Low-Power Design of Hardware One-Time Password Generators for Card-Type OTPs

Sung-Jae Lee, Jae Seong Lee, Mun-Kyu Lee, Sang Jin Lee, Doo-Ho Choi, and Dong Kyue Kim

Since card-type one-time password (OTP) generators became available, power and area consumption has been one of the main issues of hardware OTPs. Because relatively smaller batteries and smaller chip areas are available for this type of OTP compared to existing token-type OTPs, it is necessary to implement power-efficient and compact dedicated OTP hardware modules. In this paper, we design and implement a low-power small-area hardware OTP generator based on the Advanced Encryption Standard (AES). First, we implement a prototype AES hardware module using a 350 nm process to verify the effectiveness of our optimization techniques for the SubBytes transform and data storage. Next, we apply the optimized AES to a real-world OTP hardware module which is implemented using a 180 nm process. Our experimental results show the power consumption of our OTP module using the new AES implementation is only 49.4% and 15.0% of those of an HOTP and software-based OTP, respectively.

Keywords: One-time password, AES, HMAC, card-type OTP, low-power hardware implementation.

I. Introduction

User authentication is a procedure in which a user attempts to prove his or her identity to gain access to a system. It is generally accepted that there are three classes of authentication mechanisms: authentication by something the user knows, such as passwords or personal identification numbers; authentication by something the user has, such as physical tokens; and biometric verification using either physiological or behavioral characteristics of the user. For applications that require a high level of security, combinations of more than two factors out of the above three are frequently used. For example, in Internet banking applications, an ID and its corresponding password is used as the first authentication factor, and at the same time, either a security card or a one-time password (OTP) device is used as the second authentication factor.

A security card is a portable codebook that contains a short list of numbers. When the owner of this card is given a challenge index, he or she refers to the list to find an appropriate number corresponding to the given index; then, he or she can enter this number to the system for authentication. Although this mechanism has long been used for the Internet banking, especially in Korea, it is vulnerable to attacks in which an attacker accumulates the challenges and responses because the length of the list written in the security card is not sufficiently long.

The OTP is an effective countermeasure to this problem. Because an OTP user can use a new temporary key for every session and this key is generated by a pseudorandom number generator with sufficient entropy, it is difficult for an attacker to analyze the key without possessing the OTP device. However, the problem of a hardware OTP is that it requires a separate hardware device known as an OTP token, so the inconvenience

Manuscript received July 5, 2010; revised Dec. 28, 2010; accepted Apr. 7, 2011.

This work was sponsored by ETRI System Semiconductor Industry Promotion Center, Human Resource Development Project for SoC Convergence.

Sung-Jae Lee (phone: +82 10 7136 7456, email: sjlee@kisa.or.kr) is with Korea Internet & Security Agency, Seoul, Rep. of Korea.

Jae Seong Lee (email: jslee@esslab.hanyang.ac.kr) and Dong Kyue Kim (corresponding author, email: DQKIM@hanyang.ac.kr) are with the School of Electrical and Computer Engineering, Hanyang University, Seoul, Rep. of Korea.

Mun-Kyu Lee (email: mklee@inha.ac.kr) is with the School of Computer and Information Engineering, Inha University, Incheon, Rep. of Korea.

Sang Jin Lee (email: sangjin@korea.ac.kr) is with the School of Information Management and Security, Korea University, Seoul, Rep. of Korea.

Doo-Ho Choi (email: dhchoi@etri.re.kr) is with the Software Research Laboratory, ETRI, Daejeon, Rep. of Korea.

doi:10.4218/etrij.11.0110.0392

of carrying separate authentication devices has restricted the chance of popularizing hardware OTPs.

As a result, recently-proposed card-type OTPs provide promising solutions to this problem because they have higher mobility than existing token-type OTPs. It is easy to carry these OTP types within a wallet, and it is even possible to provide this as the form of a combination commodity with credit cards or smart cards; therefore, these are expected to replace the existing hardware token-type OTPs and contribute a large share to the activation of the OTP usages itself.

However, for proliferation of card-type OTPs, the power consumption issue as well as restricted area should be addressed. The card-type OTPs should be embodied in a much more restricted area. This means that their batteries as well as their OTP modules should have a small area; therefore, their batteries have shorter lifetimes (approximately 2 to 3 years) than the existing token-types. As recharging or replacing these batteries is difficult or impossible due to the characteristic of hardware OTPs, reissuing a new one is often necessary, but this is not desirable because a card-type OTP is relatively expensive. Therefore, it is necessary to reduce power and area consumption of a card-type OTP while maintaining its security and functionality. Achieving these reductions is the motivation for this paper.

In this paper, we present an Advanced Encryption Standard (AES)-based OTP device and show that the proposed device outperforms traditional hash-based OTP (HOTP) generators in terms of power and area consumption. First, we implemented a prototype AES hardware module using a simplified SubBytes operation block over the composite Galois field $GF((2^4)^2)$ as a real chip using 350 nm. It uses an 8-bit data path and an 8-bit processing unit to reduce area and power consumption. It stores the current data and key using 8×32 bit memory type storage with latches to minimize area and power consumption. Next, we implemented an OTP hardware module using this prototype AES hardware module as a real chip using a 180 nm process and compared the area and power consumption of our AES-OTP hardware module with that of the HOTP hardware module in the same chip implementation. We also compared the power consumption with software implementation over an Atmega64 microcontroller. According to our analysis, the power consumption of our AES-OTP is 49.4% and 15.0% of those of an HOTP and a software-based OTP, respectively.

The remainder of this paper is organized as follows. Section II introduces the existing hardware OTP (the token-type and card-type) and the area and battery issues of card-type OTPs. Section III shows the design of the proposed prototype AES module and compares its area and energy consumption with other cryptographic modules which can be used for a hardware OTP. Section IV gives the design of an OTP generation module using this prototype AES and analyzes its performance. Finally, we conclude in section V.

II. Preliminary

1. OTP

OTP [1] is an instant password. Different passwords are used every time we authenticate. OTP has a characteristic making it impossible to predict the next password from the current password. The process of the OTP generation consists of i) input value, ii) OTP generation, iii) OTP extraction, and iv) output value.

The OTP generation algorithm generates an OTP value from an input value. It is based on either one-way hash functions or symmetric ciphers and uses the shared input value between the server and the OTP device. This value can be a query value, a time value, a counter value (increasing as the event proceeds), and so on. It is used as the key and data of the generation algorithm. The extraction algorithm of the OTP value extracts the real OTP value (6 to 8 digit binary value) from the output value of the OTP generation algorithm.

There are some options for the OTP generation algorithms, for example, one-way hash functions, symmetric ciphers, and hash based MAC algorithms. Some OTP generation algorithms are currently available as follows:

- One-way hash: SHA-2 family, HAS-160, and FORK 256
- Symmetric crypto algorithm: 3DES, SEED, and AES
- Hash-based MAC: HMAC.

2. Hardware OTP Products

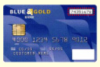
There are two main kinds of products related to the OTP: mobile OTP and hardware OTP. Mobile OTP is the embedded software for mobile phones. Since the OTP generation algorithm is embedded as a software module in mobile phones using the virtualization method (virtual machine), the advantage of using this kind of OTP is that the users do not need any separate devices; however, users do need to use mobile phones that can support OTP generation algorithms. Moreover, users are at the risk of fabrications and modifications of the OTP if their mobile phones are duplicated or stolen.

The hardware OTP is a kind of separate hardware medium. It embeds the calculation functions which separately generate OTP values using cipher algorithms. It can also be divided into token-type ones and card-type ones. Table 1 shows token-type OTPs. The token-type is the most commonly used type. It can come in the form of calculators, key rings, pagers, USB storage,

Table 1. Specification of token-type of hardware OTP.

Corporation initial	I.	V.	A.	D.
Shape				
Algorithm	HAS-160	HOTP	HOTP	HOTP
Embodiment	software	software	not available	not available
Size (mm^3)	40×25×9	48×25×9.5	45×38×11	not available
Lifetime (year)	4	7	not available	rechargeable

Table 2. Specification of hardware card-type of OTP.

Corp. initial	M.	I.	A.	D.
Shape				
Algorithm	HOTP	HOTP	HOTP	HOTP
Embodiment	software	software	not available	not available
Size (mm^3)	85×54×1.2	86×59×1.2	85.5×54 (thickness is not available)	85.5×54× 0.8
Lifetime (year)	3	3	3	2 to 3

and so on. Since important calculation modules are implemented inside the hardware device, token-type OTP has been recognized as a first degree security device and adopted by most banking services; however, the demands for the portable hardware-type OTP modules are increasing because it is inconvenient to carry a separate token-type device.

To meet this requirement, the card-type OTP was designed to be carried in a wallet as shown in Table 2. It can also be combined with a credit card or an RFID card [2]; however, the card-types are relatively more expensive and have shorter lifetimes than those of token-types.

The thickness of the batteries included by the card-type OTP is likely to be determined by the thickness of this chip. The battery thickness of 0.5 mm to 0.6 mm, which allows 30 mAh to 50 mAh of battery capacity, is required based on lithium-polymer batteries [3].

Figure 1 shows the performance comparisons of lithium-polymer batteries and thin-film batteries, an alternative [3]. Since lithium-polymer batteries have relatively high capacity, it is appropriate for an OTP to require a long lifetime, but it is difficult for a card-type OTP to meet the requirements of thickness; moreover, there are problems of reliability at very high or low temperatures [4]. This problem is critical because users directly carry it within their wallets.

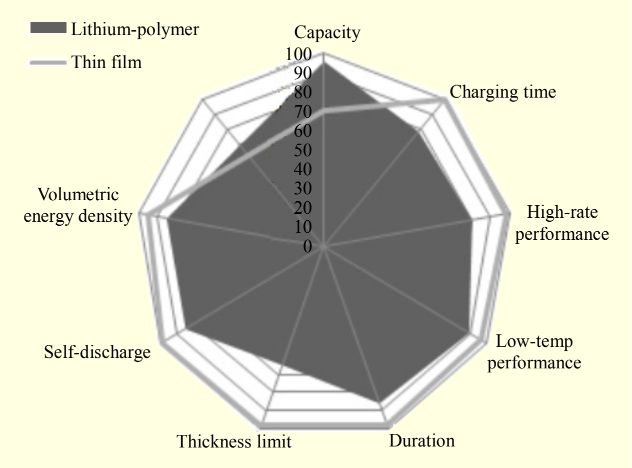


Fig. 1. Comparison of thin-film batteries and lithium-polymer batteries.

The lifetime of card-types is basically 2 to 3 years, which is 2 to 3 years less than token-types; moreover, technical problems, such as thickness and reliability, have not yet been settled. If thin-film batteries are used to avoid these problems, the lifetime of card-type OTP will be much shorter than that of token-types. To compensate for the short lifetime of smaller batteries, it is desirable to use the OTP module of low-power consumption and small-area occupation.

In this paper, we embodied a prototype of a low-power small-area prototype AES [5] hardware module. We embodied an efficient OTP hardware module using this AES module. Then, we implemented it with an HOTP [6] hardware module for the performance comparisons (power consumption and gate area) on the 180 nm chip. We also compared them with software implementations by using a rough method that compares the power consumption of our OTP module with that of Atmega64.

III. Prototype AES Hardware Module

To implement an efficient OTP hardware module, it is necessary to develop a small-area low-power cryptographic module. In this paper, we develop a prototype AES hardware module for this purpose and show that it is a better choice than other AES implementations and hash-based designs. We first present a new approach for SubBytes implementation over a composite field $GF((2^4)^2)$. Our contribution is to use a 4-bit look-up table (LUT) for the underlying subfield $GF(2^4)$. Compared to other SubBytes implementations, such as the $GF(2^8)$ LUT [7]-[9] and $GF(((2^2)^2)^2)$ [10]-[12], our SubBytes implementation consumes a much smaller area. We also present a new technique for the implementation of storage of data and a key, which is the dominant part of an AES module.

Table 3. Reduction polynomials for composite fields.

Finite fields	Reduction polynomials
$GF((2^4)^2)$	$x^2+x+(1001)_{GF(2^4)}$
$GF(2^4)$	x^4+x+1
$GF(((2^2)^2)^2)$	$x^2+x+(1100)_{GF((2^2)^2)}$
$GF((2^2)^2)$	$x^2+x+(10)_{GF(2^2)}$
$GF(2^2)$	x^2+x+1
$GF(2^8)$	$x^8+x^4+x^3+x+1$

We use latches with a modified control signal for prevention of hold violations. Then, we embody the entire AES hardware module as a narrow 8-bit architecture to reduce area and power consumption.

1. Implementation of SubBytes Operation Using Isomorphic Composite Fields and Look-up Tables

We implemented the efficient SubBytes module using the isomorphic composite field $GF((2^4)^2)$ and a 4-bit LUT. AES is based on arithmetic operations on finite field $GF(2^8)$. Since some operations such as multiplicative inversions require large resources, there needs to be some isomorphic fields, such as $GF((2^4)^2)$ and $GF(((2^2)^2)^2)$. Since these isomorphic fields have the same number of elements as the finite field $GF(2^8)$, we can convert elements and operations on $GF(2^8)$ to those of $GF((2^4)^2)$ and $GF(((2^2)^2)^2)$.

Table 3 shows the reduction polynomials for these various finite fields [11], [13]. We can write an element in $GF(2^8)$ as a polynomial of maximum degree 1 with coefficients either in $GF(2^4)$ or $GF((2^2)^2)$. Elements of $GF(2^4)$ or $GF((2^2)^2)$ can also be represented as polynomials recursively.

Figure 2 shows these conversion matrices to convert elements between isomorphic fields [11], [13]. For example, an 8-bit expression defined on $GF(2^8)$ is converted to elements in $GF((2^4)^2)$ and $GF(((2^2)^2)^2)$ by being multiplied to the conversion matrix $\mathbf{M}_{242}$ and $\mathbf{M}_{2222}$, respectively.

Figure 3 shows the process of SubBytes module. To use efficient multiplicative inversion over $GF((2^4)^2)$ or $GF(((2^2)^2)^2)$, we inserted conversion and reconversion matrices to the front and back of the SubBytes module because adding these matrices to the front and back of the multiplicative inversion [14] instead of inserting them to the front and back of the SubBytes module slightly takes up more area and consumes more power.

$$\mathbf{M}_{242}=\begin{bmatrix}1&0&1&1&1&0&1&1\\0&1&0&1&0&0&0&0\\0&1&0&0&1&0&1&0\\0&1&1&0&0&0&1&1\\0&0&0&0&1&1&1&0\\0&1&0&0&1&0&1&1\\0&0&1&1&0&1&0&1\\0&0&0&0&0&1&0&1\end{bmatrix}\quad \mathbf{M}_{242}{}^{-1}=\begin{bmatrix}1&0&0&0&1&0&1&0\\0&0&0&0&1&1&0&1\\0&1&0&0&1&1&1&0\\0&1&0&0&1&1&0&1\\0&1&0&1&1&0&1&0\\0&0&1&0&0&1&0&1\\0&1&1&1&0&1&1&1\\0&0&1&0&0&1&0&0\end{bmatrix}$$

$$\mathbf{M}_{2222}=\begin{bmatrix}1&1&0&0&0&0&1&0\\0&1&0&0&1&0&1&0\\0&1&1&1&1&0&0&1\\0&1&1&0&0&0&1&1\\0&1&1&1&0&1&0&1\\0&0&1&1&0&1&0&1\\0&1&1&1&1&0&1&1\\0&0&0&0&0&1&0&1\end{bmatrix}\quad \mathbf{M}_{2222}{}^{-1}=\begin{bmatrix}1&0&1&0&1&1&1&0\\0&0&0&0&1&1&0&0\\0&1&1&1&1&0&0&1\\0&1&1&1&1&1&0&0\\0&1&1&0&1&1&1&0\\0&1&0&0&0&1&1&0\\0&0&1&0&0&0&1&0\\0&1&0&0&0&1&1&1\end{bmatrix}$$

Fig. 2. Field conversion matrices.

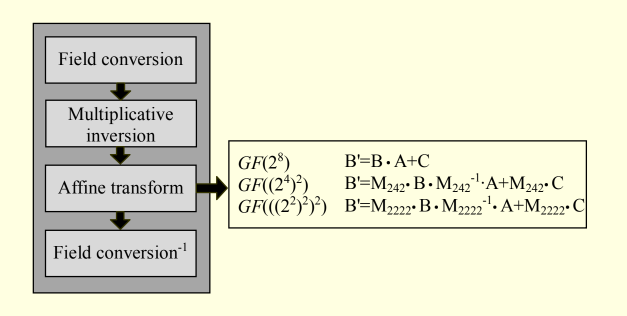


Fig. 3. Modified SubBytes modules.

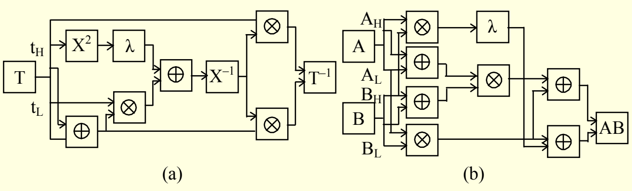


Fig. 4. Inversion circuit using (a) $GF((2^4)^2)$ and (b) $GF(((2^2)^2)^2)$.

Here, we show the details of the SubBytes operation over composite fields. First, we describe the case for $GF(((2^2)^2)^2)$ [11]. Since the affine transform is straightforward, we explain only the computation of a multiplicative inverse. Figure 4(a) shows a circuit for multiplicative inversion.

Let $T_{(x)}=t_{\mathrm{H}}x+t_{\mathrm{L}}$ be the input where $t_{\mathrm{H}},t_{\mathrm{L}}\in GF((2^2)^2)$; let $P(x)=x^2+x+\lambda$ be the reduction polynomial where $\lambda=1100\in GF((2^2)^2)$; let $\Theta=(t_{\mathrm{H}}{}^2\lambda+t_{\mathrm{L}}(t_{\mathrm{H}}+t_{\mathrm{L}}))^{-1}$; and let $Q(x)$ and $R(x)$ be the quotient and remainder, respectively, when $P(x)$ is divided by $T(x)$, that is, $P(x)=T(x)Q(x)+R(x)$. It can be easily proved by the extended Euclid algorithm that $T(x)^{-1}=(t_{\mathrm{H}}\Theta)x+(t_{\mathrm{H}}+t_{\mathrm{L}})\Theta$ [15].

Figure 4 shows the efficient inversion circuits of SubBytes using the composite field $GF((2^4)^2)$ or $GF(((2^2)^2)^2)$. We used

Table 4. SubBytes synthesis results.

SubBytes method	Gate equivalent (GE)	
	350 nm	180 nm
$GF(2^8)$	523.7	632.0
$GF(((2^2)^2)^2)$ [11]	323.0	389.3
$GF((2^4)^2)$	300.3	359.7

the inversion circuit using three Mastrovito multipliers in $GF((2^2)^2)$ [15]. This technique of substituting multipliers by adders can be applied recursively. It results in the inversion circuit of $GF(((2^2)^2)^2)$.

We implemented the inversion circuit of $GF((2^4)^2)$ that we propose. It uses the same inversion circuit but uses three different multiplier circuits using a 4-bit LUT, that is, the multiplication table over $GF(2^4)$, optimized by Karnaugh map instead of Mastrovito multipliers in the inversion circuit. Like $GF(((2^2)^2)^2)$, an element in $GF(2^4)$ can be represented in polynomial form as

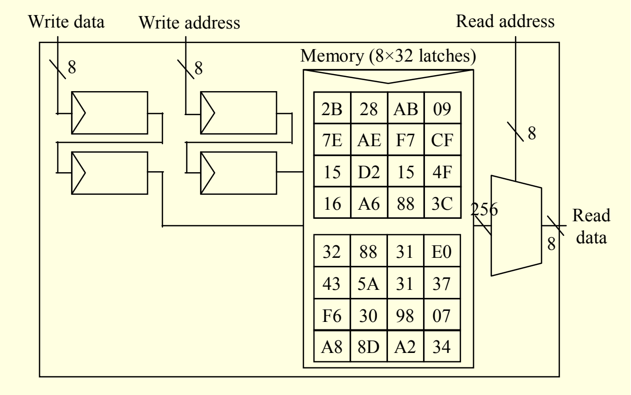
$$\begin{aligned} &T(x) = t_{\mathrm{H}}x + t_{\mathrm{L}}, && t_{\mathrm{H}}, t_{\mathrm{L}} \in GF(2^4), \\ &\text{Let}, P(x) = x^2 + x + \lambda, && \lambda(1001)_2 \in GF(2^4), \\ &\Theta = (t_{\mathrm{H}}{}^2\lambda + t_{\mathrm{L}}(t_{\mathrm{H}} + t_{\mathrm{L}}))^{-1}. && \end{aligned}$$

The LUT method is used to solve Θ . Implementation of the LUT method requires a small multiplexer. Because the length of Θ is just 4 bits and input bits are 8 bits, we use the LUT and the Karnaugh map to optimize the duplicate values.

Table 4 shows the synthesis results of SubBytes modules. This result shows that our proposed $GF((2^4)^2)$ SubBytes module has a smaller area than [11]. We synthesized our SubBytes module, the SubBytes module in [11], and a general SubBytes module using the LUT in $GF(2^8)$. We used 350 nm CMOS cell-based libraries. The SubBytes using LUT in $GF(2^8)$ uses the largest area among all SubBytes methods. Even though SubBytes modules in $GF((2^4)^2)$ use a 4-bit LUT, the result is better than that of $GF(((2^2)^2)^2)$ because a 4-bit LUT is generally provided as a built-in base cell in CMOS standard cell-based libraries; therefore, we used the SubBytes modules in $GF((2^4)^2)$ for the AES hardware module.

2. Efficient Storage Module Using Latches

We embody efficient SubBytes for compact AES; however, it is well known that SubBytes are only a small part of an AES hardware module [16]. On the contrary, the storage of data and keys takes up a very large part of an AES module. Therefore, we tried to optimize the storage of data and keys. Our technique is to use a latch-based 8×32 bit memory.



2B	28	AB	09
7E	AE	F7	CF
15	D2	15	4F
16	A6	88	3C
32	88	31	E0
43	5A	31	37
F6	30	98	07
A8	8D	A2	34

Fig. 5. 8×32 memory implementation.

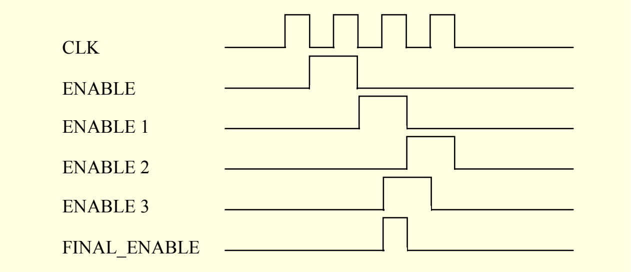


Fig. 6. Modified enable signal.

We embodied an 8×32 bit memory to store a state (current processing data) and the current key (extended key). The function of reading data is implemented using a combinational circuit. The output of the reading data at the input address comes promptly. On the other hand, the data written is stored after two cycles to avoid the collision of reading and writing at the same address at the same time.

Figure 5 shows how the input is entered. Data read is implemented by an 8-bit MUX, and registers are not used. However, registers should be used to set back the write operation by 2 cycles so that we may avoid writing the new data before reading the previous data. Thus, two registers are used to store temporary data and its target address.

To make a more compact AES, we embodied a memory using latches instead of flip-flops. Flip-flops are generally designed with two latches; moreover, flip-flops make clock trees, which take up more area and consume more power. It means that flip-flops use approximately twice the area and power than latches; however, latches often generate timing violation and signal uncertainties. Because our logic operates at very low frequencies, signal uncertainties are not big problems, but the timing violation requires one cycle of hold time. To fix this problem, we halved the hold time using the modified enable signals and some registers: Enable 1 to 3. Figure 6 shows how the modified enable signal is made.

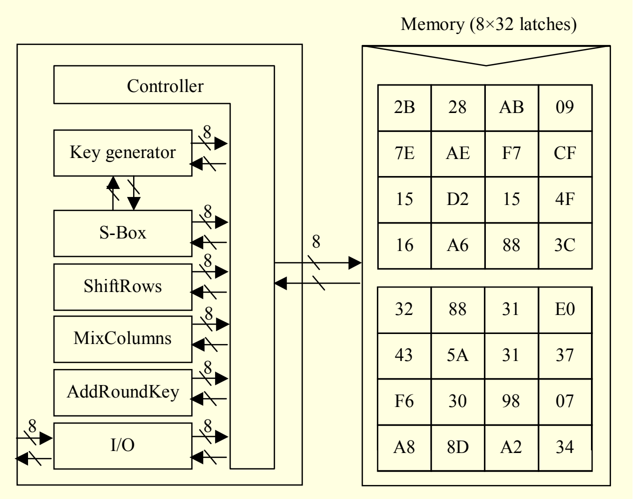


Fig. 7. Block diagram of AES hardware module.

3. Data Communication

The data path is the most significant factor to reduce areas. Due to our narrow 8-bit bus, overall combinational logics and flip-flops could be reduced. Because just 8 bits are changed at one clock, low-power can be also achieved.

Figure 7 shows our data path design. The prototype AES design can be divided into three parts. The first one is a crypto-module. It performs basic operations of AES and generates round keys. The S-Box (SubBytes implementation) is shared by a round key generator and SubBytes operations. The second part is a 256-bit memory to store the current round key and state. We used latches to implement memory. The last one is a controller. It controls data transfer between crypto-modules and memory. To reduce registers, we use only four 8-bit registers for MixColumns and two 8-bit registers for AddRoundKey instead of 8×32 memory.

4. Experimental Results and Analysis

Table 5 shows the synthesis result of the prototype AES hardware module using a 350 nm process. Compared with the results in [16], we see that the areas for SubBytes and storage were significantly reduced. As a result, the area of the entire module was also reduced.

We need to compare the prototype AES hardware module with other OTP hardware generation modules. These may be AES and SHA-1 but also other standard algorithms, such as 3DES [17] and SEED [18]. However, to the best of our knowledge, there has been no known result in the literature that focuses on OTP hardware modules. Therefore, we only compare core crypto modules, that is, we compare the prototype AES module with other hardware modules that can be used as an OTP generation algorithm. These modules are 3DES [19], SEED [20], SHA-1 [21], and alternative implementations of AES [13], [22], [23].

Table 5. Areas of submodules of the prototype AES module.

Total (GE)	SubBytes	Key expansion	Mix columns	Storage of data and key	Others
2,874	303.0	97.0	427.7	1,543.0	503.3

Table 6. Comparison of OTP generation algorithm.

Algorithm	Area (GE)	Voltage (GE) (V)	Current (mA)	Power (mW)	No. of cycles	Process (nm)
3DES [19]	4,321	1.8	1.6444	2.9600	98	180
SEED [20]	8,742	1.8	-	-	113	180
AES [16]	3,595	3.3	0.0082	0.0270	992	350
AES [22]	3,400	1.5	0.0030	0.0045	1,032	350
AES [23]	10,799	-	-	0.0470	64	600
Prototype AES	2,874	3.3 (1.5)	0.0027 (0.0020)	0.0090 (0.0030)	1,075	350
SHA-1 [21]	6,122	1.8	0.0076	0.0140	344	180
Our SHA-1	8,882	1,8	0.9500	1.7100	86	180

Table 6 shows the performance comparisons of OTP generation algorithms. The 3DES module [19] seems to have the smallest area except AES modules; however, this does not include registers because it shares registers with other cipher modules. The results in [20] for SEED hardware module do not provide power consumption data, but we believe that it will use more power than ours because it took up a much wider area than ours.

According to Table 6, the AES implementation in [22] has good power consumption behavior because it operates on 1.5 V. However, in general, a recommended normal supply voltage is from 3.0 V to 3.3 V in 0.35 μm processes. Note that our module can also operate on 1.5 V using 2.0 μA current at clocks of 100 kHz or lower. As a result, our module may have only 67% power consumption of that of [22] on the same 1.5 V. Because the numbers of clock cycles used for [22] and our implementation are almost the same, we may expect a similar improvement in energy consumption.

Now, we compare our result with a typical 32-bit implementation [23]. In general, 32-bit architectures are known as the best choice to implement energy-efficient AES hardware modules. However, we see that the energy consumption in our implementation is almost the same as that of [23] because our module has an operation time that is 16.8 times longer and consumes 16.67 times less power than that in [23].

Moreover, there is another important factor to be considered for a card-type OTP device. An OTP module in a card-type device should only take up a small area as well as consume a small amount of energy. While the energy consumption in our implementation is almost the same as that of [23], the required chip area for our module is much smaller. Although it is also possible to further reduce the energy consumption of a 32-bit AES hardware module, an 8-bit architecture is more preferable than a 32-bit architecture for a card-type OTP having the duel goals of energy and area optimization.

Finally, we compare our result with hash-based modules. According to Table 6, the SHA-1 module in [21] consumes less energy by 2.04 times and more area by 2.13 times than our prototype AES implementation in the setting that our module is implemented on a 350 nm process and it operates on 3.3 V. However, if our module operates on 1.5 V, it will use less energy than [21] by 0.67 times. Moreover, in our final embodiment of AES on a 180 nm process, which is the same process as that used in [21], the energy consumption will be reduced further.

IV. Low-Power OTP Hardware Generation Module

1. Block Diagram of OTP Hardware Generation Module

To verify the performance of our OTP module, we implemented the prototype AES hardware module together with a SHA-1 hardware module. Then we designed two OTP hardware modules (AES-OTP and HOTP), one of which is based on the MAC modules [24] using the prototype AES module, and the other using a SHA-1 module.

Figure 8 shows the block diagram of the OTP hardware module. For the symmetric crypto algorithm (AES) or hash algorithm (SHA-1) hardware module, 128-bit or 512-bit keys and messages are used as input directly. The MAC hardware module accumulates AES or SHA-1 result values and computes the MAC value. The dynamic truncation and computation of the OTP module extracts an OTP value based on the MAC value.

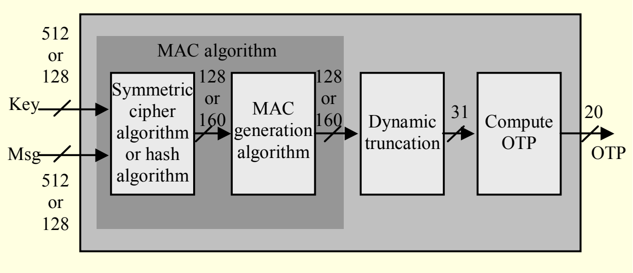


Fig. 8. Block diagram of OTP hardware module.

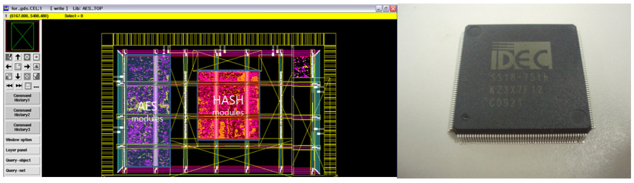


Fig. 9. Layout and real chip of OTP modules.

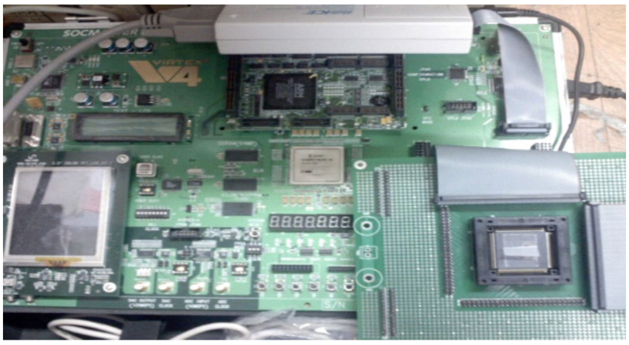


Fig. 10. Test environment.

2. Implementation of OTP Hardware Module

We synthesized and implemented our OTP hardware modules using OTP generation algorithms. These modules contain an AES-OTP module using the prototype AES hardware module and an HOTP module using SHA-1 on the 180 nm process based on the standard cell library. Figure 9 shows the layout and the real chip of OTP modules.

We evaluated the performance of these OTP modules over a Huins SoC Master3 board. This board is equipped with a Xilinx Virtex-4 XC4VLX60 FPGA with an ARM926EJ embedded CPU. The board connects with the OTP chip using an extension socket. Since the operation voltage of the chip is 1.8 V, the board supplies the voltage to the chip with an LM1117 regulator.

Figure 10 shows the test environment for our OTP chip. The PC (CoreDuoE5200, 2G RAM) connects with the board using a serial cable. We controlled and verified input/output values of the chip using the serial program at the PC.

Table 7 and Fig. 11 show the experimental results in this environment. Since the target device of our OTP module is the card-type OTP and should consume very low power, we set the operating frequency of our hardware module at 1 MHz. We observed that our AES-OTP hardware module used about half the area (54.7%) and power consumption (49.4%) of the HOTP hardware module. We had similar results for frequencies from 1 MHz to 20 MHz. AES-OTP and HOTP extract an OTP from a MAC value generated by MAC algorithms: AES-CMAC and HMAC-SHA-1. These MAC algorithms use the crypto algorithm (AES, SHA-1) twice, and

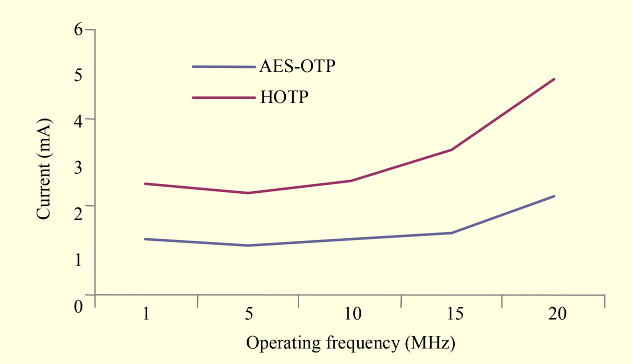


Fig. 11. Measured currents for various operating frequencies.

Table 7. Measured areas and currents.

Module	Frequency (MHz)	Area (GE)	Current (mA)	Voltage (V)
Atmega64 (S/W)	4	-	5.50	2.7 to 5.5
HOTP	1	29,480.5	2.51	1.8
AES-OTP	1	16,126.5	1.24	1.8

the remainders of the MAC modules are similar; therefore, the differences in performance of the two OTP modules result from the differences between the prototype AES hardware module and the SHA-1 hardware module.

According to Table 7, our AES-OTP module consumes 7 times less power than a software module, even under the extremely conservative assumption that during an OTP computation Atmega64 is always in its idle mode at a voltage of 2.7 V.

V. Conclusion

In this paper, we implemented a low-power small-area OTP hardware module appropriate for card-type OTPs. We first implemented an AES hardware module with the lowest power consumption and smallest area. It has an efficient SubBytes module using composite field $GF((2^4)^2)$. It stores the temporary data and key using latches instead of flip-fops. We compared this with other OTP generation algorithms (another implementation of AES [16], SEED [20], and 3DES [19]). As a result, the AES hardware module that we suggested showed the best performance in terms of area occupation and power consumption.

Using this AES hardware module, we implemented the OTP hardware module. To verify its performance, we made a real chip on the 180 nm process. This chip also contains an HOTP module as well as our AES-OTP module for comparison. We compared not only the performances between these two hardware modules but also the performances against the software OTP. As a result, we verified that our AES-OTP hardware module has the lowest power consumption among them. If we apply it to a real card-type OTP, its lifetime would be dramatically improved. We remark that, however, for power consumption evaluation in a full transaction, the energy consumption to display the OTP values should also be considered. We leave this issue for our future research.

Finally, we should consider side channel attacks because we are dealing with cryptographic operations over a small device. The most powerful side channel attacks are power analysis attacks. They are based on the assumption that the device obtains power from an external power source, and the attacker can observe the power consumption behavior of the device by probing the connection to the power source. However, it is impossible to directly measure power consumption of an OTP device because it operates using a battery inside the device and is tamper-proof. However, electromagnetic analysis (EMA) attacks may be potential threats unless appropriate shielding techniques are applied. We leave the EMA against an OTP and its countermeasure as another future research topic. As well as various well-known randomization techniques, hardware-based techniques such as adiabatic logic [25], which has exactly the same power consumption regardless of the values, would be a promising candidate for a countermeasure.

References

[1] N. Haller and C. Metz, *A One-Time Password System*, Internet RFC 1938, May 1996.

[2] ISO standards, “Information Technology–Identification Cards–Financial Transaction Cards,” ISO/IEC 7813, 2006.

[3] Emerging Issue Report, Korea Institute of Science and Technology Information (KISTI), “The Directions of Future Technologies and Market Competitions of Thin Film Batteries and Thin Lithium Batteries,” 2007.

[4] A. Patil et al., “Issue and Challenges Facing Rechargeable Thin Film Lithium Batteries,” *Mater. Res. Bull.*, vol. 43, 2008, pp. 1913-1942.

[5] National Institute of Standards and Technology, “Advanced Encryption Standard,” FIPS PUB 197, 2001.

[6] IETF Network Working Group, “HOTP: An HMAC-Based One-Time Password Algorithm,” IETF RFC 4226, 2005.

[7] N. Pramstaller and J. Wolkerstorfer, “A Universal and Efficient AES Co-processor for Field Programmable Logic Arrays,” *FPL* , *LNCS*, vol. 3203, 2004, pp. 565-574.

[8] J. Zambreno, D. Nguyen, and A. Choudhary, “Exploring Area/Delay Tradeoffs in an AES FPGA Implementation,” *FPL* , *LNCS*, vol. 3203, 2004, pp. 575-585.

[9] P. Chodowiec and K. Gaj, “Very Compact FPGA Implementation of the AES Algorithm,” *CHES*, *LNCS*, vol. 2779, 2003, pp. 319-333.

[10] J. Wolkerstorfer, E. Oswald, and M. Lamberger, “An ASIC Implementation of the AES SBoxes,” *CT-RSA*, *LNCS*, vol. 2271, 2002, pp. 29-52.

[11] A. Satoh et al., “A Compact Rijndael Hardware Architecture with S-Box Optimization,” *ASIACRYPT*, *LNCS*, vol. 2248, 2001, pp. 239-254.

[12] A. Rudra, “Efficient Rijndael Encryption Implementation with Composite Field Arithmetic,” *CHES*, *LNCS*, vol. 2162, 2001, pp. 171-184.

[13] S. Chantarawong and S. Choomchuay, “An Architecture for a Compact AES System,” *Proc. 1st Electrical Eng./Electron., Computer, Telecommun. Inf. Technol. (ECTI) Annual Conf.*, May, 2004.

[14] A. Hodjat and I. Verbauwhede, “A 21.54 Gbits/s Fully Pipelined AES Processor on FPGA.” *FCCM*, Apr., 2004, pp. 308-309.

[15] T. Good and M. Benaissa, “Pipelined AES on FPGA with Support for Feedback Modes (in a Multi-Channel Environment),” *IET Information Security*, vol. 1, no. 1, 2007, pp. 1-10.

[16] M. Feldhofer, S. Dominikus, and J. Wolkerstorfer, “Strong Authentication for RFID Systems Using the AES Algorithm,” *CHES*, *LNCS*, vol. 3156, 2004, pp. 357-370.

[17] ANS X9.52-1998, “Triple Data Encryption Algorithm Modes of Operation,” 1999.

[18] Korea Information Security Agency (KISA), “SEED Algorithm Specification,” 1999.

[19] Y. Kim and Y. Jeong, “Low Power Implementation of Integrated Cryptographic Engine for Smart Cards,” *J. Institute Electron. Eng. Korea*, vol. 45, no. 372, June 2008, pp. 80-88.

[20] J. Hwang, “Efficient Hardware Architecture of SEED S-Box for Smart Cards,” *J. Semiconductor Technol. Sci.*, vol. 4, no. 4, Dec. 2003, pp. 307-311.

[21] M. O’Neill, “Low-Cost SHA-1 Hash Function Architecture for RFID Tags,” Hand. of Conf. RFID Security, 2008.

[22] M. Feldhofer, J. Wolkerstorfer, and V. Rijmen, “AES Implementation on a Grain of Sand,” *IET Proc. Info. Security*, vol. 152, no. 1, 2005, pp. 13-20.

[23] S. Mangard, M. Aigner, and S. Dominikus, “A Highly Regular and Scalable AES Hardware Architecture,” *IEEE Trans. Comput.*, vol. 52, no. 4, 2003, pp. 483-491.

[24] D. Kim et al., “Design and Performance Analysis of Electronic Seal Protection Systems Based on AES,” *ETRI J.*, vol. 29, no. 6, Dec. 2007, pp. 755-768.

[25] W. Athas et al., “Low-Power Digital Systems Based on Adiabatic-Switching Principles,” *IEEE Trans. VLSI Syst.*, vol. 2, no. 4, Dec. 1994, pp. 398-407.

Sung-Jae Lee received his BS in mathematics from Korea University in 1999. He is working as a senior researcher in Korea Internet & Security Agency (KISA), Seoul, Rep. of Korea. His research interests include cryptography and its efficient implementations.

Jae Seong Lee is a PhD candidate at the Embedded Security System Laboratory, School of Electrical and Computer Engineering, Hanyang University, Seoul, Rep. of Korea. His research interests are in the areas of security and embedded systems.

Mun-Kyu Lee received his BS and MS in computer engineering from Seoul National University, Seoul, Rep. of Korea, in 1996 and 1998, respectively, and his PhD in electrical engineering and computer science from Seoul National University, Rep. of Korea, in 2003. From 2003 to 2005, he was a senior engineer at ETRI, Rep. of Korea. He is currently an assistant professor in the School of Computer and Information Engineering at Inha University, Incheon, Rep. of Korea. His research interests include information security and theories of computation.

Sang Jin Lee received his PhD in mathematics from Korea University, Rep. of Korea, in 1989. He was a professor in the Department of Mathematics at Korea University from March 1999 to August 2001, and has been a professor of the Graduate School of Information Management and Security at Korea University since August 2001. He is also a president of the Cryptography Research Society and a director for industry-academic cooperation at the Korea Institute of Information Security and Cryptology. His research interests include cryptography, digital forensic, information hiding, hash functions, and MACs.

Doo-Ho Choi received his BS in mathematics from Sungkyunkwan University, Seoul, Rep. of Korea, in 1994, and the MS and PhD in mathematics from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Rep. of Korea, in 1996 and 2002, respectively. He has been a senior researcher at ETRI, Daejeon, Rep. of Korea, since January 2002. His current research interests are side channel analysis and its resistant crypto design, security technologies of RFID and wireless sensor network, lightweight cryptographic protocol/module design, and cryptography based on non-commutativity. He was an editor of the ITU-T Rec. X.1171.

Dong Kyue Kim received the BS, MS, and PhD in computer engineering from Seoul National University, Seoul, Rep. of Korea, in 1992, 1994, and 1999, respectively. From 1999 to 2005, he was an assistant professor in the Division of Computer Science and Engineering at Pusan National University. He is currently an associate professor in the Division of Electronics and Computer Engineering at Hanyang University, Seoul, Rep. of Korea. His research interests are in the areas of embedded security systems, crypto-coprocessors, and information security.