CHECKING CORRECTNESS OF HARDWARE RNG ARCHITECTURE SPECIFICATIONS

Igor Fermevc,
Saša Adamović

[1]Singidunum University,
32 Danijelova Street, Belgrade, Serbia

Abstract:
In this paper we will show one possible implementation of hardware randomness generator. The device in question is based on widely available electronic components comprised of double analogue comparator operating as a free running oscillator and RISC microcontroller used for post processing. Finally, we incorporated an USB interface for communication with the device in order to acquire and evaluate its practical use in cryptography. Data generated by our device show very good randomness characteristics and have high entropy.

Key words:
random number generators, cryptography, registers, FRO, noise.

1. INTRODUCTION

Starting from the assumption that only physical processes that occur in nature can be fully unpredictable, undetermined or random, constant efforts are made in the field of modeling sources of randomness whose product will satisfy mathematically defined characteristics established in theory of probability and statistics [1]. By observing the characteristics of electronic noise, we can conclude that they comply with the basic principles of randomness, such as normal probability distribution of values and uncorrelated sampled values. With the help of surrounding electronic components, the first half of analogue comparator chip has been set to free running oscillation state in order to produce a wideband noise signal which, from mathematical point of view, has the proper characteristics as stated earlier. This noise signal is digitized with the use of the second half of comparator chip and is forwarded to input of microcontroller for further sampling and maintenance of randomness pool. Besides input signal sampling routine, the running microcontroller code implements the mechanisms for noise signal quality, randomness pool quality and protection for the user of randomness in case of potential attack or external electromagnetic interference. The communication with the user of randomness is accomplished by the use of USB/RS232 protocol translator chip, which is, from the aspect of cryptology, only part of the device we delegate the trust to.

Correspondence:
Igor Fermevc

e-mail:
igor.fermevc.12@singimail.rs

DOI: 10.15308/Sinteza-2016-179-182

2. RELATED WORK AND MOTIVATION

According to the type of device and randomness source, hardware based randomness generators can be roughly divided as shown in Figure 1.

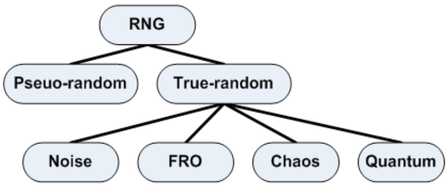

Fig. 1. Types of randomness generators

From the hardware randomness generators survey given in [2], we can assume the general block diagram of operation given in Figure 2.

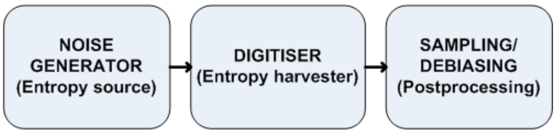
NOISE GENERATOR (Entropy source)
DIGITISER (Entropy harvester)
SAMPLING/ DEBIASING (Postprocessing)

Fig. 2. TRNG operation block diagram

Commercially available devices presented in this paper are using USB interface for communication with the user. This feature allows the ease of use and high portability, so we have accepted it as a good solution and implemented it in our device. Source of randomness of generators shown in Figure 3 is based on a physical process occuring in reverse biased PN junction [5]. Potential problem with this solution is high dependency of noise characteristics on physical properties of used semiconductors. To be more precise, besides the fact that these electronic components are working in an environment close to their breakdown, their physical properties deteriorate with time. Additional complexity in terms of design of these devices lies in the fact that USB interface provides low voltage, so one must implement power step-up circuit in order to produce good characteristics of noise signal from simple PN junction. Our goal was to simplify the design and increase reliability, so we opted for the solution proposed in [6] and based on the signal which is a product of free oscillating analogue comparator.

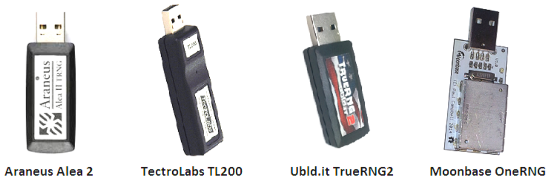

Figure 3. Commercially available randomness generators [8,9,10,11]

3. EXPERIMENTAL WORK AND EVALUATION

Hardware implementation of cryptography randomness generator

To create randomness generator the schematic diagram shown in Figure 4 was used. In order to keep the device footprint as small as possible, we opted for SMD electronic components. The circuit in Figure 4 contains a couple of segments which correspond to operation block diagram shown earlier. First, there is a double analogue comparator LM393 which functions as noise source and digitizer. Microcontroller Atmel Attiny85 is next in line and has the role of processing the digitized noise signal, randomness quality control, post processing and random pool maintenance, randomness data delivery and communication protocol. In the end, there is FTDI FT232RL chip which translates RS232 protocol to USB. During device development, we decided not to implement any kind of hardware protection against electromagnetic or other external interference in order to be able to test and confirm the protection mechanisms implemented in MCU code. As part of future

Model	Speed	Randomness source	Power consumption	Price
Alea II	100 kb/s	PN junction	max 250 mW	120 EUR
TL200	2 Mb/s	PN junction	max 100 mW	199 USD
True RNG2	350 kb/s	PN junction	No data	50 USD
One RNG	350 kb/s	PN junct. / RF noise	No data	50 USD
Our device	72 kb/s	FRO / El. noise	max 100 mW	15 EUR

Table 1. The most important characteristics of hardware based randomness generators we used for comparison with our device TRNG characteristics

work or in case of using this device in production environment, EMI shielding and power filter circuits are highly recommended.

Sinthesys of personal randomness source - TRNG

Double analogue comparator LM393 serves two purposes. The first part of this chip generates a noise signal, while the second part amplifies the noise signal and converts it to a digital signal with variable pulse length, which is further fed to appropriate input pin of MCU. MCU oversamples this "digital noise" and fast binary counter of MCU is incremented until there is a change of value in input signal. When the change occurs, LSB of counter register is stored in temporary register. The process is repeated until the whole byte is formed. The newly formed byte value is XOR-ed with previously formed and stored byte. The resulting value of XOR operation is then pushed into SRAM and this procedure constantly refreshes the randomness pool of 500KB. Due to the fact that the value of binary counter is related to input signal pulse width, this information is used to activate the security mechanism. If there is no change of value in input signal for a period of a couple of milliseconds, MCU code blocks the communication with the user of randomness, restarts the device and refreshes randomness pool prior to the establishment of a new communication with the user. Device operation test is done with the help of software application called XR232 [6] which allows us to request the certain amount of random data from generator. Data received from generator are stored in file and put to further statistical testing using NIST test battery described in [8]. Some of the results are shown in Table 2 and physical appearance of the device is shown in Figure 5. For comparative analysis, we have tested two sets of random data. One set is 16KB of data from our generator, and other set is 16KB sample collected from "random.org" service. The limitation in the amount of data is given by "random.org" service.

Test type	Data from our device (*p-value*)	Data from random.org (*p-value*)
Runs test	0.4390	0.0353
Serial test	0.3900	0.0870
Entropy (H)		
Mnobit:	0.99999670	0.99997561
Bigram:	0.99998785	0.99996795
Trigram:	0.99998438	0.99996382
Matrix 4x4:	0.99997163	0.99995819

Table 2. Comparative analysis of a couple of NIST test results (P > 0.01)

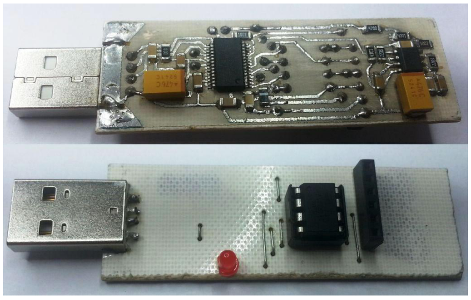

Figure 5. Physical appearance of randomness generator

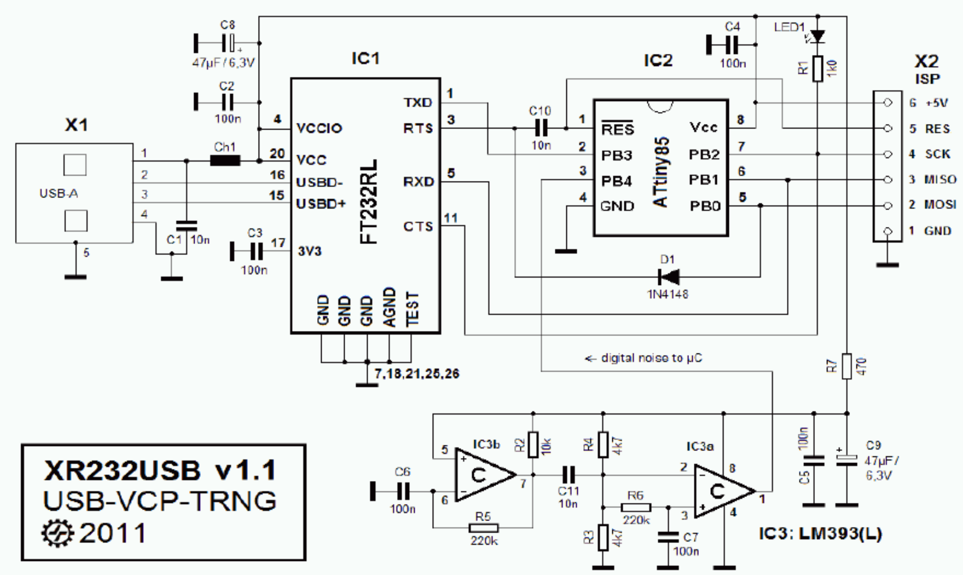

Figure 4. Circuit diagram, source [6]

Statistical testing and analysis of the results proved our initial hypothesis that our device conforms to all criteria concerning randomness characteristics. The speed of random data generation and delivery was tested by doing simple measurement of time needed for writing 1MB of random data in a file on user PC. Having repeated the process a number of times, for 1MB of data an average time of 114 seconds was obtained, so we can state that our device can deliver 72Kbs of random data. During the testing of our device, we have also simulated electromagnetic interference with the smart phone in close proximity to our device. Our device continues its normal operation until smart phone activates its GSM or WiFi transceiver. If disturbance is shorter than 2ms, our device recuperates without interrupting the communication with the user. Otherwise, the security mechanism is activated which prevents the delivery of corrupted, no random data to the user.

4. CONCLUSION

In this paper, we presented a theoretical concept for the use of electronic noise as a source of randomness and we explained one possible way of hardware implementation of this concept using widely available electronic components. In doing so, we experimentally proved good statistical properties of hardware random generators. Another contribution of our work is implementation of cryptographic element which warrants high level of security and trust. The performed testing confirmed high level of entropy and unpredictability of generated sequences. We've also verified our results using comparative analysis, in which our generator is compared with atmospheric noise. The device we've presented can be used for various purposes, mainly as an educational tool for different kinds of simulations and with minor modifications. It can also be used in real life cryptography applications with minor modifications.

REFERENCES

[1] Davenport W.Jr., Root W. (1987). *An introduction to the theory of random signals and noise*. Wiley-IEEE Press

[2] M. Stipcevic and C.K.Koc. (2014). *True Random Number Generators*. C.K.Koc. (Ed.), *Open Problems in Mathematics and Computational Science* (pp 275-315). Springer International Publishing

[3] J. F. Dynes, et. al. (2008). *A high speed, postprocessing free, quantum random number generator*. Applied Physics Letters, Vol. 93 (issue 3). DOI: 10.1063/1.2961000

[4] Wang Liao et. al. (2015). *Scalable Truly Random Number Generator*, Lecture Notes in Engineering and Computer Science: Proceedings of The World Congress on Engineering 2015, 1-3 July, 2015 (pp525-528), London, U.K.

[5] A.G. Chynoweth. (1958). *Electrical Breakdown in p-n Junctions*. Bell Laboratories Record, Vol. 36 No 2, February 1958 (pp 47-51), New York, U.S.A.

[6] JulienThomas. (2016, January). *XR232USB - True Random Numbers @ USB*. Retreived February 11, 2016, from: http://www.jtxp.org/tech/xr232usb_en.htm

[7] Charmain Kenny, (2005, April). *An Evaluation and Comparison of Random.org and Some Commonly Used Generators*. Retreived Febryary 12, 2016, from: https://www.random.org/analysis/Analysis2005.pdf

[8] Araneus Information Systems Oy, *Alea II*, Retreived February 11, 2016, from: http://www.araneus.fi/products/alea2/en/

[9] TectroLabs LLC. (Copyright 2012-2106), *TL200*, Retreived February 11, 2016, from: http://tectrolabs.com/tl200/

[10] Ubld.it (Copyright 2013) *TrueRNG2*, Retreived February 11, 2016, from: http://ubld.it/products/truerng-hardware-random-number-generator/

[11] Moonbase Otago (Copyright 2012-2014), *OneRNG*, Retreived: February 11, 2016, from:http://onerng.info/