる。

5.2　処理単位

回路規模の削減と処理速度の効率を見るために、ラウンド処理部の処理ビット n が 8 ビットの時と 32 ビットの時の回路規模を概算見積りした。n=32 に対して、n=8 では 15%(約 3K ゲート)の削減であった。一方、処理速度は 3.5 倍となり、効率的な処理の面から、32 ビット処理が適していると判断し、これを採用した。

5.3　回路規模

本試作ハードウェアでは、実用性を考えて、CBC モード処理回路および CPU インタフェースを含む周辺回路も実装されている。これらを含まない Rijndael アルゴリズムのコア処理部は、13.6K ゲートである。さらに、鍵長 128 ビット専用にした場合のコア処理部は、11.4K ゲートになった。

表5　性能評価

回路規模	17.6K ゲート	
	ラウンド処理部	8K ゲート
	鍵スケジュール部	5.6K ゲート
	その他(CBC モード処理など)	4K ゲート
処理速度	73Mbps (鍵長 128bit) 61Mbps (鍵長 192bit) 53Mbps (鍵長 256bit)	

5.4　従来の試作結果との比較

従来の小型実装として、Rijndael アルゴリズムの 1 段分のラウンド処理を実装評価した結果、回路規模 33.9K ゲート、鍵長 128 ビット時の処理速度 510Mbps であることが報告されている[2]。本ハードウェアは、回路規模が 33%に削減されたのに対し、処理速度は 14%となっていることがわかる。本ハードウェアでは、速度要件が比較的緩やかであったため、高速化のための検討を行っていない。しかし、すでに様々な高速化手法[11][12][13][15]が提案されおり、これらの手法を活用すれば、回路規模を増加することなく、高速処理が実現可能であると予想される。

6　まとめ

スマートカードに搭載可能な回路規模で Rijndael アルゴリズムを実行する回路構成を示し、FPGA 試作評価した。ラウンド処理を 32 ビット処理のループ構成とし、ラウンド処理部の各変換部および鍵スケジュール部において、暗号化と復号で共通化した回路構成とすることにより、回路規模の削減を図った。その結果、0.35 μ m CMOS ASIC で実装した場合、17.6K ゲート(CBC モード対応)となり、これは、スマートカードに搭載可能なサイズである。

尚、上記 17.6K ゲートのうち Rijndael アルゴリズムのコア処理部は、13.6K ゲートであり、鍵長 128 ビット専用にした場合、11.4K ゲートとなった。

参考文献

[1]市川, 粕谷, 松井, "128bit ブロック暗号のハードウェア実装について(II)", SCIS2000-D40, Jan. 2000.

[2]市川, 時田, 松井, "128bit ブロック暗号のハードウェア実装について(III)", SCIS2001, pp.669-674, Jan. 2001.

[3]A. Dandalis, V. K. Prasanna, and J. D. P. Rolim, "A comparative study of performance of AES final candidates using FPGAs", CHES 2000, Aug. 2000.

[4]A. J. Elbirt, W. Yip, B. Chetwynd, and C. Paar, "An FPGA Implementation and Performance Evaluation of the AES Block Cipher Candidate Algorithm Finalists", AES3, Apr. 2000.

[5]N. Weaver and J. Wawrzynek, "A Comparison of the AES Candidates Amenability to FPGA Implementation", AES3, Apr. 2000.

[6]K. Gaj and P. Chodowiec, "Comparison of the hardware performance of the AES candidates using reconfigurable hardware", AES3, Apr. 2000.

[7]J. Daemen and V. Rijmen, "AES Proposal: Rijndael"
http://csrc.nist.gov/encryption/aes/rijndael/Rijndael.pdf

[8]NIST, "Specification for the AES", Draft FIPS for the AES
http://csrc.nist.gov/publications/drafts/dfips-AES.pdf

[9]Altera, "ACEX 1K Programmable Logic Device Family Data Sheet", ver. 2.0, Mar. 2001.
http://www.altera.com/literature/ds/acex.pdf

[10]富士通半導体デバイスデータシート セミカスタム CMOS スタンダードセル CS66 シリーズ.
http://edevice.fujitsu.com/fj/DATASHEET/j-ds/j620205.pdf

[11]H. Kuo and I. Verbauwhede, "Architectural optimization for a 1.82Gbits/sec VLSI implementation of the AES Rijndael algorithm", CHES 2001, May. 2001.

[12]M. McLoone and J. V. McCanny, "High performance single-chip FPGA Rijndael algorithm implementations", CHES 2001, May. 2001.

[13]V. Fischer and M. Drutarovsky, "Two methods of Rijndael implementation in reconfigurable Hardware", CHES 2001, May. 2001.

[14]S. Okada, N. Torii, K. Itoh, and M. Takenaka, "Implementation of elliptic curve cryptographic coprocessor over GF(2^m) on an FPGA", CHES 2000, Aug. 2000.

[15]V. Rijmen, "Efficient implementation of the Rijndael S-box"
http://www.esat.kuleuven.ac.be/~rijmen/rijndael/sbox.pdf