Linear Corrector Overcoming Minimum Distance Limitation for Secure TRNG from (17, 9, 5) Quadratic Residue Code

Young-Sik Kim, Ji-Woong Jang, and Dae-Woon Lim

A true random number generator (TRNG) is widely used to generate secure random numbers for encryption, digital signatures, authentication, and so on in crypto-systems. Since TRNG is vulnerable to environmental changes, a deterministic function is normally used to reduce bias and improve the statistical properties of the TRNG output. In this paper, we propose a linear corrector for secure TRNG. The performance of a linear corrector is bounded by the minimum distance of the corresponding linear error correcting code. However, we show that it is possible to construct a linear corrector overcoming the minimum distance limitation. The proposed linear corrector shows better performance in terms of removing bias in that it can enlarge the acceptable bias range of the raw TRNG output. Moreover, it is possible to efficiently implement this linear corrector using only XOR gates, which must have a suitable hardware size for embedded security systems.

Keywords: AIS.31 standard, key generation, nonce, post-processing, statistical tests, Shannon entropy, true random number generator (TRNG).

Manuscript received Mar. 1, 2009; revised July 13, 2009; accepted Aug. 17, 2009.

This work was supported by Dongguk University.

Young-Sik Kim (phone: + 82 16 251 2418, email: mypurist@gmail.com) is with the Department of System LSI, Samsung Electronics, Co., Ltd., Rep. of Korea.

Ji-Woong Jang (corresponding author, phone: +1 858 761 7275, email: stasera.jang@gmail.com) is with the Department of Electrical and Computer Engineering, University California San Diego, San Diego, California, USA.

Dae-Woon Lim (email: daewoonlim@gmail.com) is with the Department of Information and Communication Engineering, Dongguk University, Seoul, Rep. of Korea.

doi:10.4218/etrij.10.0109.0141

I. Introduction

The security of most crypto-systems is strongly dependent on the unpredictability and irreproducibility of the digital secret key used. Therefore, most standards require the use of a cryptographically secure random number generator (RNG), to produce the digital key stream [1]. RNGs can be classified into two classes: pseudo-random number generators (PRNGs) and true random number generators (TRNGs). While PRNGs use various mathematical algorithms to generate random numbers [2]-[4], TRNGs use various physical noise sources [5]-[9]; therefore, TRNG is also called a physical random number generator. Note that the output of various PRNGs can be predicted by using previous output, and can be reproduced by using the same initial value. On the other hand, it is not possible to algorithmically predict and reproduce the TRNG output. Therefore, many crypto-systems, such as smart cards [10], [11], select a TRNG as their random number generator or a random seed generator for a PRNG.

However, since a TRNG is based on a physical random source, it can be influenced by environmental changes, such as the temperature, electromagnetic field, and so on. Moreover, the statistical quality of the random output can worsen as time advances due to aging. Thus, various biases are commonly expected in TRNG output, and these can be exploited to attack the system by statistical estimation of the next TRNG output bit. In that case, the TRNG output can fail a series of statistical tests [1], [12]-[14] which are required for secure operation of a crypto-system.

Therefore, TRNG should be used in combination with

complementary systems such as an online test [1], [14], [15], to check the quality of the TRNG output in practice or post-processing [6], [7], [16]-[21] to improve the statistical quality of the TRNG output. Because hardware and software resources are restricted in most applications, the cost of the complementary TRNG systems should be minimized.

In [21], Dichtl showed that the bijective post-processing functions proposed by Markovski and others [20] are not sufficient for cryptographic applications. Instead, he proposed new post-processing methods which were improved by a heuristic approach. Recently, Lacharme [16] reinterpreted Dichtl's post-processing methods by using a linear code [22]. By using the generator matrix of a linear code, it is possible to obtain various post-processing functions which have comparable performance to Dichtl's methods. Lacharme expected that a post-processing function generated from a linear code would be limited by the minimum distance of the linear code. Therefore, he focused on non-linear functions to further improve the performance of post-processing. As a result, he discovered another non-linear post-processing method which has better performance and is similar to Dichtl's heuristic corrector. Although Lacharme's non-linear corrector is easier to implement than Dichtl's heuristic one, it is still more complex to implement than the linear corrector.

In this paper, we show that it is possible to construct linear correctors that overcome the minimum distance limitation. As an example, a binary (17, 9, 5) quadratic residue (QR) code [22] which is a cyclic code (also a linear code) is used to construct a linear corrector that achieves better performance than Lacharme's non-linear corrector and Dichtl's heuristic corrector. The compression rate of the proposed corrector from the (17, 9, 5) QR code is 0.53 because the input size is 17 bits and the output size is 9 bits. We analyze and compare the performance of the proposed correctors with previous ones. Because the proposed correctors are based on a linear code, it is possible to implement them simply by using a series of XOR gates. The proposed corrector from the (17, 9, 5) QR code with 2 iterations can be implemented by using a hardware area smaller than 500 NAND gates, which is a small enough hardware size to be suitable for many types of embedded crypto-systems.

The remainder of this paper is organized as follows. In section II, previous post-processing methods are briefly reviewed. In section III, a criterion to construct a good corrector is presented using an information theoretic analysis, and the new linear corrector is presented using a cyclic code, namely, QR code. The performance of the proposed corrector is compared with previous correctors in section IV. Then, the possible hardware architecture of the proposed method is discussed and estimated in section V. Finally, concluding remarks are given in section VI.

II. Post-processing of TRNG

1. Classification of Post-processing Method

In this paper, we assume that the random numbers from a given random source are statistically independent. The bias of the random number is defined as

$$e = \frac{1}{2}(\Pr(x_i = 0) - \Pr(x_i = 1)).$$

We also assume that the random source is stationary. That is, the bias does not change as time advances. Because the random numbers are statistically independent, the bias of a linear combination of random numbers can be represented by using the bias e. In that case, the following definition [16] is useful for estimating the performance of a corrector.

Definition 1. Let P be a polynomial of degree d, defined by

$$P(x) = \sum_{i=0}^{d} a_i x^i.$$

The valuation of P is the minimal $i > 0$ such that $a_i \neq 0$.

Post-processing methods of TRNG can be classified into two groups according to the required input bits. The first one improves the TRNG output for a fixed length of input, and the other produces perfectly unbiased TRNG output for an infinite length of input. For the latter group, it is sometimes necessary to wait until the required amount of output is collected. The most famous method in this class is the von Neumann corrector [17]. However, because there is usually a time limitation in an actual system, a time-out violation occurs and the operation may be stopped if the response for a random number request is not generated within a limited time. For this reason, the latter group is not suitable for most systems.

Conversely, methods in the former group generate a k-bit output using an n-bit input, where $n \geq k$. In that case, a corrector can be considered as a deterministic function. Since it is information theoretically impossible to increase the entropy *per se* of a given sequence by using deterministic processing, we focus on increasing the entropy per bit. Note that if the corresponding function is bijective, then it is impossible to increase the entropy per bit. Therefore, to obtain the desired unbiased output, the input size must be greater than the output size.

2. Bounded Performance with Fixed Length of Input

One of the widely used correctors with a fixed length input is the XOR corrector, given as

$$y_i = x_{2i} + x_{2i+1} \bmod 2.$$

Because this corrector produces one output bit by using two

input bits, the compression rate is 0.5. Dichtl [21] proposed new linear correctors, namely, the D_1, D_2, and D_3[1] correctors.

Let X_1 and X_2 be two input bytes. Let + be bit-wise XOR, and let $RL(X, i)$ be i-bit cyclic rotation. Then, D_1, D_2, and D_3 correctors can be represented as in [16] as

$$D_1(X_1, X_2) = X_1 + RL(X_1, 1) + X_2,$$
$$D_2(X_1, X_2) = H_1(X_1, X_2) + RL(X_1, 2),$$
$$D_3(X_1, X_2) = H_2(X_1, X_2) + RL(X_1, 4).$$

The valuations of the bias terms of D_1, D_2, and D_3 are 3, 4, and 5, respectively. Lacharme analyzed these functions using a linear code. In the next section, we examine the details.

3. Perfect Performance with Infinite Length of Input

The von Neumann corrector can generate unbiased output if the raw random numbers are generated from an independent identically distributed (i.i.d.) random source. In the von Neumann corrector, a pair of inputs, 01 or 10, produces the output 0 and 1. Otherwise, there is no output and the corrector tests the next pair of inputs. Since the probabilities of pairs 01 and 10 are $\left(\frac{1}{2}-e\right)\left(\frac{1}{2}+e\right)$, the von Neumann corrector generates an unbiased output for the case of an i.i.d. random source.

Suppose that e is in the range $[-1/2, 1/2]$. Denote e as $\frac{1}{2}-\varepsilon$. Then, $\frac{1}{4}-e^2 = \frac{1}{4}-\left(\frac{1}{2}-\varepsilon\right)^2 = \varepsilon-\varepsilon^2$. If $\varepsilon \to 0$, then the probability of 01 or 10 in the raw inputs approaches zero. To obtain a k-bit post-processed output, we need a $k/2\varepsilon$ bit raw input. That is, a large input size is needed to produce an unbiased output for a small ε. At best, the probability of producing output is 1/4; thus, the maximum compression rate is 0.25. Moreover, if the assumption of statistical independence does not hold, the von Neumann corrector can no longer ensure an unbiased output.

III. Linear Corrector from Linear Code

1. Performance Estimation of Individual Output Bits

Let C be a linear code. For $a, b \in C$, $a+b$ is also a codeword in C. An $[n, k]$ linear code is constructed from a set of k linearly independent codewords with length n. It can be represented by using a $n \times k$ generating matrix G as

$$mG = c,$$

where m is a k-bit message, and c is the corresponding codeword.

In Lacharme's analysis, Dichtl's methods can be interpreted as a generator matrix of linear code. Let x be a raw random input from TRNG, and let y be a post-processed random output. Then, Dichtl's D_1 corrector can be represented by using the generator matrix as

$$G_{D_1} = \begin{pmatrix} 0000000100000011 \\ 0000001000000110 \\ \vdots \\ 1000000010000001 \end{pmatrix}.$$

In a linear code, except for some special code such as simplex code with equidistance property, many codewords have a larger Hamming weight than the minimum distance of the code. Moreover, since a generator matrix can be generated using any k linearly independent codewords in the code, it is possible to construct a generator matrix using the codewords with a larger Hamming weight than the minimum distance.

In [16], Lacharme derived the following result to obtain the representation of the bias for some Boolean functions. Define the linear combination of a Boolean function using non-zero k-tuple vector u as

$$\varphi_u(x) = \sum_{i=1}^{m} u_i f_i(x) = u \cdot f(x).$$

Then, the bias Δ_u can be defined as

$$\Delta_u = \frac{1}{2}(\Pr(\varphi_u(x) = 1) - \Pr(\varphi_u(x) = 0)).$$

For a non-zero linear combination $\varphi_u(x)$, the bias is dependent on the input bias e as in the theorem 1 [16].

Theorem 1. Let f be a mapping from n-tuple binary vector to k-tuple binary vector, and let v be a k-tuple binary vector obtained by f. Let e be the input bit bias. Then, the bias $\Delta_u(e)$ of v is given as

$$\Delta_u(e) = \frac{1}{2^{n+1}} \sum_{v \in F_2^n} (2e)^{w(v)} (-1)^{1+w(v)} \hat{\varphi}_u(v),$$

where $w(v)$ is the Hamming weight of vector v.

Corollary 1. Let e be the bias of X. The bias of linear Boolean function $f(x)=a \cdot x$ is given as

$$\Delta_1(e) = -\frac{1}{2}(-2e)^{w(a)}.$$

Proof. For $u=1$, since $\varphi_1(x) = a \cdot x$, the Walsh transform $\hat{\varphi}_1(v)$ of $\varphi_1(x)$ can be derived as

$$\hat{\varphi}_1(v) = \sum_{x \in F_2^n} (-1)^{(a+v) \cdot x} = \begin{cases} 2^n, & \text{for } a = -v, \\ 0, & \text{for } a \neq -v. \end{cases}$$

1) Actually, Dichtl proposed the H, H_2, and H_3 correctors [21]. However, we use the notation H to denote the Shannon entropy. In order to avoid confusion, the H, H_2, and H_3 correctors are denoted using the D_1, D_2, and D_3 correctors, respectively.

Thus, from theorem 1, the bias $\Delta_1(e)$ of the linear Boolean function is given as

$$\Delta_1(e) = \frac{2^n}{2^{n+1}}(2e)^{w(a)}(-1)^{1+w(a)} = 2^{w(a)-1}e^{w(a)}(-1)^{1+w(a)}. \quad \square$$

Because each post-processed output bit is generated by a linear combination of raw TRNG output, the probabilities of 0 and 1 can be directly determined by using the Hamming weight $w(a)$ of the linear combination function. Therefore, magnitudes of bias of Dichtl's correctors, D_1, D_2, and D_3 are given as 2^2e^3, 2^3e^4, and 2^4e^5.

The next theorem shows that the entropy of i.i.d. random variable X is always less than or equal to that of Y, which is a linear combination of X's.

Theorem 2. Let X be an i.i.d. random variable, and let Y be a linear combination of X's. Then, we have $H(Y) \geq H(X)$.

Proof. By the definition of Shannon entropy $H(X)$ of X, we have

$$\begin{aligned} H(X) &= -\left(\frac{1}{2}+e\right)\log_2\left(\frac{1}{2}+e\right)-\left(\frac{1}{2}-e\right)\log_2\left(\frac{1}{2}-e\right) \\ &= -\frac{1}{2}\log_2\left(\frac{1-4e^2}{4}\right)-e[\log_2(1+2e)-\log_2(1-2e)] \\ &= 1+\frac{1}{2\ln 2}\left(4e^2+\frac{16e^4}{2}+\cdots\right) \\ &\quad +\frac{e}{\ln 2}\left(-2e-\frac{8e^3}{3}+\cdots-2e-\frac{8e^3}{3}-\cdots\right) \\ &= 1-\frac{2e^2}{\ln 2}-\frac{4e^4}{3\ln 2}-\frac{32e^6}{15\ln 2}-\cdots. \end{aligned}$$

In the third equality, the Taylor series of $-\ln(1-x) = x+\frac{x^2}{2}+\frac{x^3}{3}+\cdots$ is used.

From corollary 1, the bias of Y, a linear combination of X, is given as $\Delta_1(e) = -\frac{1}{2}(-2e)^{w(a)}$. Therefore, $H(Y)$ is given as

$$H(Y) = 1-\frac{2\Delta_1(e)^2}{\ln 2}-\frac{4\Delta_1(e)^4}{3\ln 2} = 1-\frac{2}{\ln 2}\left[2^{2w(a)-2}e^{2w(a)}+\cdots\right].$$

Because $0 \leq e \leq 1/2$, $H(Y) \geq H(X)$ always holds. The equality holds for the case of $a=1$, that is, $Y=X$. $\square$

From theorem 2, we can straightforwardly obtain the following corollary, which indicates that a higher Hamming weight codeword has a higher entropy than a lower Hamming weight codeword.

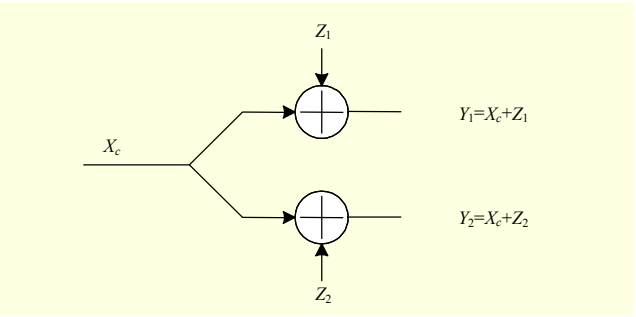

Fig. 1. Broadcasting model for two linear combinations of raw random input.

Corollary 2. Let Y_n be a linear combination of n X's. Then, we have $H(Y_n) \geq H(Y_{n-1})$.

2. Mutual Information between Adjacent Output Bits

Multi-bit entropy is as interesting as single bit entropy. To estimate the multi-bit entropy of the post-processed output, we derive the upper bound of mutual information between two output bits.

First, we will briefly introduce some basic definitions of information theory (For more information, see [23]). The entropy $H(X)$ of a discrete random variable X is defined as

$$H(X) = -\sum_{x\in X}\Pr(x)\log_2\Pr(x).$$

If $(X, Y)\sim\Pr(x, y)$, the conditional entropy $H(Y|X)$ is defined as

$$H(Y|X) = -\sum_{x\in X}\sum_{y\in Y}\Pr(x,y)\log_2\Pr(y|x).$$

The mutual information between two random variables X and Y is defined as

$$\begin{aligned} I(X;Y) &= -\sum_{x\in X}\sum_{y\in Y}\Pr(x,y)\log_2\frac{\Pr(x,y)}{\Pr(x)\Pr(y)} \\ &= H(X)-H(X|Y) = H(Y)-H(Y|X). \end{aligned}$$

Let Y_1 and Y_2 be two distinct linear combinations of w i.i.d. random variable X's. Among them, c X's are common and $w-c$ X's are distinct. To determine the mutual information between Y_1 and Y_2, we adopt the broadcast channel model shown in Fig. 1. In this model, two random variables, Y_1 and Y_2, can be represented as two received signals:

$$Y_1 = X_c + Z_1,$$
$$Y_2 = X_c + Z_2.$$

In this representation, a linear combination of common X's, X_c, can be considered a transmitted message in the channel, and linear combinations of distinct X's, Z_1 and Z_2, are additive noises for each receiver. Then, we can derive the relationship between mutual information of random variables X_c, Y_1, and Y_2

as in the following lemma.

Lemma 1. The mutual information for the broadcasting model in Fig. 1 satisfies the following inequality:

$$I(X_c;Y_1)+I(X_c;Y_2)\geq I(Y_1;Y_2).$$

Proof. By the definition of the mutual information, we have the following equalities:

$$I(Y_1;Y_2)=H(Y_1)-H(Y_1|Y_2),$$
$$I(Y_1;Y_2|X_c)=H(Y_1|X_c)+H(Y_2|X_c)-H(Y_1,Y_2|X_c),$$
$$I(X_c;Y_1)=H(Y_1)-H(Y_1|X_c),$$
$$I(X_c;Y_2)=H(Y_2)-H(Y_2|X_c).$$

By adding the last two equations, we have

$$\begin{aligned}I(X_c;Y_1)+&1(X_c;Y_2)\\&=H(Y_1)+H(Y_2)-H(Y_1|X_c)-H(Y_2|X_c)\\&=H(Y_1)+H(Y_2)-H(Y_1,Y_2|X_c)-I(Y_1;Y_2|X_c)\\&\geq H(Y_1)+H(Y_2)-H(Y_1,Y_2)-I(Y_1;Y_2|X_c)\\&=I(Y_1;Y_2)-I(Y_1;Y_2|X_c).\end{aligned}$$

We also have

$$I(Y_1;Y_2|X_c)=I(X_c+Z_1;X_c+Z_2|X_c)=I(Z_1;Z_2|X_c).$$

Because Z_1, Z_2, and X_c are pair-wisely independent, we have

$$I(Z_1;Z_2|X_c)=I(Z_1;Z_2)=0.$$

Thus, we can conclude that

$$I(X_c;Y_1)+I(X_c;Y_2)\geq I(Y_1;Y_2).$$ □

Now, it is possible to calculate the mutual information $I(X_c;Y_1)$ as in the following lemma.

Lemma 2. The mutual information $I(X_c ; Y_i)$ in Fig. 1 can be determined as

$$I(X_c;Y_i)=h\left(\frac{1}{2}-\frac{1}{2}(-2e)^w\right)-h\left(\frac{1}{2}-\frac{1}{2}(-2e)^{w-c}\right),$$

where $h(x)=-x\log_2 x-(1-x)\log_2(1-x)$ and $i=0$ or 1.

Proof. By the properties of mutual information, we have

$$\begin{aligned}I(X_c;Y_i)&=H(Y_i)-H(Y_i|X_c)\\&=H(Y_i)-H(X_c+Z_i|X_c)=H(Y_i)-H(Z_i|X_c).\end{aligned}$$

Because Z_i and X_c are statistically independent, we have

$$I(X_c:Y_i)=H(Y_i)-H(Z_i).$$

From corollary 1,

$$\Pr(y_i=1)=\frac{1}{2}-\frac{1}{2}(-2e)^w \quad\text{and}\quad \Pr(z_i=1)=\frac{1}{2}-\frac{1}{2}(-2e)^{w-c}.$$

Therefore, we have

$$H(Y_i)=h(\Pr(y_i=1)) \quad\text{and}\quad H(Z_i)=h(\Pr(z_i=1)).$$

Then, the statement is proven. □

From lemmas 1 and 2, we can easily obtain the theorem 3.

Theorem 3. The mutual information $I(Y_1;Y_2)$ in Fig. 1 is bounded by

$$I(Y_1;Y_2)\leq 2h\left(\frac{1}{2}-\frac{1}{2}(-2e)^w\right)-2h\left(\frac{1}{2}-\frac{1}{2}(-2e)^{w-c}\right).$$

As seen in theorem 3, the larger c is compared with w, the larger the mutual information $I(Y_1;Y_2)$. The large mutual information means that the common information between Y_1 and Y_2 is large, and the output bits have some dependency on each other. Therefore, we can conclude that it is not desirable to have many common components in two linear combinations. On the other hand, to reduce the bias of a single bit, it is better to choose codewords with the largest possible Hamming weights. However, if we increase the Hamming weight of each row in matrix G, then it is likely to share too many common components between rows of the matrix. Therefore, there is a tradeoff between Hamming weight and common components in the linear combinations of output bits.

A linear code can provide a solution for that tradeoff since the minimum distance of the linear code makes it possible to separate each codeword by at least some specific distance. Therefore, if we choose codewords in the linear code with large Hamming weights, it is able to satisfy both contradictory conditions. In the following subsection, we describe the proposed method in detail.

3. Linear Corrector from Quadratic Residue Code

Since the QR code is a cyclic code [22], [24], the (17, 9, 5) binary QR code can be generated using the following generating polynomial:

$$g(x)=x^8+x^5+x^4+x^3+1.$$

The Hamming weight distribution of the (17, 9, 5) binary QR code generated by $g(x)$ is given in Table 1.

As seen in Table 1, even though the minimum distance of (17, 9, 5) QR code is 5, there are many codewords which have

Table 1. Hamming weight distribution of (17, 9, 5) QR code.

Hamming weight	0	5	6	7	8
# of codewords	1	34	68	68	85
Hamming weight	9	10	11	12	17
# of codewords	85	68	68	34	1

larger Hamming weights than the minimum distance. In this paper, we suggest a method to construct a post-processing function using codewords in the code with large Hamming weights, such as 9 and 12, for (17, 9, 5) QR code.

Because the quadratic residue code is a cyclic code, a cyclically shifted version of a codeword is also a codeword. Since the length of a codeword is 17, there are 17 codewords which can be generated by cyclic shift of a codeword except for all zero and all one codewords as in Table 1.

In Table 1, there are two types of codewords with Hamming weight 12, and each type has 17 codewords which can be generated by cyclic shifts. Then, we choose one of them and construct a generating matrix using 9 codewords of a selected type. For example,

$$G_{12} = \begin{pmatrix} 11011000110111111 \\ 11101100011011111 \\ \vdots \\ 01111111101100011 \end{pmatrix}.$$

For comparison, we additionally construct the following two correctors by using codewords with Hamming weights 5 and 9:

$$G_5 = \begin{pmatrix} 00000000100111001 \\ 10000000010011100 \\ \vdots \\ 00111001000000001 \end{pmatrix}$$

and

$$G_9 = \begin{pmatrix} 11010100100010111 \\ 11101010010001011 \\ \vdots \\ 00101111101010010 \end{pmatrix}.$$

In the next section, we show that the proposed corrector with Hamming weight 12 outperforms the other correctors. For TRNG raw output x, post-processed output y is generated by $y = Gx$.

IV. Simulation and Further Improvement

1. Performance Comparison with Previous Correctors

Various correctors, including the proposed method with Hamming weight 12, were applied to correct biased random sequences with the length of 140,000 bytes. In simulations, we used the linear congruential generator with parameter (a, m) = (16807, 2147483647) to produce random numbers with defined bias.

As the first result, the 1-bit Shannon entropy of post-processed random sequences is plotted in Fig. 2. In the simulation carried out to obtain the results shown in Fig. 2, 999 random sequences

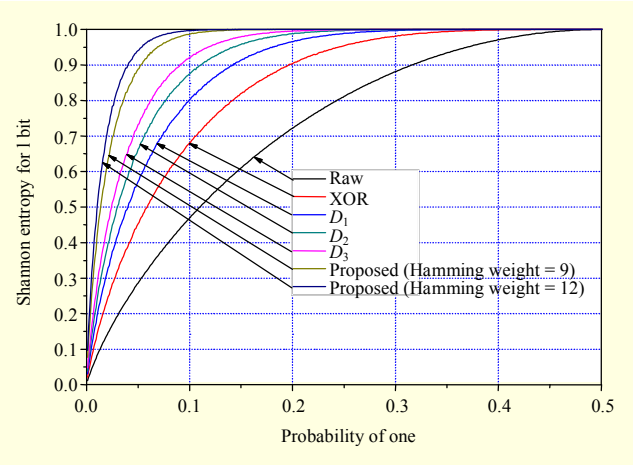

Fig. 2. Performance comparison with previous correctors.

with the length of 140,000 bytes were generated. Each random sequence has biases from 0.001 to 0.999 in terms of probability of one, Pr(1). In the AIS.31 standard [1], the tolerable 1-bit bias in TRNG output is given as a maximum of 0.0173, which can be represented in terms of 1-bit Shannon entropy 0.9991. In Fig. 2, the proposed corrector can be applied to the range (0.122, 0.878), while the applicable range of the D_3 corrector is (0.242, 0.758).

In summary, the proposed linear corrector shows better performance than any previous correctors. That is, the proposed corrector can be applied to a wider range of biased random sequences than others.

To obtain maximum performance, Dichtl proposed a heuristic construction with valuation 6 of biases. However, since implementation of Dichtl's corrector is quite complex, the lookup-table approach is the easiest way to implement it [21]. Although Lacharme used a non-linear corrector from (16, 256, 6) Nordstrom-Robinson nonlinear code [22] with valuation 6 of biases to decrease the difficulty of implementation, it is still more complex than the linear corrector. Also, he did not present an example to demonstrate a nonlinear corrector. Note that, for (17, 9, 5) QR code, it is possible to construct a linear corrector with valuation up to 12. It achieves better performance than others, such as the XOR corrector, D_1, D_2, and D_3 by Dichtl. Therefore, the proposed construction method for a linear corrector shows the best performance for fixing random output from a TRNG, and it can be implemented using simple architecture.

2. Entropy Test of AIS.31 Standard

To check multi-bit dependency, we apply an entropy test in AIS.31 [1] for each corrector. The entropy test proposed by Coron [25] is introduced as a method to estimate multi-bit

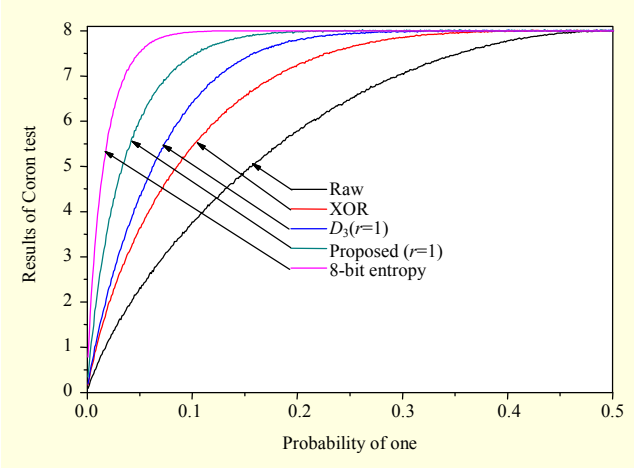

Fig. 3. Results of entropy test (Coron test) in AIS.31 standard.

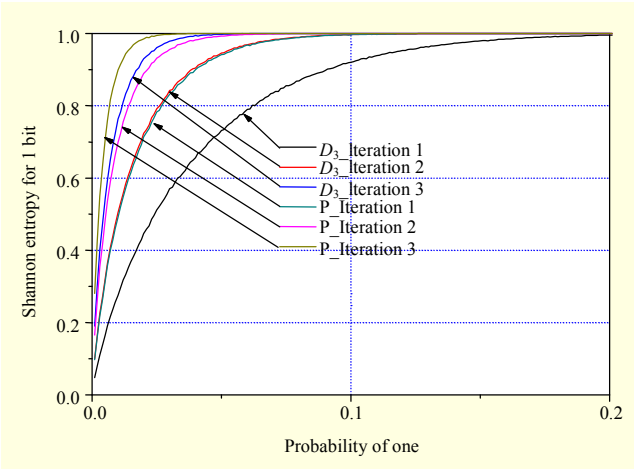

Fig. 4. Performance improvement of iterative post-processing. The proposed corrector is constructed from obtained with Hamming weight 12.

entropy in AIS.31. Although the entropy test is applied to the raw output of TRNG in AIS.31, it is applied to the post-processed output to check the dependency between adjacent output bits in this study. The 8-bit-based test is used as in AIS.31, and the acceptable threshold is 7.976 in that case. Since the threshold is changed according to the tolerable probability of false alarm which is dependent on security levels and applications, the specific value is not critical for the current purpose of checking the dependency between post-processed output bits.

Note that Coron test results are clearly different with 1-bit Shannon entropy multiplied by 8 because the dependency is not considered in 1-bit Shannon entropy. The test results are shown in Fig. 3.

In Fig. 3, 1-bit Shannon entropy multiplied by 8 is also presented for a reference. The apparent result for 1-bit Shannon entropy seems like the best one. However, that result should not be trusted as real entropy because the dependency is not considered in 1-bit Shannon entropy.

In AIS.31, the tolerable range for the Coron test is almost (0.086, 0.914) for the proposed corrector from a weight 12 codeword. For the D_3 corrector, the corresponding range is almost (0.286, 0.714). As a conclusion of the test, the proposed corrector can be applied to a larger range of biases than the other correctors.

3. Iterative Post-processing Scheme

To obtain further improvement of the random output, it is possible to apply the post-processed random output to the corrector iteratively [18]. For r iterations, the bias is reduced at the cost of decreasing the compression rate $(1/2)^r$. Although the iterating method can be applied to every corrector, the adopted corrector should be the best one to minimize the number of iterations because of the exponentially decreasing compression rate.

Figure 4 shows improved performance for both the D_3 corrector and the proposed corrector with Hamming weight 12. In the figure, D_3_iteration i and P_iteration i mean that D_3 and the proposed corrector with Hamming weight 12 are iteratively applied i times. Note that the performance of the proposed corrector (r=1) is almost the same as the performance of D_3 corrector (r=2). Even though the performance of the two correctors is similar, the compression rate of the proposed corrector is 0.53, while that of the D_3 corrector with r=2 is 0.25. This means that the proposed corrector has a throughput almost twice that of the D_3 corrector at the same level of performance.

V. Hardware Architecture for Proposed Correctors

The proposed corrector produces an improved 9-bit output from a 17-bit raw TRNG output. If TRNG generates 1 byte of output at a time, it is not difficult to adapt the input-output size by introducing input-output buffers.

Figure 5 shows a component of the hardware architecture for the proposed linear corrector using weight-12 codewords. The single XOR array can generate a 1-bit output from a 17-bit input, and this requires nine similar XOR arrays with distinct input connections. Because there are many overlapped components between rows of the generating matrix, it is possible to share the 2-input XORs or 3-input XORs for distinct output bits. This property enables us to optimize the hardware architecture for the linear corrector. However, for a conservative estimation, we assume that the 9-bit output is produced by using the XOR arrays shown in Fig. 5.

In Fig. 5, there are four 3-input XORs and three 2-input

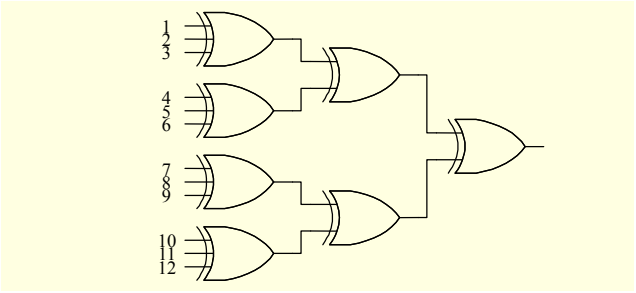

Fig. 5. Component of the proposed linear corrector (w=12).

XORs. To produce a 9-bit output, 36 3-input XORs and 27 2-input XORs are required. Normally, the XOR gate is larger than other gates, such as NAND or NOR. Even though the actual size is dependent on the hardware library and actual silicon implementation, we use the heuristic that the sizes of the 2-input XOR and 3-input XOR gates are 2.5 and 4.5 times bigger than those of the NAND gate, respectively.

Based on this assumption, the size of the proposed linear corrector using codewords with Hamming weight 12 is equivalent to that of 255 NAND gates. Therefore, the conservative conclusion is that the actual size is always less than 300 NAND gates.

Furthermore, it is possible to implement an iterative scheme in hardware. For an iterative scheme, we need a 17 (2:1) multiplexer to choose inputs between raw random data from TRNG and post-processed data in the previous cycles. Because the compression rate is equal to 0.53 in the 17-bit input and 9-bit output architecture, the 34-bit raw random data must be post-processed before iteration. Then, we finally obtain a 9-bit output from a 34-bit random input, which means that the compression rate is equal to 0.265. To store the 9-bit intermediate data, 9-bit registers are required. Because the (2:1) multiplexer and the 1-bit register require a silicon area equivalent to 2.5 to 4 NAND gates and 6.5 to 7.5 NAND gates, respectively, we need a maximum of 200 NAND gates for implementation of the iterative scheme. Overall, a hardware area equivalent to 500 NAND gates is enough to implement a 2-iteration linear corrector. This is an acceptable size for most systems with strong hardware restrictions such as smart cards.

VI. Concluding Remarks

In this paper, we proposed a method of constructing a linear corrector for TRNG output from a linear code. In our method, we select the codeword in a cyclic code with the largest Hamming weight, except for all-one codewords, and make a generator matrix via the cyclic shift of the chosen codeword. As a result, it is possible to construct a linear corrector which overcomes the minimum distance bound, which is discussed in previous research [16]. We showed that the performance of the proposed corrector is better than that of previous ones, in that a wider range of bias can be corrected by the proposed scheme.

In simulations of the 1-bit Shannon entropy and the 8-bit entropy test (Coron test) as in the AIS.31 standards, the proposed corrector outperformed other correctors. Moreover, the iterative scheme can improve post-processing at the cost of the compression rate. The proposed method is advantageous depending on the compression rate or the number of iterations.

Finally, we discussed the possible hardware architecture of the proposed scheme. Even in a conservative estimation, an implementation of the proposed corrector would require a small hardware size, which is suitable for embedded systems with strong hardware size restrictions.

References

[1] W. Killmann and W. Schindler, "A Proposal for Functionality Classes and Evaluation Methodology for True (Physical) Random Number Generators," AIS.31 Standard, 2001, URL: http://www. bsi.bund.de/ zertifiz/zert/interpr/trngk31e.pdf

[2] J.-S. No and P.V. Kumar, "A New Family of Binary Pseudorandom Sequences Having Optimal Periodic Correlation Properties and Large Linear Span," *IEEE Trans. Inf. Theory*, vol. IT-35, no. 2, Mar. 1989, pp. 371-379.

[3] O. Farooq and S. Datta, "Signal-Dependent Chaotic-State-Modulated Digital Secure Communication," *ETRI J.*, vol. 28, no. 2, Apr. 2006, pp. 250-252.

[4] Y.S. Kim et al., "New Constructions of *p*-ary Bent Sequences," *IEICE Trans. Fundamentals*, vol. E87-A no. 2, Feb. 2004, pp. 489-494.

[5] M. Bucci and R. Luzzi, "Design of Testable Random Bit Generators," *CHES 2005*, *LNCS*, vol. 3659, 2005, pp. 147-156.

[6] J.D. Golic, "New Methods for Digital Generation and Postprocessing of Random Data," *IEEE Trans. Computers*, vol. 55, no. 10, 2006, pp. 1217-1229.

[7] B. Sunar, W. Martin, and D. Stinson, "A Provably Secure True Random Number Generator with Built-In Tolerance to Active Attacks," *IEEE Trans. Computers*, vol. 56, no. 1, 2007, pp. 109-119.

[8] M. Dichtl and J. Golic, "High-Speed True Random Number Generation with Logic Gates Only," *CHES 2007*, *LNCS*, vol. 4727, 2007, pp. 45-62.

[9] I. Vasyltsov et al., "Fast Digital TRNG Based on Metastable Ring Oscillator," *CHES 2008*, *LNCS*, vol. 5154, 2008, pp. 164-180.

[10] E. Trichina et al., "Supplemental Cryptographic Hardware for Smart Cards," *IEEE Micro*., vol. 21, no. 6, 2001, pp. 26-35.

[11] W. Kim et al., "A Platform-Based SoC Design of a 32-Bit Smart Card," *ETRI J.*, vol. 25, no. 6, Dec. 2003, pp. 510-516.

[12] FIPS PUB 140-1: *Security Requirements for Cryptographic*

Modules, 1994.

[13] FIPS PUB 140-2: *Security Requirements for Cryptographic Modules*, 2001.

[14] W. Schindler and W. Killmann, "Evaluation Criteria for True (Physical) Random Number Generators Used in Cryptographic Applications," *CHES 2002*, *LNCS*, vol. 2523, 2003, pp. 431-449.

[15] Y.-S. Kim and I. Vasyltsov, "New Methods for Efficient Online Test of TRNG," *Samsung Journal of Innovative Technology, Communication & Network Technology*, vol. 4, no. 1, Feb. 2008, pp. 117-131.

[16] P. Lacharme, "Post-processing Functions for a Biased Physical Random Number Generator," *FSE 2008*, *LNCS 5086*, 2008, pp. 334-342.

[17] J. von Neumann, "Various Techniques for Use in Connection with Random Digits," *Von Neumann's Collected Works*, London: Pergamon, 1963, pp. 768-770.

[18] Y. Peres, "Iterating von Neumann's Procedure for Extracting Random Bits," *Annals of Statistics*, vol. 20, no. 1, 1992, pp. 590-597.

[19] A. Juels et al., "How to Turn Loaded Dice into Fair Coins," *IEEE Trans. Inf. Theory*, vol. 46, no. 3, 2000, pp. 911-921.

[20] S. Markovski, D. Gligoroski, and L. Kocarev, "Unbiased Random Sequences from Quasigroup String Transformations," *FSE 2005*, LNCS, vol. 3557, 2005, pp. 163-180.

[21] M. Dichtl, "Bad and Good Ways of Post-processing Biased Physical Random Numbers," *FSE 2007*, *LNCS 4593*, 2007, pp. 137-152.

[22] F.J. Mac Williams and N.J.A Sloane, *The Theory of Error Correcting Codes*, Amsterdam: North-Holland Pub., 1977.

[23] T.M. Cover and J.A. Thomas, *Elements of Information Theory,* 2nd ed., Hoboken, New Jersey: John Wiley and Sons, 2006.

[24] T.K. Truong, Y. Chang, and C.D. Lee, "The Weight Distributions of Some Binary Quadratic Residue Codes," *IEEE Trans. Inf. Theory*, vol. 51, no. 5, May 2005, pp. 1776-1782.

[25] J.-S. Coron, "On the Security of Random Source," *PKC'99*, *LNCS*, vol. 1560, 1999, pp. 29-42.

Young-Sik Kim received the BS, MS, and PhD degrees in electrical engineering and computer science from Seoul National University, Seoul, Korea, in 2001, 2003, and 2007, respectively. Since March 2007, he has been a senior engineer with Samsung Electronics, Co., Ltd., Korea. His research interests include secure implementation of crypto systems, pseudo/true random number generation, online tests/post-processing of TRNG, security of wireless communications, combinatorics, sequences, and information theory.

Ji-Woong Jang received the BS, MS, and PhD degrees in electronic engineering and computer science from Seoul National University, Seoul, Korea, in 2000, 2002, and 2006, respectively. He has been a senior engineer with Samsung Electronics since 2006. His research interests include pseudo-noise sequences, difference sets, cryptography, error correcting codes, and wireless communication systems.

Dae-Woon Lim received the BS and MS degrees in electrical engineering from KAIST in 1994 and 1997, respectively. In 2006, he received the PhD degree in electrical engineering and computer science from Seoul National University. From 1997 to 2002, he was with LG Industrial Systems as a senior research engineer, where he developed a recognition algorithm, a real-time tracking algorithm, and an electric toll collection system. He is currently a professor with the Department of Information and Communication Engineering of Dongguk University, Seoul, Korea. His research interests are in the areas of signal processing, wireless communications, and channel coding.