ハードウェアセキュリティ組込み型スレッドレベル同時処理マルチメディアモバイルプロセッサの開発

佐藤 陽一，佐藤 友暁，深瀬 政秋

Abstract
With respect to privacy on Internet, it is a matter of keen interest to provision security for multimedia mobile communication. This is characterized by the power conscious high speed cryptography of large quantity data. Such demand should be covered by a single VLSI processor system. We describe in this article a hardware security-embedded multimedia mobile processor named HSgorilla (hardware security-embedded gorilla) by sophisticatedly unifying five up-to-date processor techniques. They are single processor SMT (simultaneous multithreading), VLIW (very long instruction word), Java, and random addressing. RAC (random addressing cryptography) achieved by HSgorilla is very promising compared with regular cryptographic operations in view of running time and secure strength. Carefully considering features of image cryptography by RAC, it is very promising technique in the era of ubiquitous computing.

1．はじめに

近年のネットワーク事情は，ブロードバンド（高速大容量回線）の急速な普及により利用者の増加，ネットワーク環境の急速な発展をもたらした．更には，無線 LAN などのモバイルアクセス環境の進化（Ohtsuka, 1997）（Sato, 1997）携帯電話の普及，高機能化，情報家電の登場などによりユビキタス・コンピューティングが現実のものとして近づいてきた．このことにより，現代社会に飛躍的な利便性が持たされている反面，プライバシーの欠如や侵害という深刻，且つ重大な問題を引き起こすことになる（Saha, 2003）．ネットワークインフラの普及した現代社会では個人レベルでネットワーク環境の利用が容易なものとなり，それに伴いプライバシー保護の対策も個人レベルでいつでもどこでも行うことを可能にすることが不可欠となる．そのために，マルチメディアモバイルコンピューティングのさらなる発展とセキュリティ機能のハードウェア化が必要不可欠である．

現在普及している公開鍵暗号 RSA（Rivest-Shamir-Adelman）（Rivest, 1978）は，高い信頼性を持つが多大な処理時間を必要とする欠点を持つ．また，ECC（Elliptic

Sato Yoichi 弘前大学理工学部
Sato Tomoaki 札幌学院大学社会情報学部
Fukase Masa-aki 弘前大学理工学部

Curve Cryptography)（Koblitz, 1987）を利用可能な専用プロセッサの開発は，RSA より強い暗号強度を持つことから有望であるが，多くのソフトウェアもしくはハードウェアの計算リソースを必要とする欠点を持つ．公開鍵では，データ量に比例して処理時間が増加することから，大容量のマルチメディアデータに対応しうる公開鍵方式が要求される．従って，以上の要求を満たす VLSI プロセッサの開発は有意義であると考えられる．

我々はこれまでに，ユビキタス・コンピューティング向けに，小型化且つ省電力で高速処理可能なプロセッサの実現を目指し，マルチメディアモバイルプロセッサアーキテクチャ gorilla を開発してきた（Mikuni, 2003）（三国，2004）（Fukase, 2004）（Fukase, 2004a）（Fukase, 2004b）（Fukase, 2004c）．以下に gorilla アーキテクチャの特徴を示す．

1）インタプリタ型の Java CPU を核とする
2）マルチメディアに対処するために SMT（Simultaneous Multithreading）の導入
3）個々のスレッドブランチに対して VLIW（Very Long Instruction Word）方式と PicoJava のマイクロプログラミングの技術の融合

また，我々は別に RAP（Random Addressing-accelerated Processor）を開発してきた（深瀬，2002）（深瀬，2003）（尾山，2001）（Z, Liu, 1999）（深瀬，2004）．RAP は，内蔵されたハードウェア乱数発生器 RNG（Random Number Generator）とデータキャッシュの直結を特徴とする．この直接接続は本論文で提唱する暗号システム RAC（Random Addressing Cryptography）の原理となる構造である．

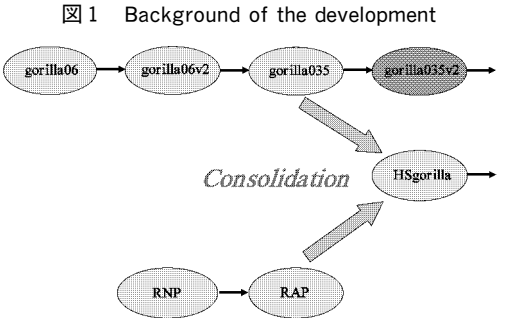
今回我々は，モバイル用途に特化したマルチメディアセキュリティの為，以上に述べた gorilla と RAP の両機能を統合し，ハードウェアセキュリティ強化型マルチメディアモバイルプロセッサアーキテクチャ HSgorilla（Hardware Security-embedded gorilla）を開発した．更に，これを実装するために，HSgorilla035 と名づけたプロセッサを設計し，0.35 μm CMOS スタンダードセル方式で試作チップを開発する．

この論文では，HSgorilla のアーキテクチャとその実装，暗号システムについて順に述べる．第 2 章では，マルチメディアモバイル用プロセッサアーキテクチャと独自ハードウェア暗号システム RAC を統合し，構築したプロセッサアーキテクチャ HSgorilla について述べる．第 3 章では，アーキテクチャ HSgorilla の実装について述べる．第 4 章では，ハードウェア暗号システム RAC について述べる．最終章はまとめである．

2．Architecture HSgorilla

本章では，本研究で構築されたランダムアドレッシング機能強化型マルチメディアプロセッサアーキテクチャ HSgorilla について取りまとめる．プロセッサアーキテクチャ HSgorilla は，我々の研究室で既に開発された，マルチメディアモバイルプロセッサ用アーキテクチャ gorilla と，ランダムアドレッシング機能を強化することで独自ハードウェア暗号システム RAC を確立したプロセッサアーキテクチャ RAP の 2 つのタイプのプロセッサアーキテクチャを統合することによって開発を行う．HSgorilla に関する開発経緯

図 1 Background of the development



を図1に示す.

HSgorillaは，マルチメディアモバイルプロセッサ用アーキテクチャgorillaを基に構成されている．このため，マルチメディアに対処するためにJava対応とし，SMTとVLIW方式を合わせ持つ．更に，ここでは，マルチメディア暗号の実現のためにランダムアドレッシングの核となるハードウェア化乱数発生器RNGを搭載し，ハードウェア暗号システムRACを実現している．表1にアーキテクチャHSgorillaと同時に，基となる2つのアーキテクチャgorillaとRAPの特徴を示す.

2.1 Hardware Organization

図2はHSgorillaの全体構造を示したものである．基本的には，2つのスレッドブランチから成り，更に，それぞれのスレッドブランチには独立実行可能な演算ユニットが2つずつ配されている．この他に，各スレッドブランチから共有されるユニットFIFO（First In First Out），RNG，Data cacheが配され，構成されている．また，このHSgorillaは，総数6ステージのパイプライン機構を持つ．それぞれのステージは，Instruction Fetch Stage，Decode/Operand Fetch Stage，Operand Access Stage，Execute1 Stage，Execute2 Stage，Write Back/Memory Access Stageとなっている．図3にパイプライン機構と，これをデュアルモードで動作させた場合のスレッドブランチ動作モデルをまとめたものを示す.

表2には，TLP（Thread Level Parallelism），ILP（Instruction Level Parallelism）とHSgorilla全体の並列度をまとめたものを示す．並列度（degree）は並列に実行される命令の数である．HSgorillaの最大並列モードであるデュアルモードは，SMTとVLIW（Very Long Instruction Word）-likeの同時使用時である．よって，デュアルモードはHSgorillaの場合，4 IPC（Instructions/Clock），12 Java Byte Codes/Clockを実現する.

表2 Parallelism of HSgorilla

Parallel mode	TLP (degree)	ILP (degree)	Total parallelism (degree)
Dual	2	2	4
SMT		1	2
VLIW-like	1	2	
Serial		1	1

表1 Architectures related to HSgorilla

Architecture	SMT	VLIW-like	Pipelining		Hardware security	Core derivative	Process	Clock speed	Outcome
			Regular	Waved					
gorilla	2-degree	Inactive	4-degree microinstruction level	Not available	Not available	gorilla06	0.6-μm	170MHz	Chip
			8-degree instruction level			gorilla06v2	0.6-μm	400MHz	Synthesis
		2-degree	7-degree instruction level	2-degree waved-execute unit		gorilla035	0.35-μm	240MHz	Chip
						gorilla035v2			Synthesis
RAP	Not available		5-degree	Not available	Embedded		FPGA	45MHz	FPGA
HSgorilla	2-degree	2-degree	6-degree instruction level	2-degree waved-execute unit		HSgorilla0.35	0.35-μm	150MHz	Chip

図2 Organization of HSgorilla

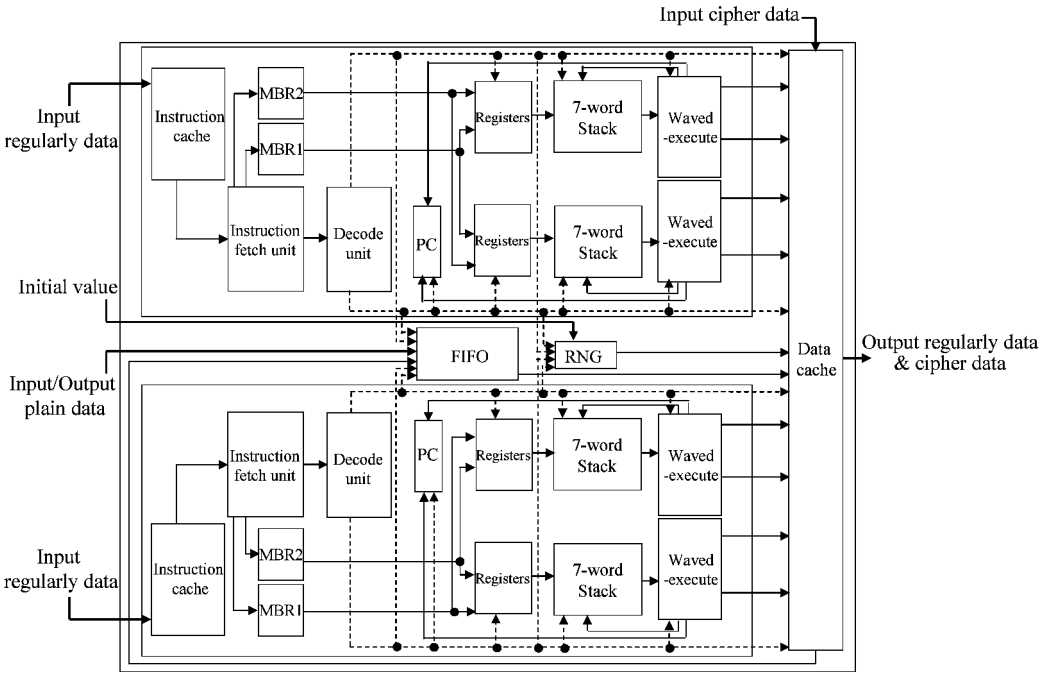
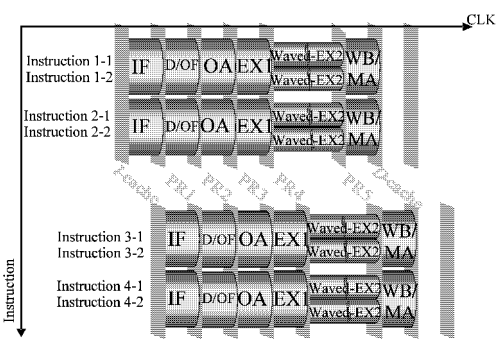


図3 Dual mode execution by HSgorilla



2.2 ISA (Instruction Set Architecture)

表3にHSgorillaのISAを示す．HSgorillaは，基本的にアーキテクチャgorillaで確立されたRISC型Java ISAで構成されている．そして，最も特徴的な命令は，RACの命令となるrsw（Random number-addressing Store Word），rlw（Random number-addressing Load Word）である．命令rswは，暗号化の対象となるデータに対し，RNGで発生させた乱数を同期させ，その乱数をデータキャッシュへのストア・アドレスとして使用することによって，RACの暗号化となるランダム・ストアを行う．

命令rlwは，暗号化されたデータにRNGで発生させた乱数を同期させ，その乱数をデータキャッシュからのロード・アドレスとして使用することによって，RACの復号化となるランダム・ロードを行う．これらのランダム・ストア，ランダム・ロードは，総称してランダムアドレッシングと呼ぶ．このランダムアドレッシングについて，詳しくは後述する．

3．Implementation

本章では，HSgorilla035のプロセスルール0.35−μm CMOSへの実装について取りまとめる．表4にHSgorilla035開発環境を示す．設計開発は，東京大学大規模集積システム設計教育研究センターを通し，シノプシス株式会社，ケイデンス株式会社，ローム株式会社および凸版印刷株式会社の協力で行っ

表3　Instruction of HSgorilla

Opcode				Operand (bytes)	Latency (clocks)
category	byte code	Mnemonic	動作		
Java	0x00	nop	何もしない	0	7
	0x10	bipush	byte で表される数値を int 型に符号拡張したものをオペランドスタックへプッシュする.	1	
	0x15	iload	index で指定したローカル変数をオペランドスタックへプッシュする.	2	
	0x36	istore	オペランドスタックより数値をポップし，それを index で指定したローカル変数へ代入する.	2	
	0x60	iadd	int の加算	0	
	0x65	isub	int の減算	0	
	0x75	ineg	int の符号反転	0	
	0x81	ior	int の or（論理和）	0	
	0x7f	iand	int の and（論理積）	0	
	0x83	ixor	int の xor（排他的論理和）	0	
	0x78	ishl	int の左シフト	0	
	0x7a	ishr	int の算術右シフト	0	
	0xa7	goto	PC＋branch へ分岐する.	2	
	0x99	ifeq	value＝ 0 なら PC＋branch へ分岐	2	
	0x9a	ifne	value≠ 0 なら PC＋branch へ分岐	2	
	0xa1	if_icmplt	value 1 ＜value 2 なら PC＋branch へ分岐	2	
RAC	0x0c	rlw	Load the content of RNG－addressed D cache into FIFO	0	4
	0x0d	rsw	Store FIFO to RNG－addressed D cache	0	5

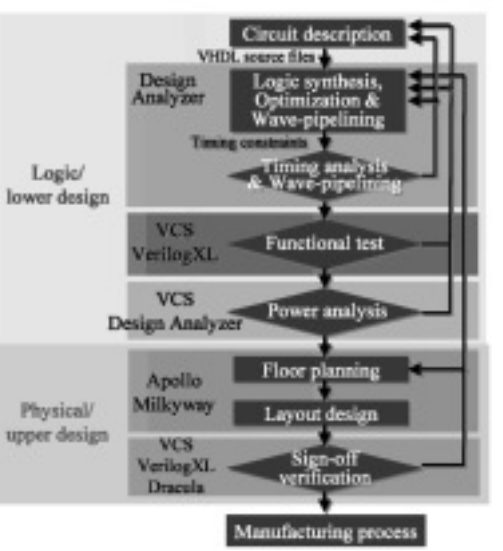
表4　Instruction of HSgorilla

Hard ware	
CPU	Ultra Spark II 750MHz
Main memory	1024MB
Swap memory	1787MB

Soft ware	
OS	Solaris8
Synthesis tool	Synopsys-Dsign Analyzer ver. 2001. 08-SP2
Simulation tool	Synopsys-VCS ver. 7.0 Synopsys-VirSim ver. 4.3. R2
Layout tool	Synopsys-Apollo version 2000.2.3.4.0.9 Synopsys-Milkyway version 2000.2.3.4.0.9

Language	
Synthesis	VHDL
Simulation	Verilog-HDL

図4　Design environment and steps



た．設計方式は開発期間の短縮，設計の柔軟性，トランジスタの集積度の向上を実現するスタンダードセル方式を採用した．

図 4 に HSgorilla035 の開発フローを示す．アーキテクチャHSgorilla を搭載するための回路仕様記述は，読解性に優れ，高い記述能力を持つ VHDL（Very High Speed IC

HDL）を用いた．論理合成ではASICベンダーのテクノロジーを指定し，目標とする回路性能を設計制約条件として与え，論理回路の自動生成を行なう．機能検証，サインオフ検証は記述が容易で，ゲートレベルシミュレーションの機能が充実しているVerilog-HDLを用い行なう．

物理設計はフロアプランにおいて，LSIチップ上で回路の配置の大枠を決定し，論理設計で生成したセル同士の結線情報であるネットリストを用いて，自動配置配線を行なう．レイアウトした実際の配線はサインオフ検証環境に移され，最終的なタイミングを確認する．その後，レイアウトの終了した最終的なネットリストと，物理設計において得られた実配線情報を基にLSI製造後の動作保証を行なう為，サインオフ検証を行なう．これらを経て，実際にレイアウトを行ったHSgorilla035のチップレイアウト図を図5に示す．

4．RAC

本章では，HSgorillaでのランダムアドレッシングを用いた暗号方式RACの原理について述べる．HSgorillaでのRACは，マルチ処理に対応させるために，その暗号化・復号化は共にハードウェアレベルでの特殊な動作となる．このRACを用いた暗号システムの提示，そして，HSgorillaを用いたRACのシミュレーション結果をそれぞれ示す．

4.1 RACの原理

RACの基本原理は，ランダムアドレッシングによる転置暗号化である．これは，古典的なシーザー暗号などのように，配置の変え方に規則性があるものではなく，RNGで発生させた乱数と対象となるデータを同期させてデータキャッシュへランダムにアクセスを行うものである．HSgorillaでは，これをRNGとデータキャッシュを直結させることによっ

図5 Chip layout of HSgorilla035

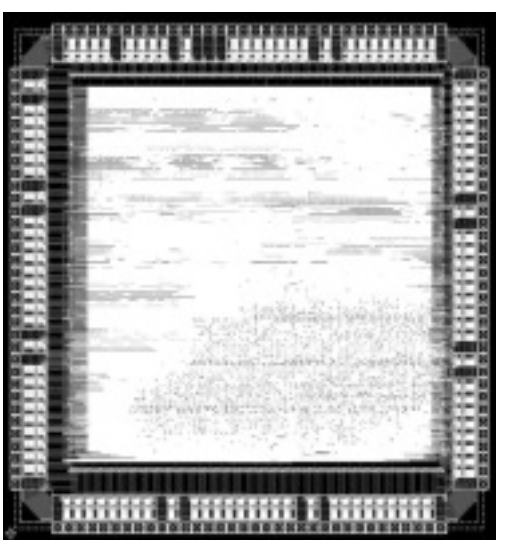


図6 Principle of RAC

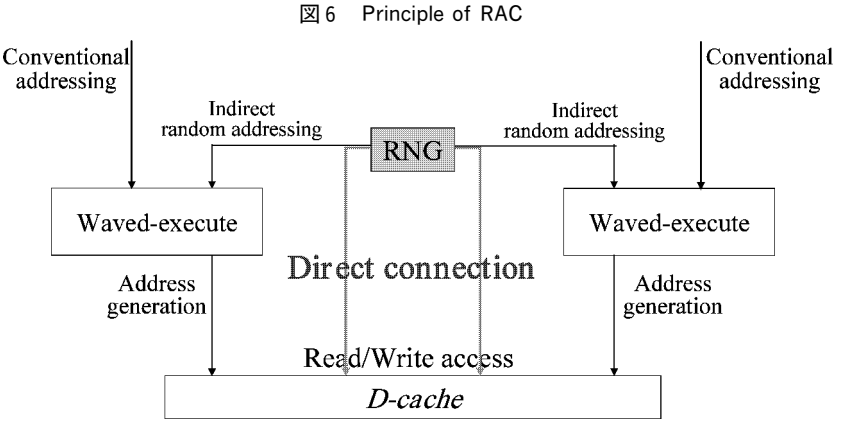
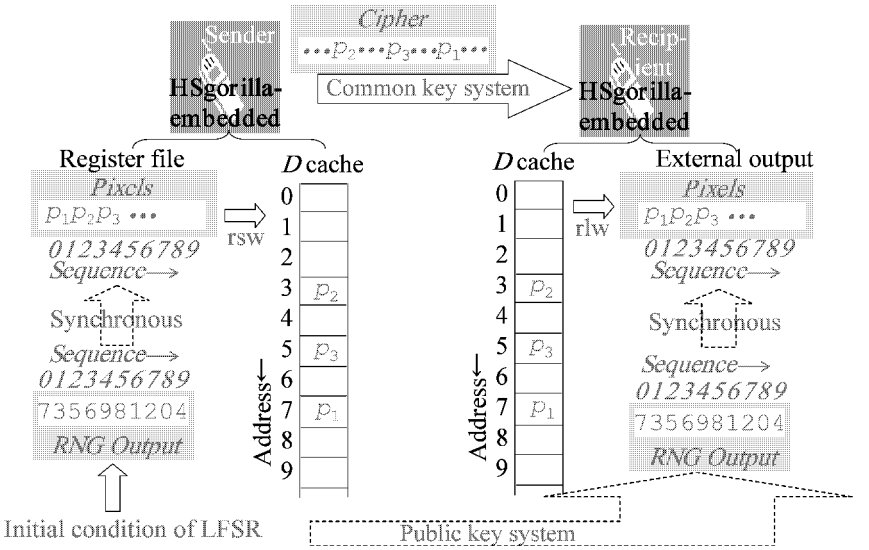


図7　Image cryptography system



て実現している．HSgorillaの各スレッドブランチでのランダムアドレッシングの原理を図6に示す．

図7ではイメージデータに対するRACシステムを示す．ここで，p_iは$_i$番目のピクセルデータを示す．このとき，〝p_1，p_2，p_3，…〟と〝…，p_2，…，p_3，p_1，…〟はそれぞれ入力イメージと暗号化されたイメージを指す．〝7356981204〟は対応する乱数で，そのデータ長は入力イメージと等しい．

RACシステムでは，送信側において，命令rswによるランダム・ストアを行うことによって，イメージを暗号化し書き込む事が可能である．受信側も同様に命令rlwによるランダム・ロードを行うことによって復号化が可能である．転置暗号方式で，対称暗号方式であるRACでは，送信側，受信側共に，他のソフトウェア暗号システムなどにみられる複雑なアルゴリズムは必要としない．

RACは，送信側と受信側で同一の乱数を使用することによって暗号処理と復号処理が対称に行えるものである．従って，RNGで発生させる乱数は送受信双方で同一でなければならない．HSgorillaに組み込まれているRNGは，LFSR（Linear Feedback Shift Register）という構造を用いている．この構造は，同一の初期値を与えることによって同一の乱数を発生させることができる．HSgorillaでの暗号システムでは，ユーザがこの初期値を任意に指定できる．この初期値を双方のユーザがキーとして共有することによって，RAC暗号システムが成り立つ．その情報は公開鍵方式を用い共有する．暗号化を成されたデータはRNGの初期値とは別経路で，公開鍵方式を用いて転送を行う．

4.2　RACの実用化

実際にハードウェア暗号システムRACを組み込んだHSgorillaによる暗号化・復号化の際の動作シミュレーションビューを図8,9に示す．それぞれ①～③は，時系列を示している．

まず，暗号化（送信側）では，①の入力で対象となるデータを入力する．②の暗号化で，クロックの立ち上がり時にFIFOから出力されたデータ（RAC_DATA 11～RAC_DATA 22）とRNGからの乱数（RNG 11～RNG 22）とを同期させてデータキャッシュへ

図8 Encryption simulation

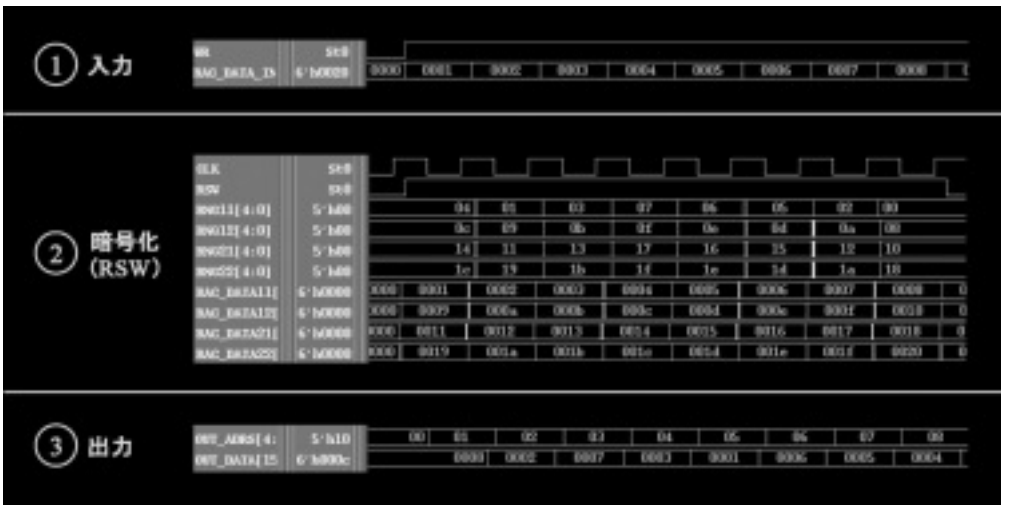
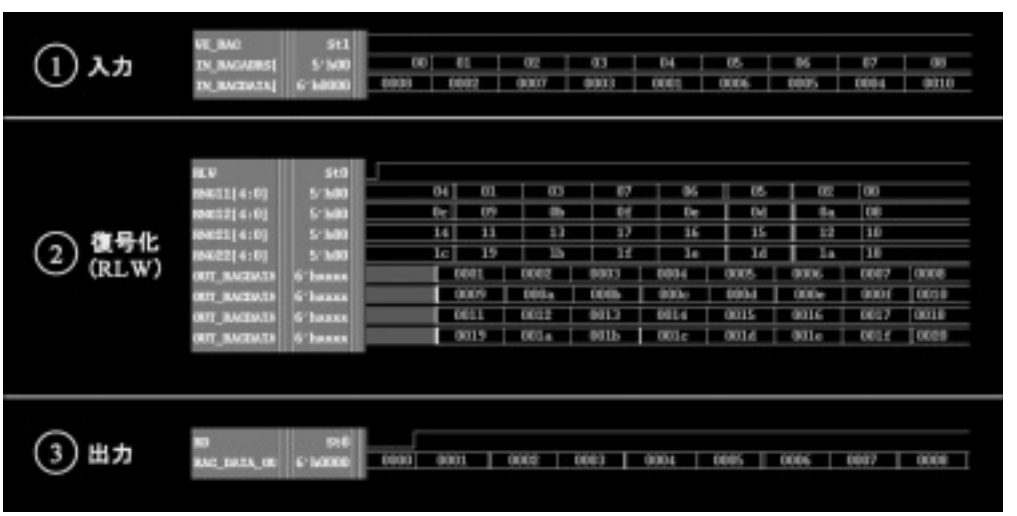


図9 Decryption simulation



ランダム・ストアを行う．③の出力では，データキャッシュに格納されたデータを 0 番地から順に読み出している．出力の通り暗号化が成されている．

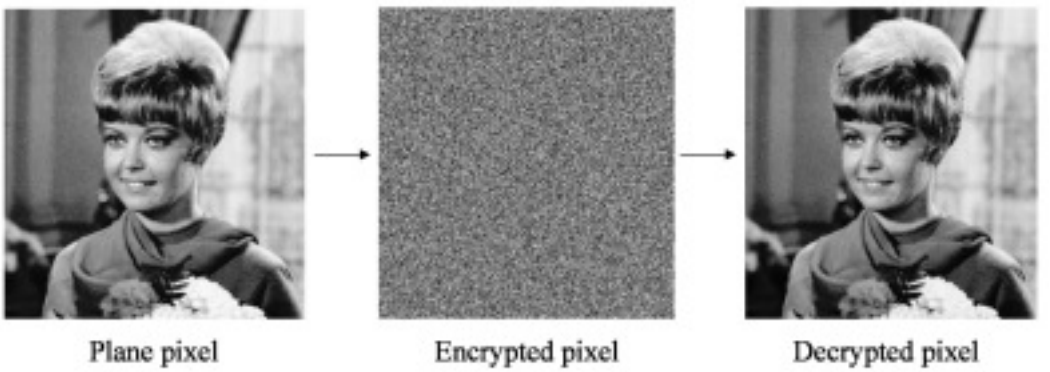
復号化（受信側）は，①の入力で暗号化されたデータを入力する．②の復号化で，RNG からの乱数（RNG 11～RNG 22）をアドレスとして使用し，データキャッシュからランダム・ロードを行い，その出力を FIFO へ格納する．③の出力では，FIFO に格納されたデータを順に読み出している．出力の通り復号化が成されている．

図 10 には，実際にソフトウェア上で C 言語を用い，RAC 暗号システムを構築し，対象となるピクセルデータに対して暗号化・復号化の処理を施したものを示す．

4.3 RAC の比較・考察

表 5 に RAC と他の暗号システムの特徴をまとめたものを示す．RAC の最も特徴的なポ

図10　Encryption and decryption for pixel data by RAC



イントは，対象データをブロックに区切ることなく，全データを暗号化することである．キー長は，サイクルがビット幅により指数関数的に上昇するLFSRによって容易に拡張でき，大容量データに対応可能である．また，RACはXOR方式ではなく，無作為の転置であること，対称暗号方式であるということが特徴として挙げられる．

図11は，イメージの暗号化にかかるクロック数を基に，処理時間の観点からRACの優位性を示したものである．RACの比較対象と

図11　Running time of RAC VS.　Standard XOR

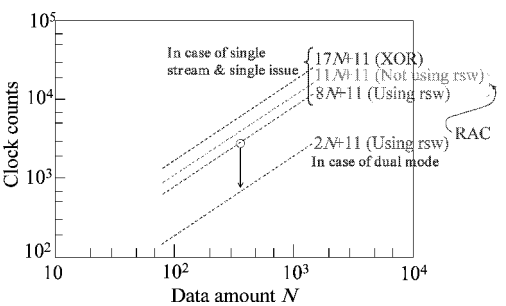


表5　Cryptographic features of RAC VS.　Others

	Cipher	Division of a plaintext encrypted at a time	Cryptographic means	Encryption operation	Remarks
Modern	Block	Large fixed length, normally 64/128 bit, exceptionally 1024 bits	Key≦Division length	Bitwise XOR	Established de fact standard
	Stream	Smaller variable length, a few bits of character			Not yet de fact standard
	DES	Same as block cipher		XOR, scramble, shift, etc.	Established de fact standard
Classical	Caesar cipher	Rather short full text	Key of shift counts	Alphabet shift	
	Substitution			Alphabet exchange	
	Transposition		Short-length key/diagrams	Random exchange/ scramble	Rarely encountered
	RAC	Extremely long full text	Key of full-length random numbers		One-time pad with ideal cipher strength

して，DES などの共通鍵を用いた対称暗号方式で使われる XOR 処理を取り上げている．Y 軸は実行時間で，クロックサイクル時間にクロック数を掛けることで測定する．X 軸は，対象となるデータ量を表す．グラフから読み取れるように，XOR 方式の暗号化に比べ，本論文で提唱した RAC は高速である．

5．おわりに

本論文では，プロセッサアーキテクチャ gorilla と RAP を統合することによって，ハードウェア暗号システム組込み型マルチメディアプロセッサアーキテクチャ HSgorilla の構築を行った．わずかなハードウェア量の増加で，より高いスループット，省電力化へ貢献しうる，マルチメディア暗号に有効なアーキテクチャを実現した．また，我々は，このアーキテクチャ HSgorilla をプロセッサ HSgroilla035 として実装を行った．

今後，チップ化される HSgorilla035 は，そのチップに対して処理速度評価，暗号強度評価を行うことによって，その優位性を示す．この 2 つの評価は現在主流となっている RSA や ECC などのソフトウェアによる暗号システムとの比較を行い，本研究で開発したプロセッサによるハードウェア暗号システムが処理速度，暗号強度の両面で優れていることを示す．また，それをもとに，実用化のための更なる改良へと段階を進める．

参考文献

Rivest, R. L., Shamir, A., and Adleman, L. M., (1978) A Method for Obtaining Digital Signatures and Public-Key Cryptosystems, *Communications of the ACM*, v. 21, n. 2, pp. 120-126.

Koblitz, N., (1987) Elliptic Curve Cryptosystems, *Mathematics of Computation*, v. 48, n. 177, pp. 203-209.

Ohtsuka., M. Sato., T. Rashid, E. Fukase, M., and Nakamura, T (1997) Digital Broadcasting for Multimedia System, *Proc. of International Wireless and Telecomm. Sympo*, Vol. 3, 107-110

Sato., T. Rashid, E. Hamada, K. Fukase, M., and Nakamura, T. (1997) Performance Analysis of the Wireless Hypermedia System, *Proc. of ICPWC'97*, 293-296

Liu, Z. and Fukase, M. (1999) An Application of a Random Number Generator，情報処理学会東北支部研究会

尾山武志（2001）「Random number-addressing processor の高性能化に関する研究」弘前大学大学院理学研究科修士論文

深瀬政秋・尾山武志・劉哲（2002）「ランダムサンプリングの高速化 ── 機能強化プロセッサの設計と試作 ──」『信学技報』Vol. 102，No. 272（SDM 2002-154，ICD 2002-65），7-12

Fukase, M. Khondkar, P., and Nakamura, T. (2002) Designing a Low Power and High Performance Multithreaded Java Microprocessor, *Proc. of 10th NASA Symposium on VLSI Design, Albuquerque*, 10. 4. 1-10. 4. 7

今井礼大(2002)「マルチメディアプロセッサに関する研究.」弘前大学理工学研究科修士論文

Saha, D. and Mukherjee, A. (2003) Pervasive Computing: A Paradigm for the 21st Century, *Computer Magazine*, Vol. 36, No. 3, 25-31

Mikuni, K. Nakamura, Y. and Fukase, M. (2003) Architectural Aspects of Multimedia Mobile Processor Cores，平成 15 年度電気関係学会東北支部連合大会

Nakamura, Y. Mikuni, K., and Fukase, M. (2003) Design Process of a Multimedia Mobile Processor Core，平成 15 年度電気関係学会東北支部連合大会

三国勝志・中村吉樹・今井礼大・深瀬政秋・佐藤友暁(2003)「モバイルコンピューティング用マ

ルチメディアプロセッサの実装」, *FIT2003*

深瀬朝子・佐藤陽一・佐藤友暁・深瀬政秋・荒木喬(2003)「ランダムアドレッシング機能強化型マルチメディアプロセッサによる暗号システム」, *FIT2003*

三国勝志 (2004)「マルチメディアモバイルプロセッサの開発に関する研究」弘前大学理工学研究科修士論文

深瀬朝子(2004)「特定用途 VLSI プロセッサに関する研究」, 弘前大学理工学研究科修士論文

Fukase, M. Fukase, A. Sato, Y., and Sato, T. (2004) Exploiting a Hardware Security-Embedded Multimedia Mobile Processor System and its Application, *Proc. of ITC-CSCC2004*, 7C3L-3-1-7C3L-3-4

Fukase, M. Fukase, A. Sato, Y., and Sato, T. (2004a) Cryptographic System by a Random Addressing-Accelerated Multimedia Mobile Processor, *Proc. of 8th SCI2004*, Vol. II, 174- 179

Fukase, M. Nakamura, Y. Akaoka, R. and Sato, T. (2004b) Development of a Multimedia Mobile Processor, *Proc. of ISCIT2004*, 66

Fukase, M. Sato, Y., and Sato, T. (2004c) Design of a Hardware Security-Embedded Multimedia Mobile Processor, *Proc. of ISCIT2004*, 40

謝辞

筆者の一人である佐藤陽一の母校である札幌学院大学, 並びに札幌学院大学の先生方の本研究に至るまでの有益な御指導に対し深く感謝致します. 本論文の執筆にあたり, 御支援を頂いた, 弘前大学理工学研究科深瀬研究室佐久間玲奈様に厚く御礼申し上げます.

本研究は, 東京大学大規模集積システム設計教育研究センターの終始懇切な御指導, 並びに東京大学大規模集積システム設計教育研究センターを通し, シノプシス株式会社, ケイデンス株式会社, ローム株式会社および凸版印刷株式会社の協力を受け行いました. ここに深く感謝致します.