**May 21, 2013; second version. http://vixra.org/abs/1305.0068 http://arxiv.org/abs/1305.3248**

# PHYSICAL UNCLONEABLE FUNCTION HARDWARE KEYS UTILIZING KIRCHHOFF-LAW-JOHNSON-NOISE SECURE KEY EXCHANGE AND NOISE-BASED LOGIC

LASZLO B. KISH [(1)], CHIMAN KWAN [(2)]

[(1)] *Texas A&M University, Department of Electrical and Computer Engineering, College Station, TX 77843-3128, USA*

[(2)] *Signal Processing Inc, 13619 Valley Oak Circle, Rockville, MD 20850, USA*

**Abstract.** Weak physical uncloneable function (WPUF) encryption key means that the manufacturer of the hardware can clone the key but anybody else is unable to so that. Strong physical uncloneable function (SPUF) encryption key means that even the manufacturer of the hardware is unable to clone the key. In this paper, first we introduce a "ultra"-strong PUF with intrinsic dynamical randomness, which is not only not cloneable but it also gets renewed to an independent key (with fresh randomness) during each use via the unconditionally secure key exchange. The solution utilizes the Kirchhoff-law-Johnson-noise (KLJN) method for dynamical key renewal and a one-time-pad secure key for the challenge/response process. The secure key is stored in a flash memory on the chip to provide tamper-resistance and non-volatile storage with zero power requirements in standby mode. Simplified PUF keys are shown: a strong PUF utilizing KLJN protocol during the first run and noise-based logic (NBL) hyperspace vector string verification method for the challenge/response during the rest of its life or until it is re-initialized. Finally, the simplest PUF utilizes NBL without KLJN thus it can be cloned by the manufacturer but not by anybody else.

## 1. Introduction

Intellectual Property (IP) protection of hardware designs such as FPGA and IC has been becoming important in recent years. It has been estimated that US$100 billion of global IT industry revenue has been lost due to counterfeiting [1]. Attackers can easily tap the bit stream containing hardware configuration information and hence clone the hardware system.

To protect IP of hardware systems, researchers developed a physical unclonable function (PUF) that is embodied in a physical structure and is easy to evaluate but hard to predict even given the exact manufacturing process that produced it [2]. Earliest research of PUF started in 1983 [3] and 1984 [4,5]. The term PUF (physical unclonable function) was coined in 2001 [6] and 2002 [7]. In [7], the first integrated PUF was produced in the same electrical circuit.

In PUF, a challenge $C$ and a response $R$, called a challenge-response pair (*CRP*) is needed. That is, a PUF is considered as a function that maps challenges to responses [1,2]. For PUF to work, there are 3 conditions [1]: (1) it is assumed that a response $R_i$ (to a challenge $C_i$) gives only a negligible amount of information on another response $R_j$ (to a different challenge $C_j$) with $i \neq j$; (2) without having the corresponding PUF at hand, it is impossible to come up with the response $R_i$ corresponding to a challenge $C_i$, except with negligible probability; and (3) it is assumed that PUFs are tamper evident.

Concerning the randomness they utilize, there are 2 types of PUFs. The first category uses explicitly-introduced randomness. Examples include optical PUF and coating PUF [2]. One key advantage of this type of PUF is that a much greater ability to distinguish devices from one another and have minimal environmental variations compared to PUFs that utilize intrinsic randomness. The second type is the PUF using *intrinsic randomness*. Typical examples include delay PUF, SRAM PUF, Butterfly PUF, Bistable

Ring PUF, and Magnetic PUF. Advantages of PUF include the following. First, there is no need for a dedicated power-source (e.g., a battery) to keep the key memory alive. Second, data remanence issues are eliminated. Third, additional physical security measures to protect the memory device are minimized.

However, it has been pointed out in [8] that all of the existing PUF in the literature still lack statistical uniqueness and robustness with respect to environmental conditions. In addition, it is concluded that the ability of the PUF to reliably and repeatedly generate a unique PUF of a minimum 256 bits in length still needs significant research. Furthermore, [9] points out that many PUFs, including standard Arbiter PUFs and Ring Oscillator PUFs of arbitrary sizes, and XOR Arbiter PUFs, Lightweight Secure PUFs, and Feed-Forward Arbiter PUFs of up to a given size and complexity, are all vulnerable to numerical modeling attacks.

PUF can be classified also according to its strength. *Weak* PUF key can be cloned by its maker but by anybody else. *Strong* PUF cannot be cloned even by its maker. However, even stronger PUF, ultra-strong ones can be realized, which we will show below.

In this paper, we will introduce an *ultra-strong*, a *strong* and a *weak* PUF hardware key. These systems use two tools of noise-based informatics: the Kirchhoff-law-Johnson-noise (KLJN) unconditionally secure key exchange method [9-14], and noise-based logic (NBL) [16-27].

In Section 2, we briefly describe the KLJN system [9-14] and the required version of NBL tool [26].

In Section 3, we introduce a novel PUF key with *intrinsic randomness*. It may be called "*ultra-strong*" PUF because not only it cannot be copied by its maker but during each use it is randomly changing by generating and sharing a new independent unconditional secret key with the lock thus, if a counterfeiter accidentally succeeds to make a clone (this has exponentially low probability), that clone will be useless after next time the legal user applies the PUF key. The generation and regeneration of the intrinsic randomness in the key utilizes the unconditionally secure key distribution based on the Kirchhoff-law-Johnson-Noise (KLJN) method [9-14], which offers comparable or higher security than quantum key exchange (QKD) schemes do [13]. To generate the challenge/response a one-time-pad is used, which gets renewed during each use of the PUF key.

In Section 4, we present a simplified version of the PUF key described above, which is still "strong", that is even the manufacturer is unable to clone it after its first use. The key refreshment is used only first time when the PUF key is applied to the lock. Then during the subsequent applications the secure key does not act as a one-time-pad anymore. The challenge-response system utilizes noise-based logic (NBL) [16-25] hyperspace vectors [19] and the string verification method [26] based on that.

In Section 5, we present the simplest PUF utilizing noise-based informatics: a key that can be cloned by the manufacturer (this means it is not a strong version). However, it cannot be cloned by anybody else. This system contains a secret key installed by the manufacturer. The challenge-response system utilizes the hyperspace vector [19] of noise-based logic (NBL) [16-25] and the string verification method [26] based on that.

## 2. Brief description of the tools: the KLJN secure key exchange and noise-based logic

### *2.1 The KLJN key exchange method with information theoretic (unconditional) security*

The Kirchhoff-law-Johnson-noise (KLJN) secure key exchange scheme [9-14] was proposed in 2005 [9,10]. It is a statistical/physical competitor to quantum key exchange and its security is based on Kirchhoff's Loop Law and the Fluctuation-Dissipation Theorem. More generally, it is founded on the Second Law of Thermodynamics, which indicates that the security of the ideal scheme is similarly strong as the impossibility to build a perpetual motion machine (of the second kind).

The basic KLJN system [9,13] is shown in Figure 1. Two identical pairs of resistors ($R_L$ and $R_H$) and two binary switches with corresponding states (representing bit values) are used, one at Alice's side and one at Bob's. Depending on the state of a switch, which is randomly selected at the beginning of each clock cycle, one of resistors (with its own Johnson noise generator) is connected to the cable. Thus there are four possible states in which the two binary switches can be in: *LL*, *LH*, *HL*, and *HH*. The two mixed states, *LH* and *HL*, have identical mean-square channel noise voltage $U_{ch}$ and current $I_{ch}$ amplitudes, which cannot be distinguished by the eavesdropper (Eve). However, Alice and Bob know not only the noise levels in the channel but also the resistance value of their own resistor connected to the line. So, they can logically deduce the state at the other side from the resultant of the resistance values (deduced from channel noise intensity) and from state of their own switch.

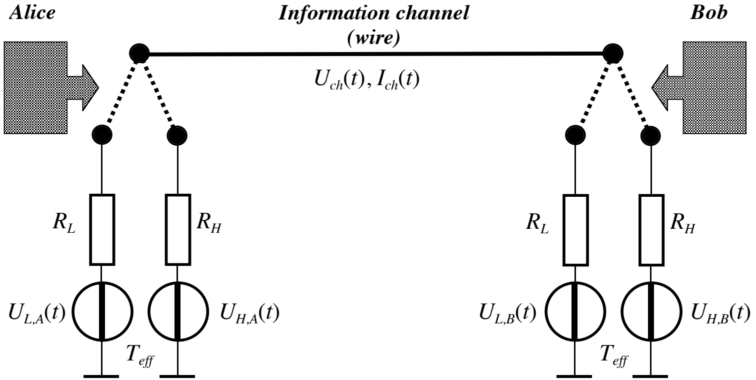

**Figure 1.** Basic circuitry of the KLJN secure key exchange system [9,10]. This simple version is not protected against active (invasive) attacks or attacks that utilize non-ideal features of the buildig elements.

This system, when ideal, offers unconditional (information theoretic) security against passive (listening) attacks only. To achieve unconditional security against active (invasive) attacks and/or attacks that utilize non-ideal features of the building elements, the following extra elements are essential [9,13,14]:

*a)* External noise generator to have stable noise temperature with high accuracy.

*b)* Filters to limit the frequency range to the measurement bandwidth and to the no-wave limit.

*c)* Measurement of the instantaneous voltage and current values at the two ends (by Alice and Bob) and to compare them via an authenticated public channel. When they differ, those situation may indicate invasive eavesdropping and those bits must be discarded.

While there were several attacks published against the KLJN system by utilizing non-ideal building elements; the above described defense method has been able to protect the system and maintain the unconditional security at the practically-perfect level [13]. Recently seven new KLJN schemes have been proposed which achieve the practically-perfect unconditional security with higher bit exchange rate.

The KLJN system has been proposed to provide unconditional security for hardware in computers, games and instruments [15].

### *2.2 Noise-based logic; its hyperspace; and the string verification method*

Noise-based logic [16-26] is a new class of deterministic multivalued logic schemes where the information carrier is a system of random orthogonal time functions, as discussed below. For example, the stochastic time functions can have continuum amplitude distribution; they can be random spikes; or random telegraph signals. The universality of the binary versions of these logic schemes is proven.

Here we focus on the random telegraph wave scheme [22,23,26] and briefly introduce it based on [26]. There is a set of a simple type of random telegraph waves (RTW), $R_i(t_j)$, with discrete amplitudes $\pm 1$ and discrete time, where the index $i$ stands for the $i$-th RTW and the index $j$ for the $j$-th clock cycle. At the beginning of each clock period, a given random telegraph wave takes 1 or -1 amplitude with 50% probability. There is no memory in the system, except that the chosen amplitude is held until the end of the clock period where a new random selection takes place. Thus, for the sake of simplicity, we drop the continuum time parameter in the notation for an RTW, and refer only to the amplitude of the *i*th RTW during the *j*th clock cycle by $R_i(j)$. In other words, and RTW can be generated by a discrete random number generator that generates +1 or -1 with 50% probability each.

The product of an arbitrary number of independently generated RTWs:

$$W_x(j) = \prod_{i=1}^{N} R_i(j) \tag{1}$$

is called a hyperspace vector, which is also an RTW with the same statistical properties, and it is orthogonal to the original RTWs or any other RTW generated independently, i.e.:

$$\left\langle W_x(j)R_k(j) \right\rangle = \left\langle R_k(j)\prod_{i=1}^{N} R_i(j) \right\rangle = 0 \quad , \tag{2}$$

where either $k \in \{1,...,N\}$ or $k > N$ , which means that $R_k$ is an RTW generated independently of $R_1, R_2, ..., R_N$ .

Such a $W_x(j)$ product string can efficiently be decomposed to its elements by the Stacho method [27] provided the reference RTW system used for its generation is known. When the reference system is not known, or the system of RTWs are known but it is unknown which bit values do they correspond to, the decomposition is exponentially difficult [27] because it requires brute force.

While, the product string cannot be decomposed without the knowledge of the reference RTW system, the difference of two product strings can be efficiently detected and it is called by the somewhat misleading name as *string verification method*. The string verification method utilizes the fact that, if two RTWs are generated independently, the probability $P(m)$ that their amplitudes are identical over *m* clock steps is

$$P(m) = 2^{-m} \tag{3}$$

To represent an *N* long bit string by such a product (hyperspace vector) we need 2*N* independently generated RTW signals to assign a different RTW to the *H* and *L* values of the *N* bits.

Suppose, both Alice and Bob have an *N*-bit long string and they would like to detect if one or more of their bit values are different [26]. Then they use the same publicly agreed set of 2*N* independent random number generators and assign each of them to the publicly agreed relevant bit and its value. Then using the *N* RTWs (random number generators) that represent the bit values in the string, they make the RTW product described by Eq. 1. Let as call Alice's product $W_A(t)$ and Bob's one $W_B(t)$ .

Suppose that the bit strings of Alice and Bob differ by at least a single bit value. Then at least one RTW in these products (see Eq. 1) is different.

Obviously, if the $W_A(t)$ and $W_B(t)$ products contain the same RTW elements, which means that the two strings are identical, the following relations hold:

$$W_A(t_j) - W_B(t_j) = 0 \tag{4}$$

$$W_A(t_j)W_B(t_j) = 1 \tag{5}$$

Thus, a single comparator device checking for non-zero values in the first case; or a multiplier and a comparator checking for negative values in the second case can verify if the assumption about identical strings have been violated [26].

In conclusion, if the strings are different, the generated hyperspace vectors will also be different or deviate in a short time. Thus the comparison of short sample of $W_A(t)$ and $W_B(t)$ is enough to detect any difference with high probability. The probability that the difference of the strings is not detected is given by Eq. 3. This error probability gets infinitesimally small very quickly. For example, to reach the theoretical error probability of appr. $10^{-25}$ of the logic gates in computer chips at idealistic conditions, the required cock steps of comparison *m* is only 83 (because $0.5^{83} \approx 10^{-25}$). Thus, Alice and Bob are able to detect that there is an arbitrary difference between their bit strings *of arbitrary length* of *N* by communicating only 83 bits [26].

Note, an analogous representation of RTWs and the string verification can be achieved by using 0 and 1 amplitudes in the RTW and applying the XOR logic operation instead of multiplication.

After the above preparations, the description of the three different PUF systems is straightforward.

## 3. Ultra-strong PUF hardware key with KLJN and one-time-pad

The system is illustrated in Fig. 2. Both the Key and the Lock are integrated on chips. The manufacturer prepares the PUF with a secure key, which is cloneable. When the user uses the key the first time (initialization), the Lock recognizes the key by this cloneable secure key and then the Lock and the Key performs a KLJN key exchange and generates a new key that is unknown to even the manufacturer. The new secure key is saved in a flash memory in the chip thus it is virtually impossible to access this information.

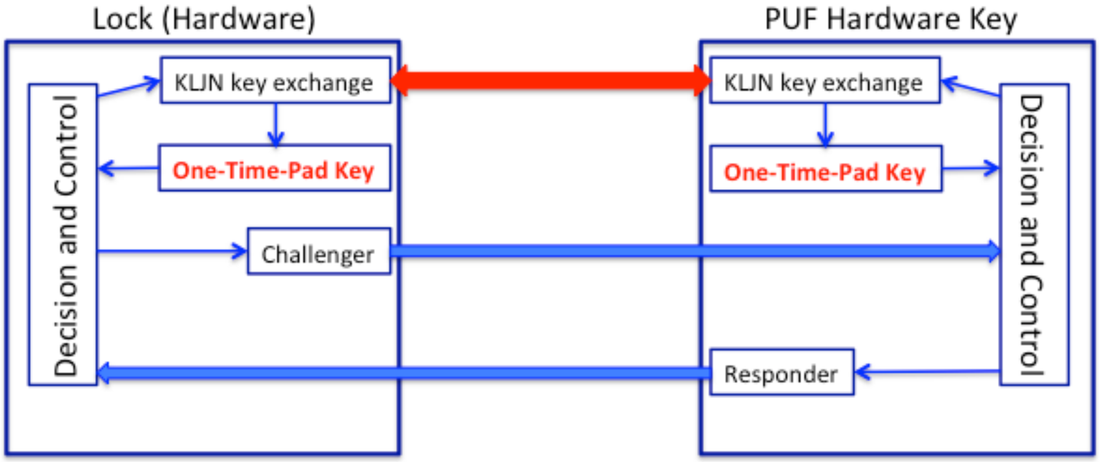

**Figure 2**. The Ultra-strong PUF hardware key with unconditionally secure KLJN key exchange and one-time pad.

After this initialization procedure, the uncloneable key is ready to be used and it is *ultra-strong* because at each new application of the key, the Lock and the Key again exchange a new independent unconditionally secure key. This PUF is ultra-strong because, if a counterfeiter accidentally succeeds to make a clone (this has exponentially low probability), that clone will be useless after next time the legal user applies the PUF key.

The challenge/response process is extremely simple: the Lock requests the PUF key to send the secure key they exchanged during last use of the key. It is important to note that the exchanged secure key is used as a *one-time pad*, which is the most secure way of secure communications, because during the next application of the PUF, already a new independent secure key will be used.

If a counterfeiter is trying to open the lock, his chance to succeed with a randomly generated $N$-bit long key is $2^{-N}$ , that is exponentially small. It is also important to note that there is no information about the next key and no modeling/simulation of system behavior can provide any information.

## 4. Strong PUF hardware key with KLJN and noise-based logic

Here we present a simplified version of the PUF key described above, which is still "strong", that is, even the manufacturer is unable to clone it after its first use, see Fig 3. The key refreshment is used only first time, during the initialization (which can be repeated if needed). Then during the subsequent applications the secure key cannot be used as a one-time-pad because the same secure key must be used. To provide and exponentially difficult job for the counterfeiter the challenge must proceed via a secure communication where the secure key is used in the cipher. The Lock generates a random string challenge, which then gets encrypted by the cipher and sent to the PUF key. After decrypting the challenge, the response system in the PUF key utilizes noise-based logic (NBL) and it generates and sends the product RTW described by Eq. 1, that corresponds to the given bit string. A counterfeiter has again $2^{-N}$ chance to succeed if he tries with his only available tool: brute force.

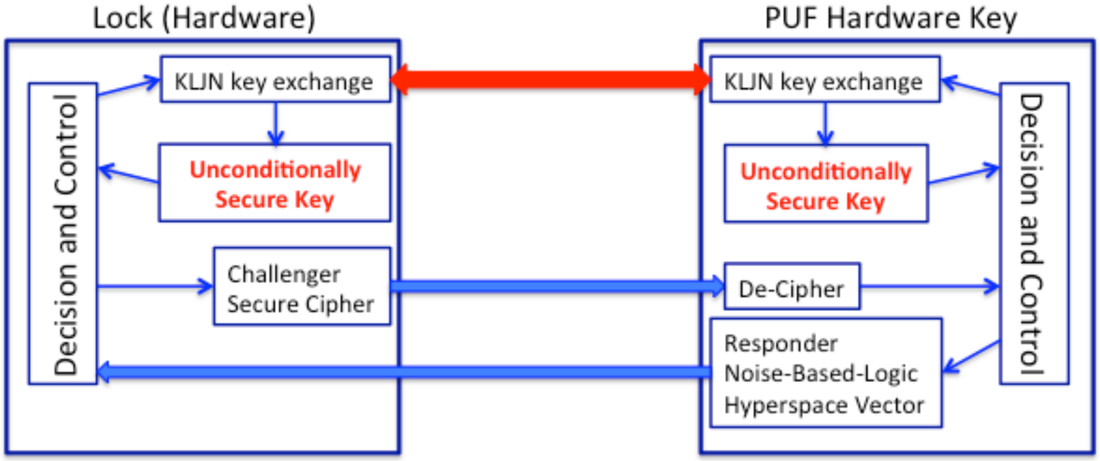

**Figure 3**. The Strong PUF hardware key with unconditionally secure KLJN key exchange and noise-based logic.

Then the Lock will use the string verification method described above: it also generates the required RTW product and uses Eqs. 4 and 5 to check if the sequence coming from the Key is satisfactory. If the Key sends $N$ clock steps of the product, the chance that a brute force counterfeiter escapes detection is again $2^{-N}$ .

The outline of the Responder is shown in Figure 4. The secure key bits shared by the unconditionally secure KLJN protocol determine for each bit if its pseudo-random-number generator A or B is assigned to its high or low value. All the $2N$ generators are independent and publicly known except their secret assignment.

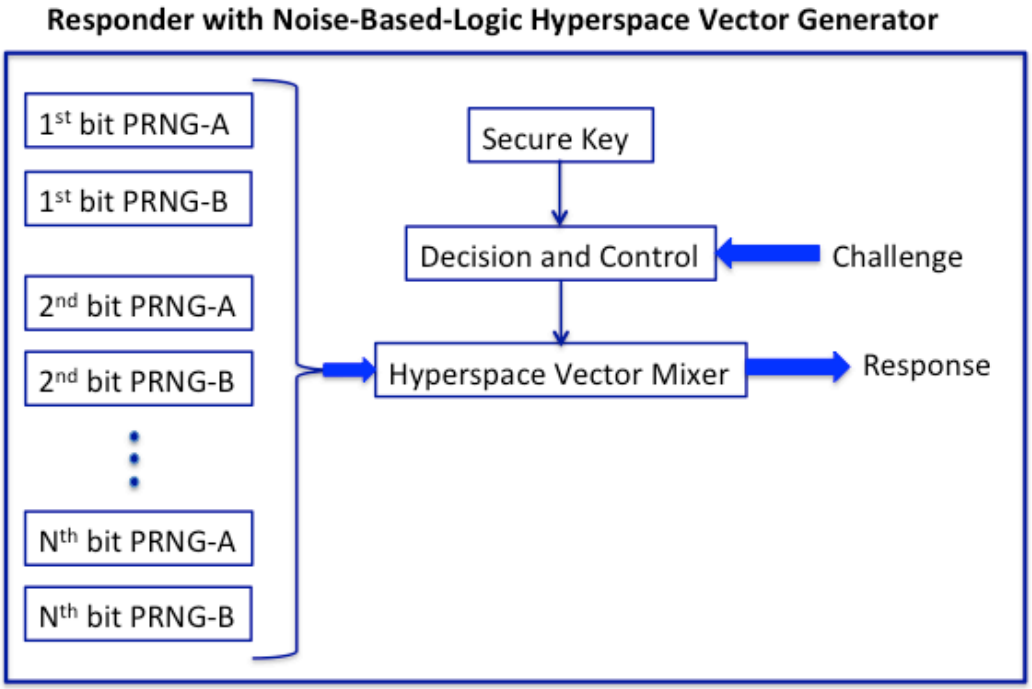

**Figure 4.** The Responder with RTW-based noise-based logic.

## 5. Simple PUF hardware key with noise-based logic

The simplest PUF utilizing noise-based informatics is shown in Fig. 5. It has a secure key that is implemented by the manufacturer thus it can be cloned (this means it is not a strong version) however this PUF cannot be cloned by anybody else. Similarly to the former system, the challenge-response system utilizes secure communication and the NBL scheme shown in Fig 4. and described in Section 2.2.

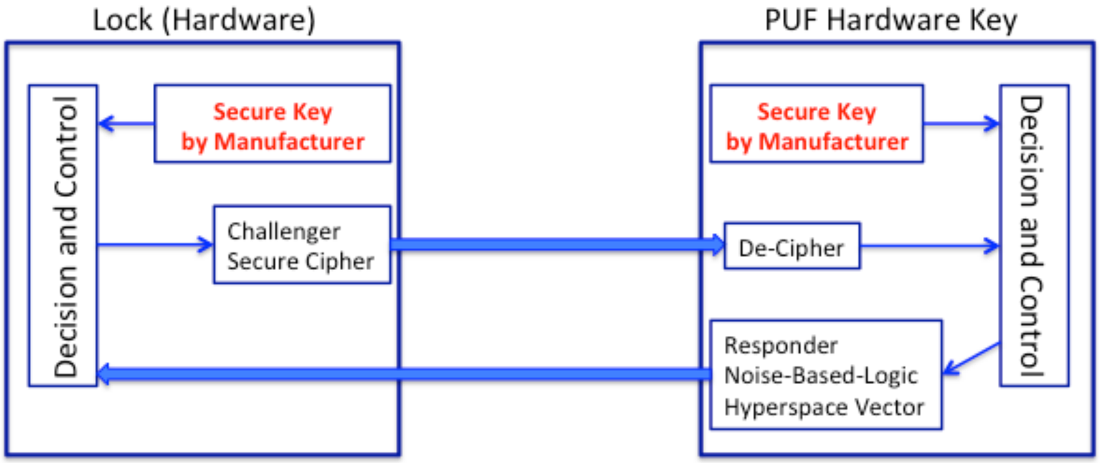

**Figure 5.** Simple PUF hardware key with noise-based logic

## 6. Conclusion

We showed three ways to realize physical uncloneable function (PUF) encryption keys where noise-based informatics is utilized. The key lengths can easily be much longer than 256 bits. All these systems can be integrated on a chip, which provides robustness against tampering and environmental effects.

## References

1. J. Guajardo, S. Kumar, G.-J. Schrijen, P. Tuyls, "Physical Unclonable Functions and Public-Key Crypto for FPGA IP Protection", In Field Programmable Logic and Applications, 2007. FPL 2007. International Conference on. Aug. 2007., pp. 189-195.
2. D.W. Bauder, "An anti-counterfeiting concept for currency systems", Research report PTK-11990. Sandia National Labs. Albuquerque, NM, 1983.
3. G. Simmons, "A system for verifying user identity and authorization at the point-of sale or access," *Cryptologia* **8** (1984) 1–21.
4. G. Simmons, "Identification of data, devices, documents and individuals," in IEEE International Carnahan Conference on Security Technology, 1991, pp. 197–218.
5. R. Pappu, B. Recht, J. Taylor, and N. Gershenfeld, "Physical one-way functions", *Science*, 297 (2002) 2026–2030. http://dx.doi.org/10.1126/science.1074376.
6. B. Gassend, D. Clarke, M. van Dijk and S. Devadas. Silicon Physical Random Functions. Proceedings of the Computer and Communications Security Conference, November 2002.
7. http://www.dodsbir.net/sitis/display_topic.asp?Bookmark=44320.
8. U. Rührmair et al., "Modeling Attacks on Physical Unclonable Functions," CCS'10, October 4–8, 2010, Chicago, Illinois, USA.
9. L.B. Kish, "Totally secure classical communication utilizing Johnson(-like) noise and Kirchhoff's law", *Phys. Lett. A* 352 (2006) 178-182.
10. A. Cho, "Simple noise may stymie spies without quantum weirdness", *Science* 309 (2005) 2148; http://www.ece.tamu.edu/~noise/news_files/science_secure.pdf.
11. D.J. Palmer, "Noise encryption keeps spooks out of the loop", *New Scientist* (2007) issue 2605 p.32; http://www.newscientist.com/article/mg19426055.300-noise-keeps-spooks-out-of-the-loop.html.
12. L.B. Kish, "Enhanced secure key exchange systems based on the Johnson-noise scheme", *Metrol. Meas. Syst.* **XX** (June 2013), in press,. http://arxiv.org/abs/1302.3901 , http://vixra.org/abs/1302.0055
13. R. Mingesz, L.B. Kish, Z. Gingl, C.G. Granqvist, H. Wen, F. Peper, T. Eubanks, G. Schmera, "Unconditional Security by the Laws of Classical Physics," *Metrol. Meas. Syst.* **XX** (2013), No. 1, pp. 3–16. open access: http://www.degruyter.com/view/j/mms.2013.20.issue-1/mms-2013-0001/mms-2013-0001.xml
14. L.B. Kish, "Protection against the man-in-the-middle-attack for the Kirchhoff-loop-Johnson(-like)-noise cipher and expansion by voltage-based security", *Fluct. Noise Lett.* **6** (2006) L57-L63.
15. L.B. Kish, O. Saidi, "Unconditionally secure computers, algorithms and hardware", *Fluct. Noise Lett.* **8** (2008) L95-L98.
16. L.B. Kish, "Noise-based logic: Binary, multi-valued, or fuzzy, with optional superposition of logic states", *Phys. Lett. A* **373** (2009) 911–918. http://arxiv.org/abs/0808.3162 .
17. J. Mullins, "Breaking the noise barrier", *New Scientist*, issue 2780 (29 September 2010); http://www.newscientist.com/article/mg20827801.500-breaking-the-noise-barrier. html?full=true.
18. H. Wen, L.B. Kish, " Noise based logic: why noise? A comparative study of the necessity of randomness out of orthogonality", *Fluct. Noise Lett.* **11** (2012) 1250021. http://arxiv.org/abs/1204.2545 .
19. L.B. Kish, S. Khatri, S. Sethuraman, "Noise-based logic hyperspace with the superposition of $2^N$ states in a single wire", *Phys. Lett. A* **373** (2009) 1928–1934. http://arxiv.org/abs/0901.3947 .
20. S.M. Bezrukov, L.B. Kish, "Deterministic multivalued logic scheme for information processing and routing in the brain", *Phys. Lett. A* **373** (2009) 2338-2342. http://arxiv.org/abs/0902.2033 .
21. Z. Gingl, S. Khatri, L.B. Kish, "Towards brain-inspired computing", *Fluct. Noise Lett.* **9** (2010) 403–412. http://arxiv.org/abs/1003.3932 .
22. L.B. Kish, S. Khatri, F. Peper, "Instantaneous noise-based logic", *Fluct. Noise Lett.* **9** (2010) 323-330. http://arxiv.org/abs/1004.2652 .
23. F. Peper, L.B. Kish, "Instantaneous, non-squeezed, noise-based logic", *Fluct. Noise Lett.* **10** (June 2011), open access: http://www.worldscinet.com/fnl/00/0001/open-access/S0219477511000521.pdf
24. H. Wen, L.B. Kish, A. Klappenecker, F. Peper, "New noise-based logic representations to avoid some problems with time complexity", *Fluct. Noise Lett.* **11** (2012), 1250003. http://arxiv.org/abs/1111.3859 .
25. H. Wen, L.B. Kish, A. Klappenecker, "Complex noise-bits and large-scale instantaneous parallel operations with low complexity", *Fluct. Noise Lett.* **12** (2013) 1350002. http://vixra.org/abs/1208.0226 .
26. L.B. Kish, S. Khatri, T. Horvath, "Computation using Noise-based Logic: Efficient String Verification over a Slow Communication Channel", *Eur. J. Phys. B*. **79** (2011) 85-90. http://arxiv.org/abs/1005.1560 .
27. L. Stacho, "Fast measurement of hyperspace vectors in noise-based logic", *Fluct. Noise Lett.* 11 (2012) 1250001.