Practical Randomness Measure tool

Boaz Shahar
Shahar.Boaz@gmail.com

Abstract. This report addresses the development of a pseudo random bit generator (PRBG) for constraint silicon devices. NIST.SP800-22 "Statistical test suite for Pseudo Random Generators" suggests a suite of tests that can confirm or deny the randomness of a given bit sequence. However, although providing a “pass / fail” criteria for the property of randomness of an arbitrary sequence, it is hard to get from the NIST suite the sense for the “level of randomness” for a given sequence, a measure that is sometimes required for the development process of PRBG. This post suggests a tool that can measure randomness, and therefore allows gradual changes in the PRBG algorithm, that helps trading power / time / area constraints versus quantifiable measure of the resulting randomness.

Keywords: Binomial distribution, commutative distribution function (CDF), PRBG, Bernoulli density

1 Introduction

Implementing standard techniques for the construction of Pseudo Random Bit Generators (PRBG) in low power or constraint devices is challenging: The cost of implementing a standard RBG, like in [3] for instance, may be quite high in terms of silicon area and time / power consumption.

Furthermore, a standard level RBG is not always required: For instance, the Common Criteria [4] security standard requires that only cryptographic functions that are exposed to the customer of a chip, or appear in chip’s security feature list for customer use, will be standard compliant cryptographic functions. Within a typical chip, there might be a lot of cryptographic mechanisms that are not directly used externally by the chip’s customers. The compliance of those cryptographic mechanisms is a somewhat weaker requirement, even for a pretty high Evaluation Assurance Level (EAL).

These facts, and especially the need for power and silicon area saving, sometimes motivates the development of an RBG mechanism with a pretty good randomness, and relatively low power / silicon area. However, this raises the question of what is “pretty good randomness”, and how algorithms for RBG may compare to each other.

The development of RBG algorithms is an iterative process composed of steps of changes in the algorithm, and measuring the effect of this change on the resulting pseudo random bit sequence. Once a change in an RBG algorithm is done, the developer would like to know whether this change improved or disproved the quality of the random sequence.

The difficulties with the NIST random test suite. Using the NIST random test suite for this purpose is not easy. The NIST test suite includes 15 independent tests. During the RBG development process, once a change in the RBG algorithm is done, it usually happens that some of the 15 tests give better randomness indication, while other tests give worse randomness indication. So it's hard to know whether the algorithm is improved or disproved.

TRNG vs. PRNG. TRNG can be constructed by adding an entropy source to a PRNG. This requires an entropy source, and a PRNG that can be initialized by this entropy source and can generate a Pseudo Random Bit Sequence with a cycle longer than the chip's life span. This report refers only to PRBG.

This report describes a practical criterion that was used to measure the amount of randomness of a sequence of bits that is generated by PRBG (2). This tool was used to develop an HW based low-power and low-area PRBG for constraint devices.

2 Background

Suppose a PRBG generates a pseudo random sequence of bits ***R***, of length n, $\{r_1,r_2...r_n\}$, where $r_i \in \{0,1\}$ for i=1...n. Then for ***R*** to be a “good” random sequence, the following requirements should be fulfilled: r_i ~ **Bernoulli**(½), that is, $p(r_i=1) = p(r_i=0) = ½$.
A “good” pseudo random sequence ***R*** is a finite sequence of binary random variables that take two values, 0 or 1, with equal probability. The Bernoulli random variables components of the sequence r_i should be identically distributed and statistically independent.

<u>The test sequence</u>
Let's look at a random bit sequence ***L (The Test Sequence)*** of length l, where l << n, such that $L=\{a_1,a_2...a_l\}$, where $a_i \in \{0,1\}$ for i=1...l, And let's define the random variable **x**, that counts the number of matches between the elements of ***L*** to the elements of the sequence under test, ***R,*** at location (i)***:***

$$x(i,l)=\sum_{j=1}^{j=l} \neg(r_{i+j} \oplus a_{i+j})$$

Where $x \oplus y$ is Boolean XOR of x and y and $\neg x$ indicates the inverse of x. The sigma is an algebraic sum. The random variable **x(i,l)** gets any integer value between 0 to l:

$$x(i,l) \in [0,l]$$

Its probability density for a real random sequence ***R*** can be calculated as follows:

$$Pr(x=k)=\binom{l}{k} p^k (1-p)^{l-k}$$

Where p is the probability that $r_{i+j}=a_j$. For ***R*** to be a pseudo random sequence, p=1/2, regardless of the distribution of a_j , and the probability density of ***x*** is independent of i and is given by:

$$Binomial(l,1/2): Pr(x=k)=\binom{l}{k}2^{-l}$$

That is, if ***R*** is a pseudo random sequence of bits, X ~ Binomial (l,1/2), that is, the random variable x has a binomial probability distribution with length l and p=1/2. The distribution of x is not dependent in the distribution of the bits of the sequence ***L***, but only in the length of ***L***: For all sequences ***L*** with the same length it has the same binomial distribution, if ***R*** is a pseudo random sequence.

2.1 *Statistical measure for randomness*

We are given a **PRNG**, that generates a sample sequence of bits, ***R***, that is hopefully very close to pseudo random sequence. We want to measure how far is this sample sequence from a pseudo random sequence. As we saw, the random variable x defined above a binary sequence ***R*** has a Binomial distribution with parameter ½, if it is constructed over good pseudo random sequence. Therefore, we can measure the level of randomness of the sample sequence generated of the PRNG under test by constructing the sample random variable $\hat{x}$ above the sequence under test, and calculate the distance of the sample distribution of $\hat{x}$ from the reference random variable x with **Binomial**(l, ½) distribution.

2.2 *Kolmogorov–Smirnov test*

One of the well known tests used to compare between sample probability distribution to reference probability distributions is the Kolmogorov Smirnov test (K-S test) [5]. The Kolmogorov-Smirnov statistics quantifies the distance between the **empirical cumulative distribution** function of the samples of a random variable $\hat{x}$, and the **cumulative distribution function** of the reference distribution function.

The K-S statistics for a cumulative distribution function F(x) is given by:

$$D_{K-S}=max_x|F_n(x)-F(x)|$$

Where $F_n(x)$ is the empirical cumulative distribution function, n is the number of samples, max is the maximal distance over the random variable x, and D_{K-S} is the Kolmogorov Smirnov distance.

The randomness measure tool described here is using the K-S test in order to measure the distance between the sample cumulative probability distribution $\hat{P}(\hat{x}=k)$ calculated over a sample pseudo random sequence generated by the PRNG mechanism under test of length n, to the **reference** ideal cumulative **Binomial**(l, ½) distribution.

3 Construction of a PRBG surveyor

A “surveyor” is an operator applied to an input bit sequence, that generates a measure of the distance of this input sequence from a pseudo random sequence. The surveyor applies the Kolmogorov-Smirnove (K-S) tests to the input sample sequence, and outputs the K-S distance of the input bit sequence to the reference distribution.

In our case, the reference distribution is the binomial probability distribution **Binomial**(l, ½), and the sample probability distribution is the measured sample distribution of $x(i,l) \in [0,l]$.

The sample probability distribution is given by:

$$\hat{P}(\hat{x}=k)=(\frac{1}{n})\sum_{i=1}^{n}(\hat{x}_i=k)_{k=0.\ldots l}$$

that counts all the locations $i, i=1\ldots n$ in the random sequence under test ***R*** for which the value of $x(i,l)=k$ and divide it by n, the length of the pseudo random sequence.

The sample cumulative distribution is given by:

$$Sample\,CDF(K,l)=\hat{P}(\hat{x}\leqslant K)=\sum_{j=0}^{j=K}\hat{P}(\hat{x}=j)=\sum_{j=0}^{K}(\frac{1}{n})\sum_{i=1}^{n}(\hat{x}_i=j)_{k=0.\ldots l}$$

The reference cumulative distribution is given by:

$$BinomialCDF(K,l)=\hat{P}(\hat{x}\leqslant K)=\sum_{j=0}^{j=K}P(x=j)=\sum_{j=0}^{j=K}\binom{l}{k}2^{-l}$$

The K-S distance is given by:

$$D_{K-S}(l)=max_K\left|SampleCDF(K,l)-BinomialCDF(K,l)_{k=0\ldots l}\right|$$

$D_{K-S}(l)$ expresses the maximum distance between the sample cumulative distribution function to the reference binomial cumulative distribution function over all values of K, for a given test sequence of length l bits.

The reference cumulative distribution function, *SampleCFF(K,l)*, depends on the sample sequence *L={a$_1$,a$_2$...a$_l$}*. As a result, $D_{K-S}(l)$ also depends on the sample sequence L={a$_1$,a$_2$...a$_l$}. Since we want to find the maximal distance from the ideal binary cumulative distribution function, we look for the L={a$_1$,a$_2$...a$_l$} that brings $D_{K-S}(l)$ to its maximum value. There are 2^l sequences *L={a$_1$,a$_2$...a$_l$}*, so we calculate $D_{K-S}(l)$ for all 2^l values of *L={a$_1$,a$_2$...a$_l$}*. The sequence *L={a$_1$,a$_2$...a$_l$}* for which $D_{K-S}(l)$ get a maximal value for a given pseudo random sequence {r$_1$,r$_2$...r$_n$} under test, is called the “Lackmus sequence of {r$_1$,r$_2$...r$_n$}”. The maximum value of $D_{K-S}(l)$ is marked by $D_{max}(l,R)$. It is a function of the sequence under test ***R,*** and the length of the sample sequence L={a$_1$,a$_2$...a$_l$}.

The Lackmus sequence associated with a given sequence under test ***R,*** and the K-S distance $D_{K-S}(l)$ that is associated with this sequence $D_{max}(l,R)$ **,** can be viewed as the best distinguisher of ***R*** from ideal pseudo random sequence of length l bits.

4 Refinement of the “measure of randomness”

The values of $D_{max}(l,R)$ are always between 0 to 1: It is 0 when the sequence under test ***R*** has exact Binomial distribution, reflecting the best possible randomness, and its 1 when the sequence under test ***R,*** is deterministic, e.g., all its bits are either 0 or 1, reflecting the lowest level of randomness. So, it makes sense to compare the level of randomness of various sequences under test ***R*** according to 1/ $D_{max}(l,R)$. Instead of using this quantity as a basis for comparison, we can use -log($D_{max}(l,R)$) as a measure, to improve the sensitivity of the measure for smaller values of $D_{max}(l,R)$.

In this way, the measure of randomness for a deterministic sequence is 0 (i.e., not random at all)

4.1 Normalization of the measure of randomness to AES-128

Any stream cipher can be used as a PRNG. Specifically, the AES cipher [5] in counter lock mode is a standard stream cipher, and therefore can be used as a pseudo random sequence generator. As AES is one of the most popular ciphers available and used in the industry, we defined the randomness of the sequence ***R,*** generated by AES_128(Key=0, IV=0, Counter) to be 100, for a sequence under test ***R*** of length 1Mbit, and calibrate the measures done on other PRNG according to this point. So the level of randomness of a general sequence under test ***R*** is given by:

$$Randomness\,Mark\,(RM)\,of\ sequence\ R=\frac{-\log[D_{max}(l,R)]\,x100}{\log[D_{max}(l,AES(0,0,CNT))]}$$

5 Expectations and Requirements

In order for the randomness mark to be useful, we require it to fulfill some properties:

5.1 Longer sample sequences

The measure of randomness of a good PRBG is expected to be monotonically increasing with sequence length. For a real random binary sequence, or for a very good pseudo random sample binary sequence, it is expected that the longer the sequence, its randomness mark is higher. This is because if the PRNG generates more bits, it should get closer to the ideal Bernoulli(1/2) distribution.

5.2 Cyclic Bit Sequences

Some PRNG may happen to generate a cyclic bit sequences, or a bit sequence with high correlation to previous bits. Since the statistics of cyclic sequence is also cyclic, in the case of cyclic sample bit sequence, when the length of the sequence exceeds the cycle of the sequence, the randomness mark is first drops, due to the fact that the test sequence ***L*** becomes longer, and finally it is asymptotically constant, and does not changed as the length of the sample bit sequence becomes larger. In some cases, it even decreases when the length of the test sequence, l, becomes larger, as more bits of the sequence under test ***R*** are tested simultaneously.

5.3 Comparison to NIST random tests

When the length of the test sequence ***L*** is l=1, the test is similar to the frequency test of the NIST suit. For any other l, the test considers the run length test of length l.

A fundamental difference between this test and the NIST test suit in[1] is that The approach is different. While NIST tests measure the sampled probability of a certain event, and compare it to a threshold, the approach presented here compares between two bit sequences and says who is harder to distinguish from random sequence.

The test takes into account statistical dependency, linear dependency, and cyclic correlation, as explained below.

5.4 Statistical dependency

Let's assume that a PRBG generates a pseudo random sequence of bits ***R***, of length n, $\{r_1,r_2...r_n\}$, where $r_i \in \{0,1\}$ for i=1...n. Then for ***R*** to be a "good" random sequence, the following requirements should be fulfilled: $\mathbf{r_i}$ ~ **Bernoulli**(½), that is, $p(r_i=1) = p(r_i=0) = ½$. If there is statistical dependency, the distribution function of $\mathbf{r_i}$ depends on the values of some other bits in the sequence, and the result will be that the distribution function of x(i,l) in:

$$x(i,l)=\sum_{j=1}^{j=l} \neg(r_{i+j} \oplus a_{i+j})$$

Will deviate from the expected binomial distribution of x(i,l):

$$Binomial(l,1/2): Pr(x=k)=\binom{l}{k}2^{-l}$$

which assumes that $\mathbf{r_i}$ are independent random Bernoulli variables. and this will result in a larger K-S distance. That is, the K-S distance is relative to the level of statistical dependency in the bit sequence ***R***.

6 Measure of randomness for various PRBG

The measure of randomness for the following PRBG are compared in this report:

Deterministic sequences A sequence of 1's and 0's.

The following primitive polynomial LFSR constructions used in order to generate PRBG with cycle $2^{degree}-1$:

LFSR-5, defined by the generator polynomial $G(x)=x^5+x^3+1$

LFSR-8, defined by the generator polynomial $G(x)=x^8+x^6+x^5+x+1$

LFSR-17, defined by the generator polynomial $G(x)=x^{17}+x^3+1$

LFSR-31, defined by the generator polynomial $G(x)=x^{31}+x^3+1$

LFSR-127, defined by the generator polynomial $G(x)=x^{127}+x+1$

The following ciphers:

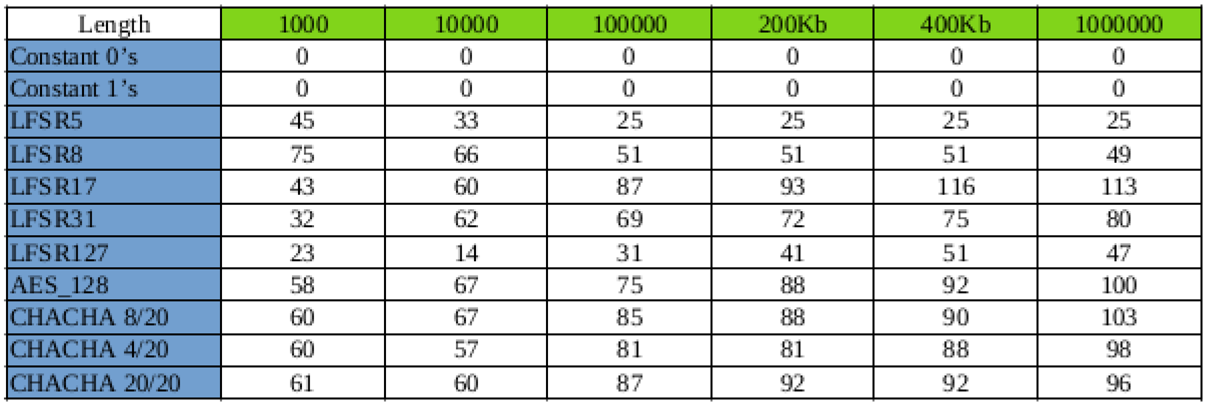
AES-128, AES in CNT block mode, with Key=0x0, and IV=0x0 | 32 bit counter

CHACHA 8/20, CHACHA cipher, 256 bit key of 0, IV 96 bit of 0, 32 bit counter, 8 rounds out of 20 rounds

CHACHA 4/20, CHACHA cipher, 256 bit key of 0, IV 96 bit of 0, 32 bit counter, 4 rounds out of 20 rounds

CHACHA 20/20, CHACHA cipher, 256 bit key of 0, IV 96 bit of 0, 32 bit counter, 20 rounds out of 20 round

Randomness Mark for various PRBG



Length	1000	10000	100000	200Kb	400Kb	1000000
Constant 0's	0	0	0	0	0	0
Constant 1's	0	0	0	0	0	0
LFSR5	45	33	25	25	25	25
LFSR8	75	66	51	51	51	49
LFSR17	43	60	87	93	116	113
LFSR31	32	62	69	72	75	80
LFSR127	23	14	31	41	51	47
AES_128	58	67	75	88	92	100
CHACHA 8/20	60	67	85	88	90	103
CHACHA 4/20	60	57	81	81	88	98
CHACHA 20/20	61	60	87	92	92	96

7 Observations

- The measure of randomness for sequences generated by AES-128(0,0,Counter) for the length of 1m bits is 100, as expected by construction of the measure of randomness.
- The measure of randomness for sequences of constant bit (all 0 or all 1) is 0. This is expected by construction of the measure of randomness.
- **LFSR5**
  - The randomness mark is pretty low (45-25)
  - The PRBG generated by LFSR5 has a cycle of $2^5-1=31$. The randomness mark fails from 45 to 33 for 10,000 bit sequence, and further to 25 for 100,000 bits sequence, as expected when the length of the sequence is much longer than the cycle
  - The measure of randomness for LFSR5, is asymptotically 25, and is not changed when the sequence length increased. This is expected as the cycle of this sequence is 31
- **LFSR8**
  - The randomness mark is medium (75-49).

    - The cycle is 255.
    - The mark for short sequence is 75, and then it decreases to 66, 51 and 49 as the seuence becomes longer, indicating statistical dependency
- **LFSR17**
    - The cycle is 131071
    - The randomness mark is improved up to 400Kb, then starts to decreases
    - LFSR17 presents a randomness mark better then this of AES, and this means that the sequence generated by LFSR17 is closer to random sequence then this of AES. Nevertheless, it start to decrease for large length, as opposed to AES that is always monotonically increasing.
- **LFSR31**
    - Cycle is $2^{31}-1$, pretty large (the same as the cycle of the PRBG that are based on ciphers that is using 32 bit counter block mode)
    - As expected, the randomness mark is monotonically increased with the length of the sequence under test ***R***, although the marks are lower than those of LFSR17
- **LFSR127**
    - Cycle is $2^{127}-1$, pretty high, larger than the cipher based PRNG. Nevertheless, its randomness mark is relatively low, in the range of LFSR8.
- **AES-128(0,0,CNT)**
    - AES with 128 bit key of value 0, and IV of 0 concatenated with 32 bit counter. Cycle is 2^{32}
    - The randomness mark of AES is monotonically increasing with the length of the sequence under test, as expected from PRNG with cycle higher than the length of the sequence under test ***R***. For length of 1e6 bits, the randomness mark is 100, as dictated by the way the randomness mark is calculated
- **CHACHA20**
    - In 2014, ChaCha20 was selected as one of the cipher suites for use in the Transport Layer Security (TLS) protocol by the Internet Engineering Task Force (IETF). Specifically, ChaCha20 was chosen for use with the Poly1305 message authentication code, which together provide a high level of security and performance in TLS.
    - CHACHA8/20 refers to the number of rounds used in the algorithm. Specifically, it means that the ChaCha20 algorithm is used with 8 rounds of processing for small devices with limited resources
    - CHACHA4/20 means that the ChaCha20 algorithm is used with 4 rounds of processing for small devices with limited resources
    - ChaCha 20 illustrates very nice behavior. Its randomness mark is consistently increasing when the length of ***R*** increased. Its grade in length of 1Mbit is 96, slightly less than AES

with grade of 100.

- Surprisingly, ChaCha 8/20 introduces higher randomness almost in all length, with grade of 103 at 1Mbit
- Chacha 4/20 introduces pretty good randomness with mark of 98 at 1Mbit

7.1 Cipher based PRBG vs. LFSR based PRBG

It can be seen from the table that the cipher based PRBG demonstrates higher randomness grade than the LFSR based PRBG, and consistent monotonic randomness grade that is increased when the length of the sequence increased. Exception is LFSR17, that demonstrates excellent randomness grade and characteristics, however, its PRBG is not monotonically increased with sequence length, an indication for statistical dependency as expected by its relatively low cycle.

8 Conclusions

A practical tool that used for the development of PRBG in constraint devices was introduced. For the generation of internal pseudo random sequences in a constraint silicon devices, either for a use as a streaming key, or other purposes, LFSR is the simplest and cheapest construction. LFSR17 is shown to produce excellent pseudo random binary sequence with the cost of just 17 flops and 3 XOR gates.

However, LFSR has a security disadvantage: once a partial pattern of the output stream of an LFSR is known, the entire bit sequence can be calculated. This can weaken the security of certain systems.

The security disadvantage of LFSR can be overcome by PRBG that are based on ciphers. All ciphers are inherently more complex and more costly than LFSR, but there are examples of very simple cipher implementations, like CHACHA 8/20 and CHACHA 4/20, that produces PRBG with very high randomness.

There are additional simplifications that can be done in CHACHA */20 that can simplify the implementation and decrease the cost of implementation significantly which could not be described in this report, with randomness grade higher than 96 (the randomness grade of ChaCha 20/20, which is a standard cipher [8]).

9 References

[1] “A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications “ NIST. SP800-22 https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-22r1a.pdf
[2] “A randomness test for block ciphers”, Vasillios Katos, ACM digital library 2005, https://dl.acm.org/doi/10.1016/j.amc.2003.12.122
[3] “ *Recommendation for Random Bit Generator (RBG) Constructions*”, NIST.SP800-90C
[4] “Common Criteria ”https://www.corsec.com/common-criteria/?gclid=CjwKCAiArbv_BRA8EiwAYGs23ANAQ8Yysf24LJ_9ot5D6CsHB41wbMs_WuZmA-eUL3dpovNgN3bKjRoCjY8QAvD_BwE
[5] Estimate of deviation between empirical distribution functions in two independent samples
NV Smirnov - Bulletin **Moscow University**, **1939**

[6] Advanced Encryption Standard (AES), NIST.FIPS 197,
https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf
[7] AES vs LFSR Based Test Pattern Generation: A Comparative Study, Marion Doulcier, Marie-Lise Flottes, Bruno Rouzeyre.
https://hal-lirmm.ccsd.cnrs.fr/lirmm-00138831/document
[8] ChaCha20 and Poly1305 for IETF Protocols, IETF RFC8439