Random number generators

Cryptography: course for master's degree in EDGE COMPUTING

Michał Melosik, PhD

Lecture outline

1. **Randomness**

2. **PRBG**

3. **TRBG**

4. **CSPRBG**

(*). QRBG - some practical aspects of cryptography

5. **Tools for testing randomness**

6. **Discussion**

How to define randomness?

Randomness

What does wikipedia have to say about it?

WIKIPEDIA
The Free Encyclopedia

Randomness

From Wikipedia, the free encyclopedia

"Random" redirects here. For other uses, see Random (disambiguation).

For a random Wikipedia article, see Special:Random. For information about Wikipedia's random article feature, see Wikipedia:Ra

In common usage, **randomness** is the apparent or actual lack of pattern or predictability in events.[1][2] A random sequence of events, symbols or steps often has no order and does not follow an intelligible pattern or combination. Individual random events are, by definition, unpredictable, but if the probability distribution is known, the frequency of different outcomes over repeated events (or "trials") is predictable.[note 1] For example, when throwing two dice, the outcome of any particular roll is unpredictable, but a sum of 7 will tend to occur twice as often as 4. In this view, randomness is not haphazardness; it is a measure of uncertainty of an outcome. Randomness applies to concepts of chance, probability, and information entropy.

Randomness

What does wikipedia have to say about it?

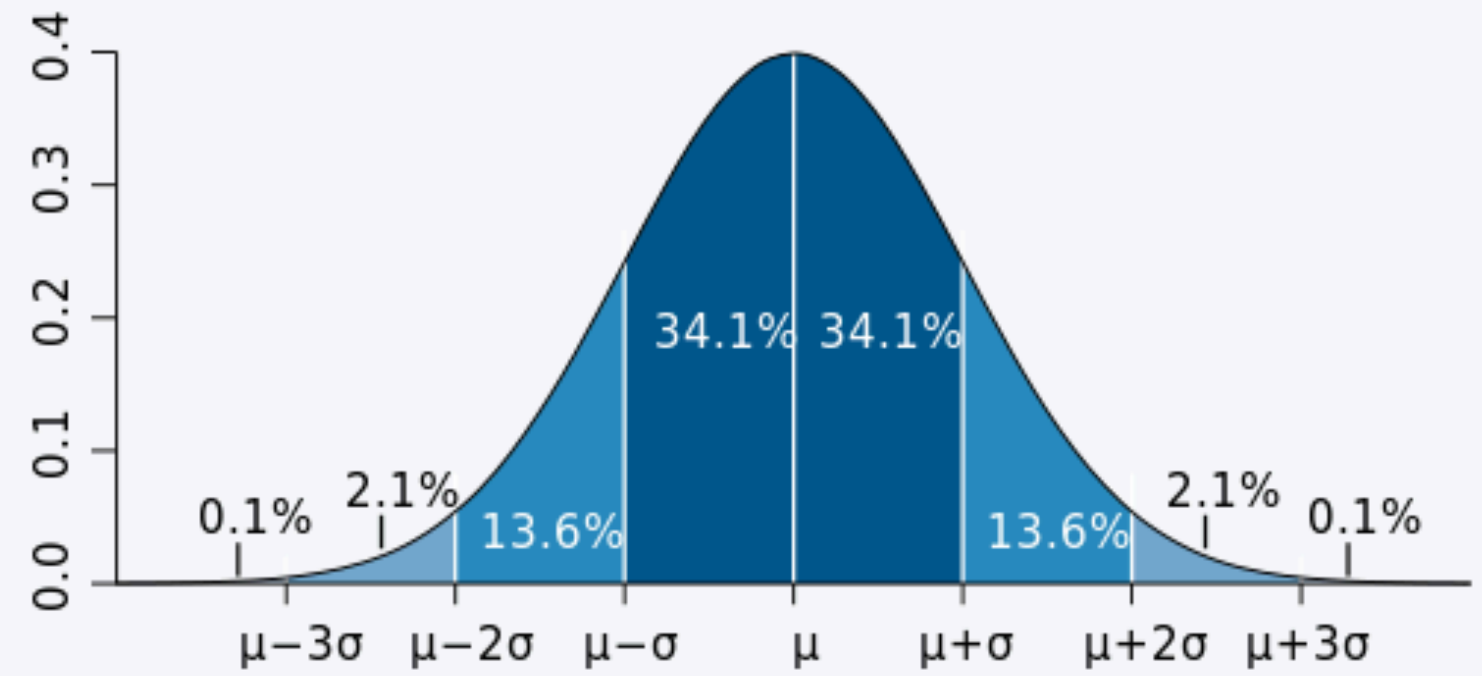
In science

Many scientific fields are concerned with randomness:

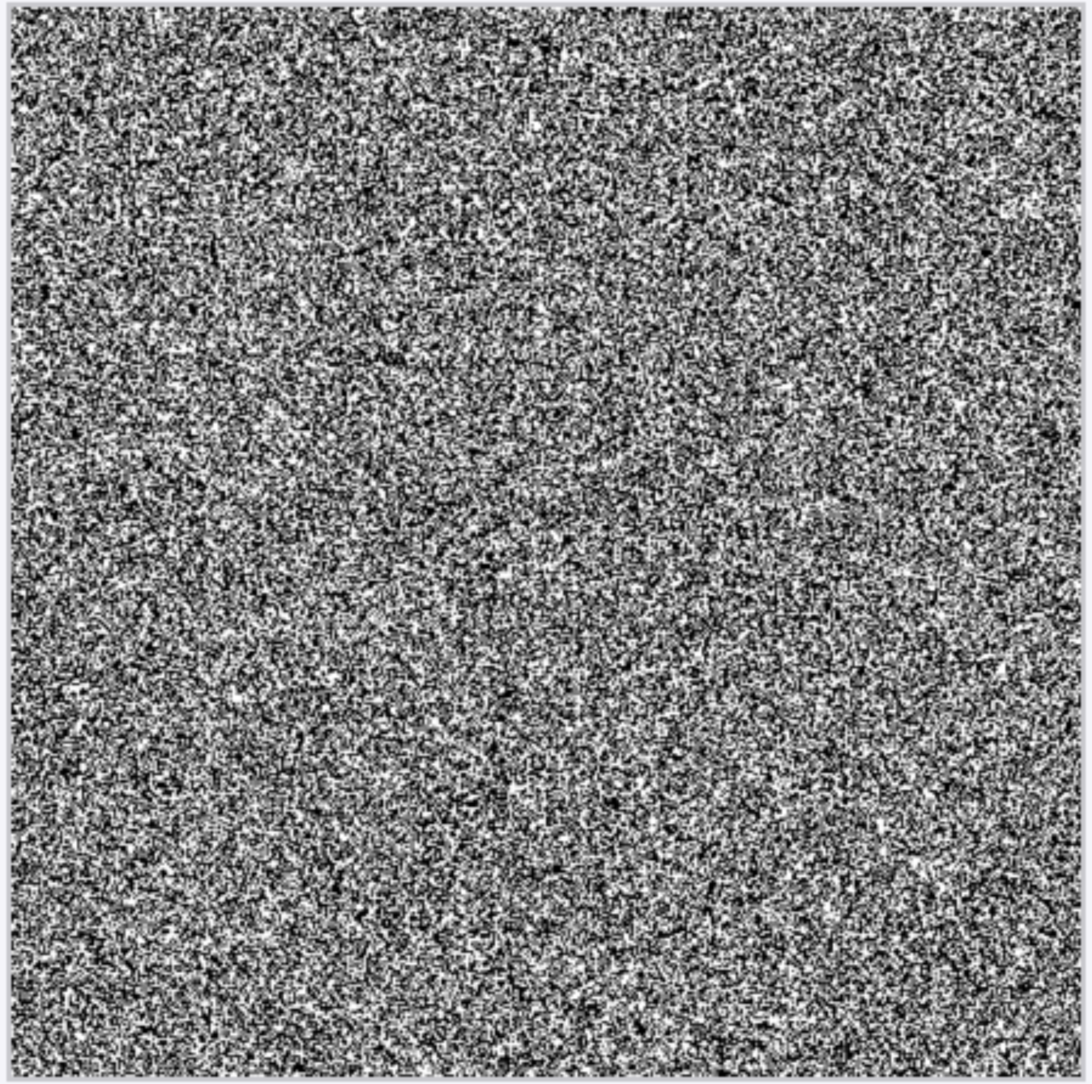
- Algorithmic probability
- Chaos theory
- Cryptography
- Game theory
- Information theory
- Pattern recognition
- Percolation theory
- Probability theory
- Quantum mechanics
- Random walk
- Statistical mechanics
- Statistics

Part of a series on statistics

Probability theory



Probability (Axioms) · Determinism (System) · Indeterminism · **Randomness**



A pseudorandomly generated bitmap.

Randomness and pseudorandomness

Randomness and pseudo-randomness

What are the differences?

difference 1: (source ?)

difference 2: (generation time ?)

difference 3: (applicability ?)

difference 4: (possibility of bitstream prediction)

difference 5: (ability to influence the observation/recording/generation process ?)

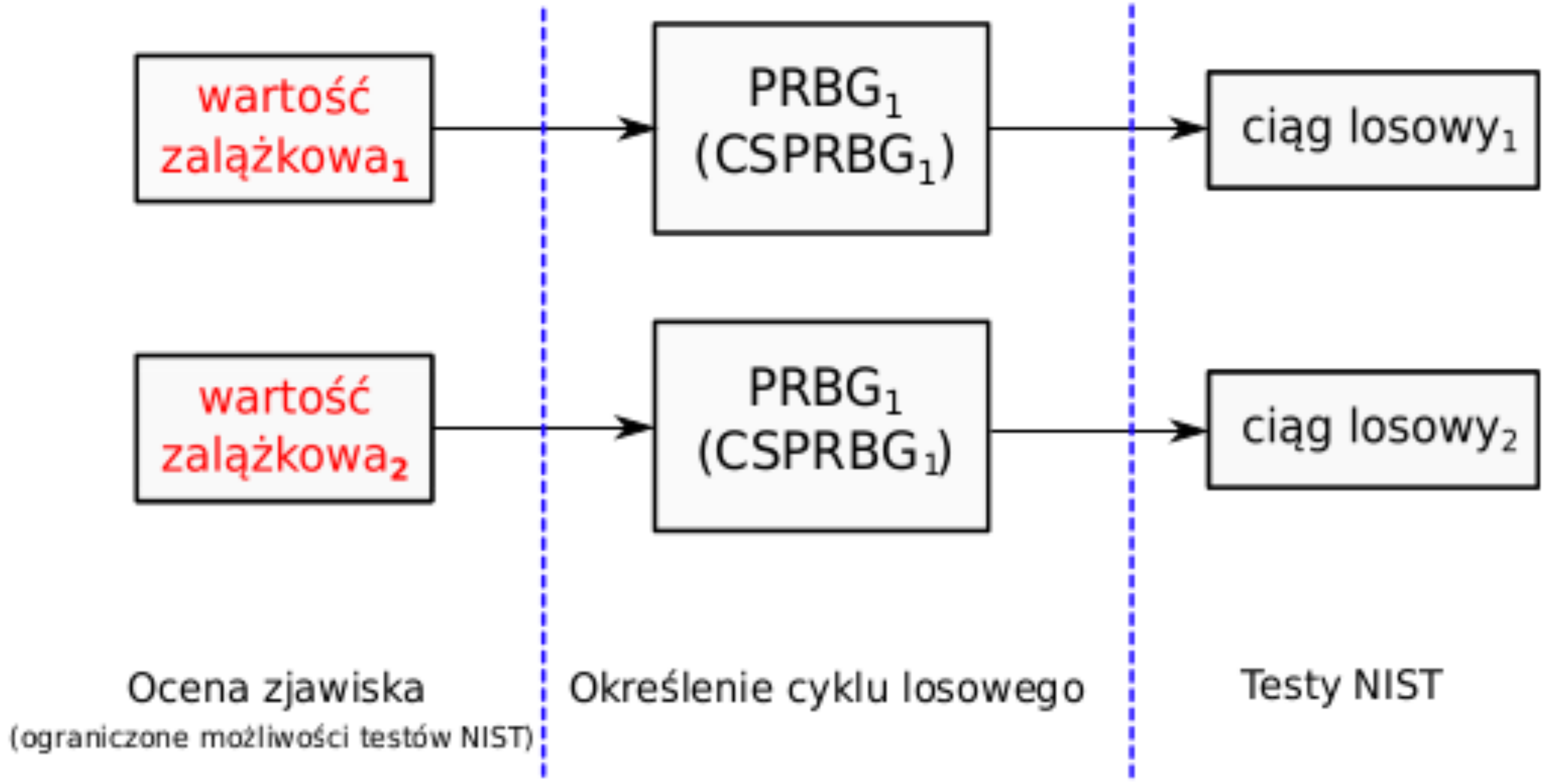
difference 6: (safety ?)

difference 7: (reproducibility / uniqueness ?)

PRBG

Pseudorandom bitstream generation

From seed to bitstream



PRBG and NIST Recommendations

NIST SP 800-90A January 2012

NIST Special Publication 800-90A

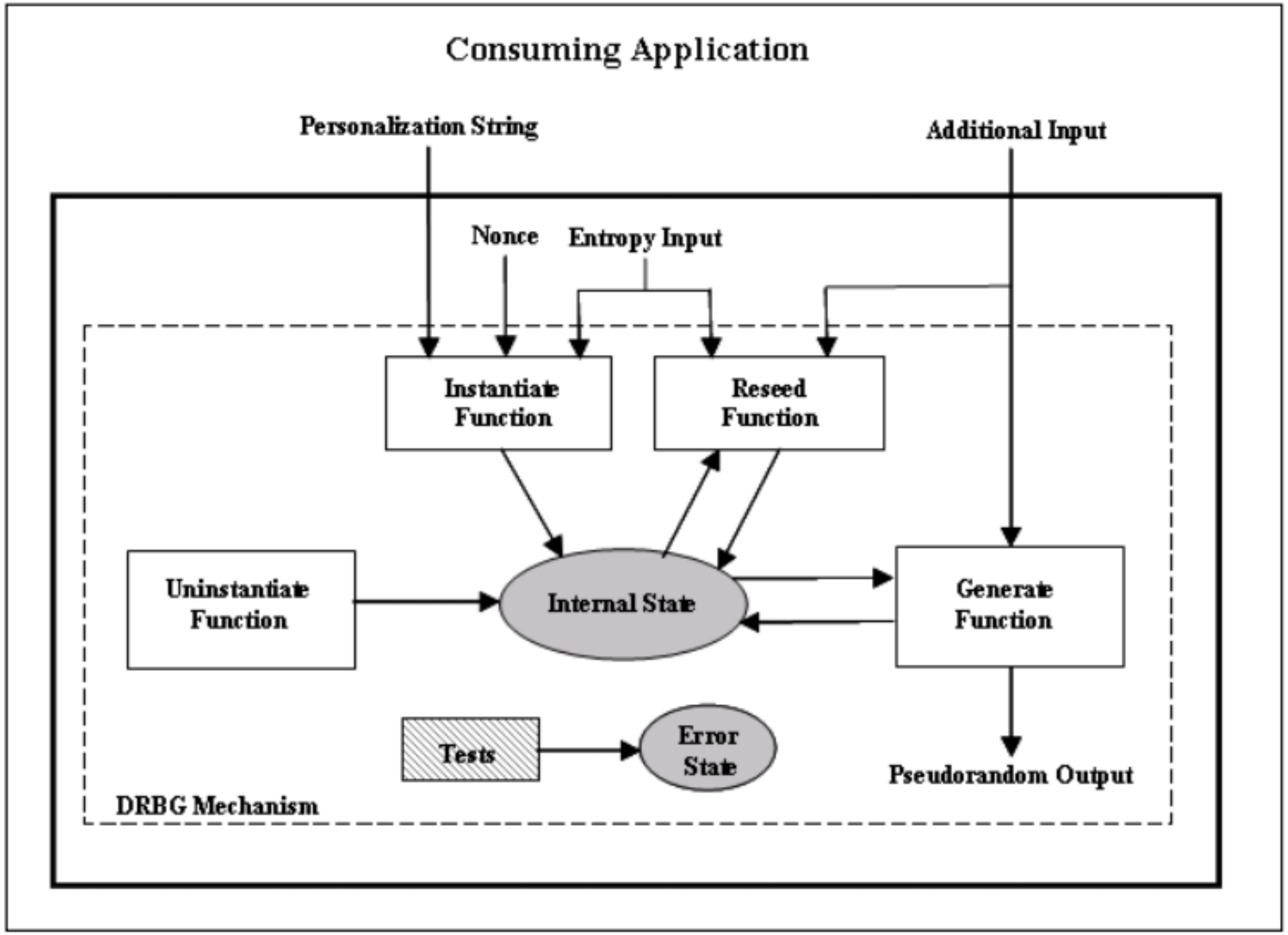
Recommendation for Random Number Generation Using Deterministic Random Bit Generators

Elaine Barker and John Kelsey

Computer Security Division
Information Technology Laboratory

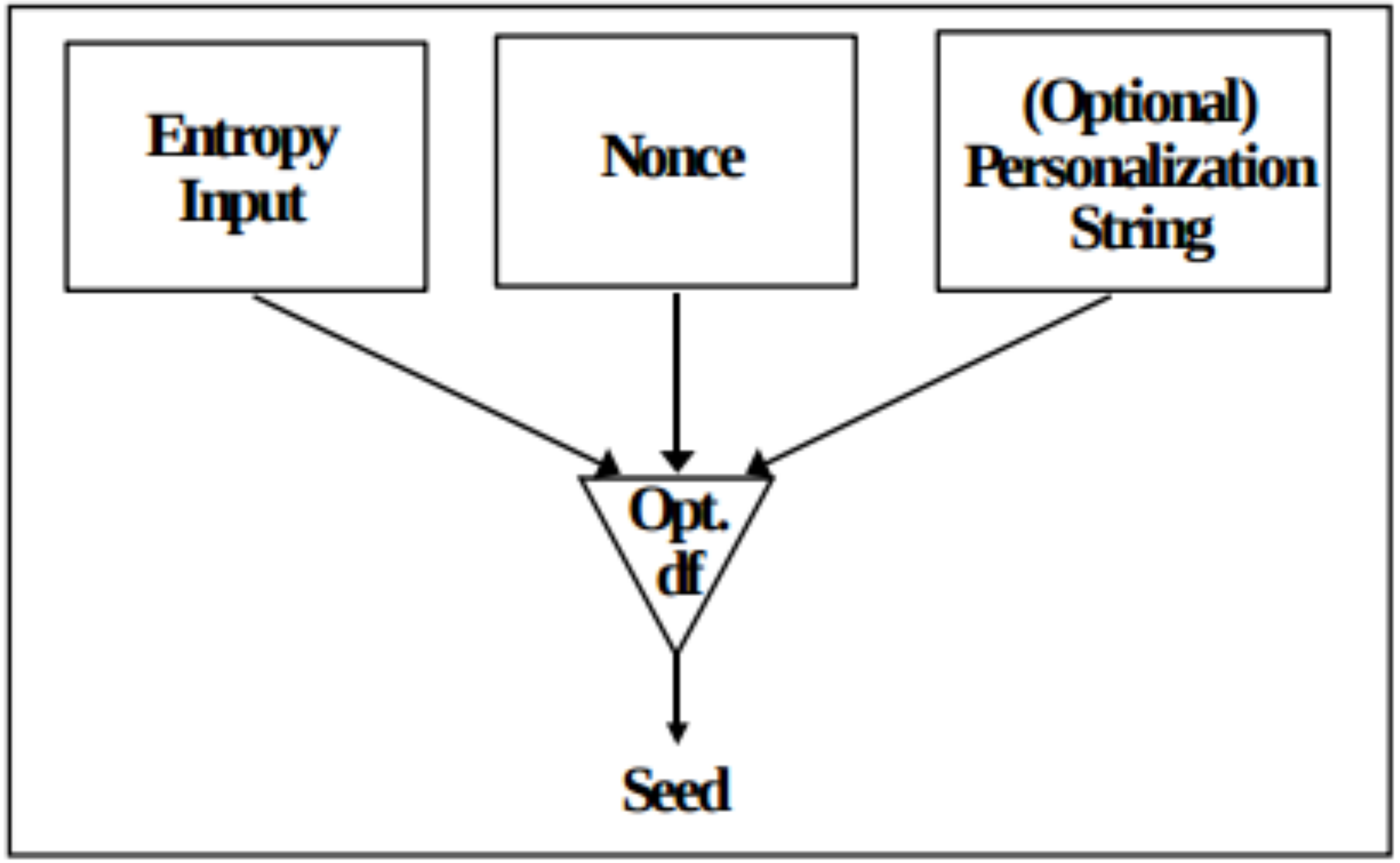
COMPUTER SECURITY

Deterministic random generator structure



Seed

How to create a seed?

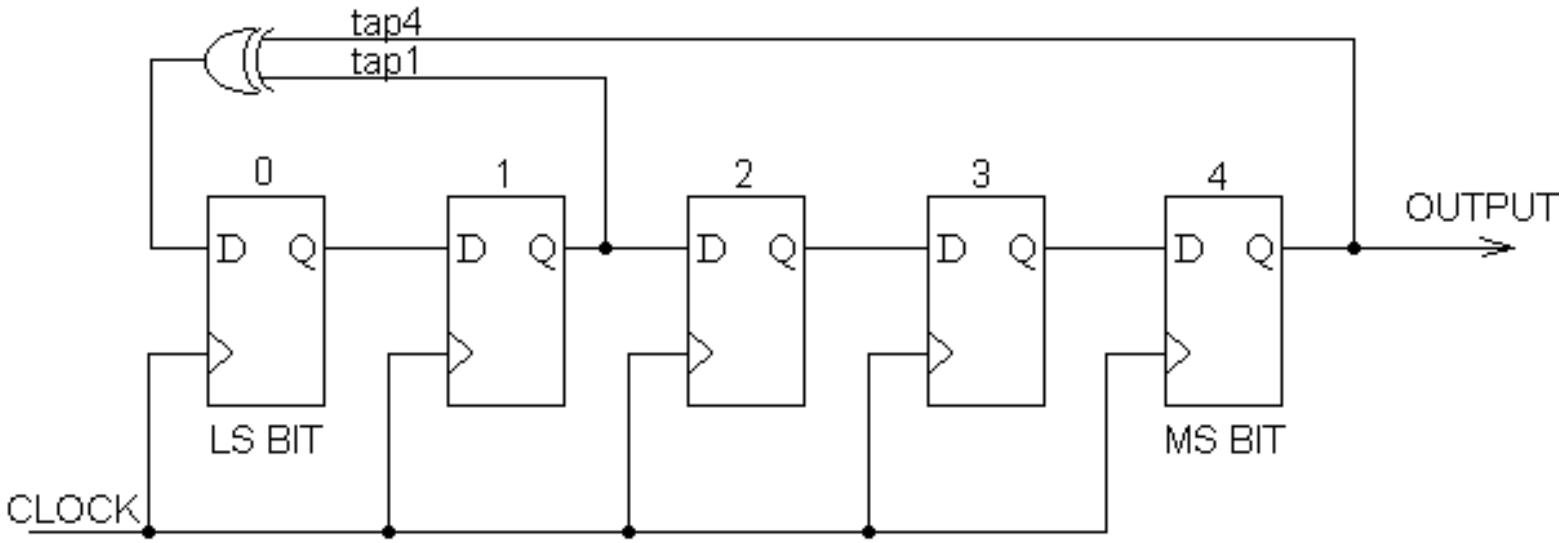


<u>Question to discuss with students at lecture:</u>

Where to get entropy from?

An example of a poor PRBG

PRBG based on LFSR



<u>Question to discuss with students at lecture:</u>

What are the disadvantages of the solution in this example?

Source(10.10.22): http://www.oocities.org/siliconvalley/screen/2257/vhdl/lfsr/lfsr.html

An example of a poor PRBG

PRBG based on LFSR

<u>Question to discuss with students at lecture:</u>

What are the disadvantages of the solution in this example?

It will produce a pseudorandom sequence of length 2n-1 states (where n is the number of stages) if the LFSR is of maximal length.

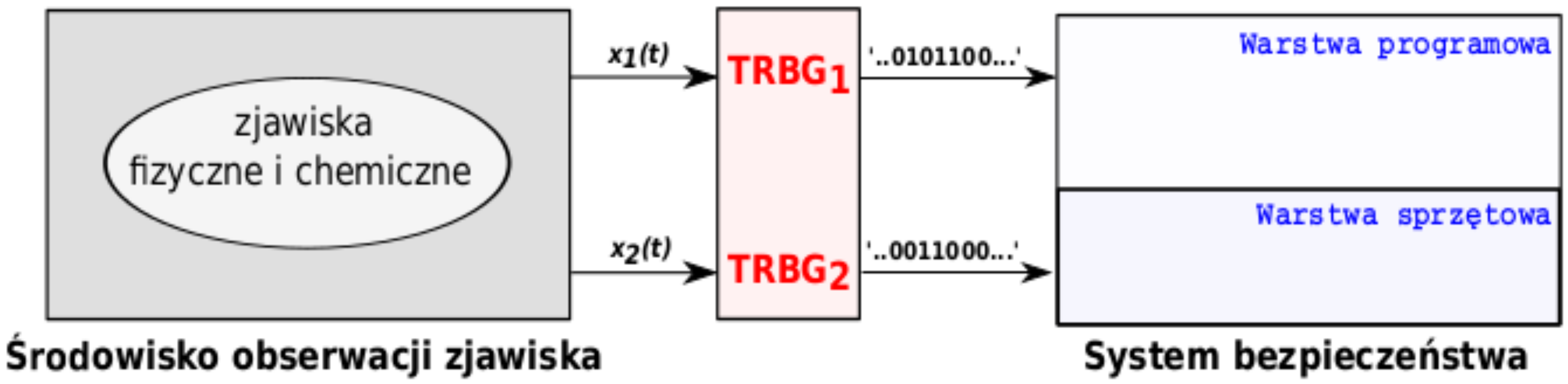
The sequence will then repeat from the initial state for as long as the LFSR is clocked.

Source: http://www.oocities.org/siliconvalley/screen/2257/vhdl/lfsr/lfsr.html

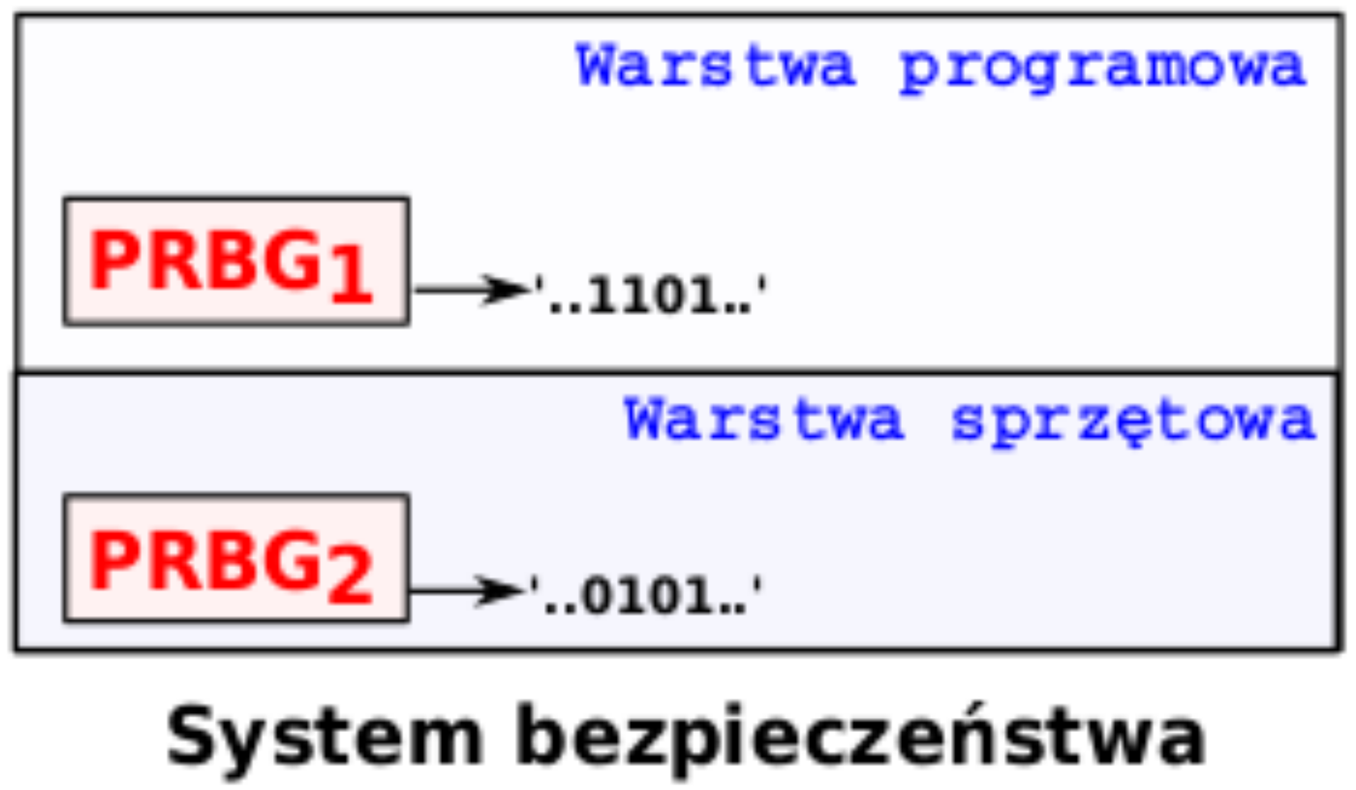
TRBG

True Random Bit Generator

Entropy from phenomena



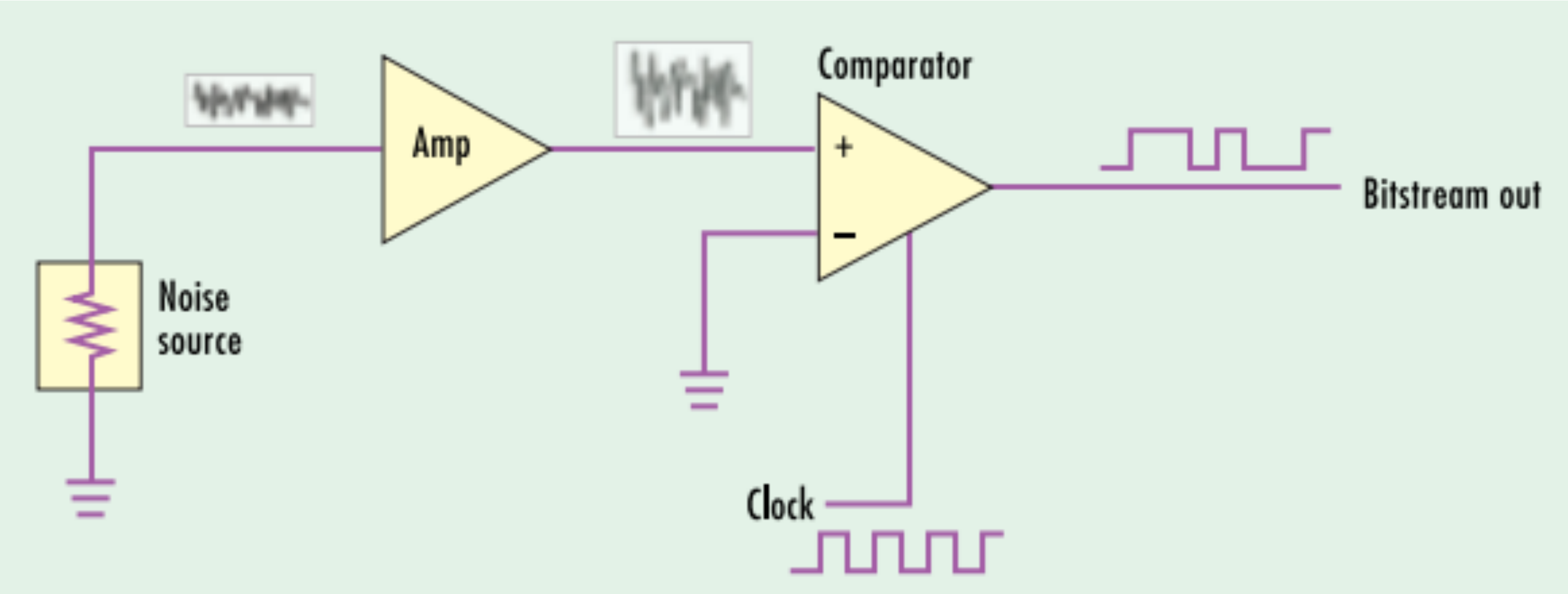
(a)



(b)

The simplest hardware TRBG

Entropy from phenomena



<u>Question to discuss with students at lecture:</u>

Why isn't this generator perfect?

Source (10.12.22): https://www.design-reuse.com/articles/5228/lockdown-random-numbers-secure-network-soc-designs.html

Bias in random bitstream

An example on the classroom whiteboard

0000000101010100010100111111111111111111111111110000000111111111111111111111110000000110101010010110010101011
1111010101001010100101001001010101010010101010101001010101010010100111111110101111000000111100111010010101

randomness extractor

Post-processing and von Neumann corrector

000000010101010001010011111111111111111111111110000000111111111111111111111000000011010101001011001010101
111101010100101010010100100101010101001010101010100101010100101001111111101011110000001111001110100101 0

VON NEUMANN CORRECTOR

Input Value	Output
00	Discard
01	0
10	1
11	Discard

Question to discuss with students at lecture:

What bitstream will be formed after such correction?

Quantum True Random Bit Generator

Solutions from IDQ

Quantis-USB-4M module

- 4Mbps of true quantum randomness
- Certified by Swiss National Laboratory
- USB 2.0 interface
- OS Support: Windows, Linux, Solaris, FreeBSD, MAC OS X
- Demo application

€ 990

Quantity : 0 (Promotional offer : free shipping for online purchases)

Quantum True Random Bit Generator

Solutions from IDQ

Quantis-PCIe-4M Card

- 4Mbps of true quantum randomness
- PCI Express interface
- Certified by Swiss National Laboratory
- OS Support: Windows, Linux, Solaris, FreeBSD
- Demo application

€ 1299

Quantity : 0 (Promotional offer : free shipping for online purchases)

Quantis-PCIe-16M Card

- 16Mbps of true quantum randomness
- PCI Express interface
- Certified by Swiss National Laboratory
- OS Support: Windows, Linux, Solaris, FreeBSD
- Demo application

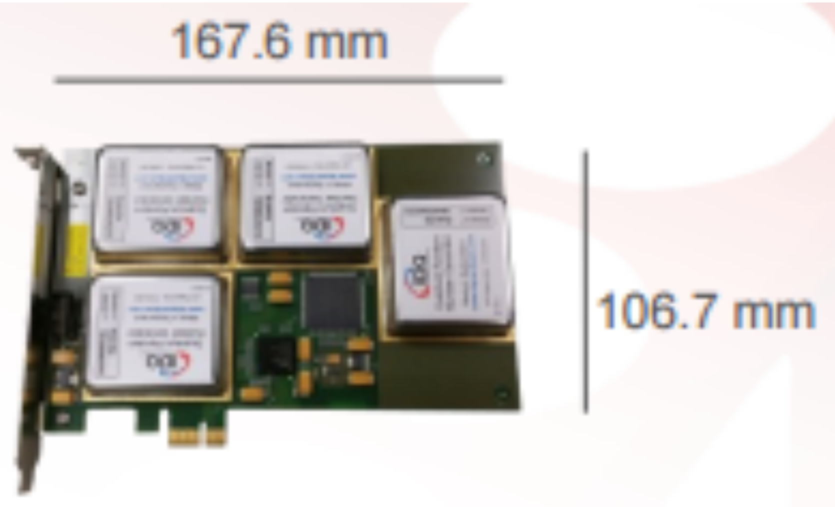
€ 2990

Quantity : 0 (Promotional offer : free shipping for online purchases)

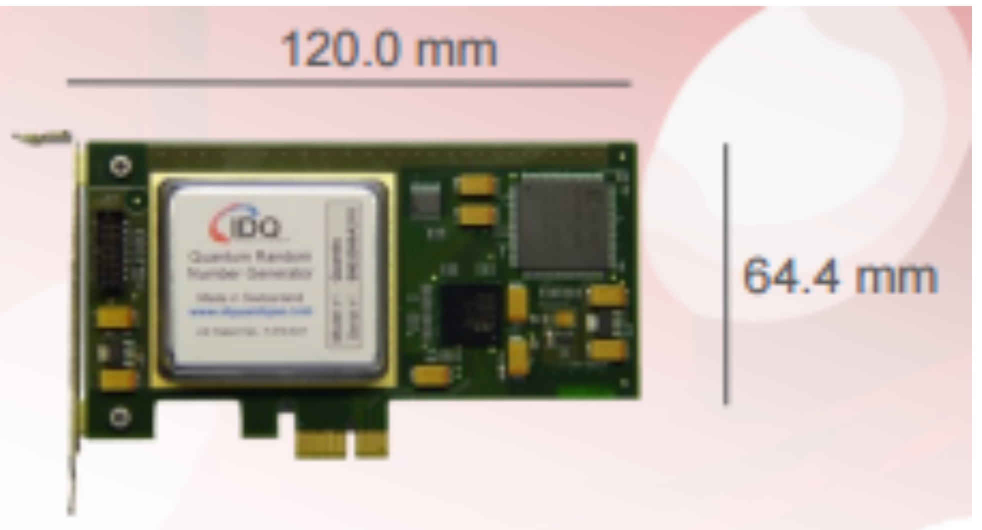
Quantum True Random Bit Generator

Solutions from IDQ

QUANTIS PCIe Card



Quantis-PCIe-16M



Quantis-PCIe-4M

GENERAL SPECIFICATIONS

1 : Hardware bit rate prior to randomness extraction

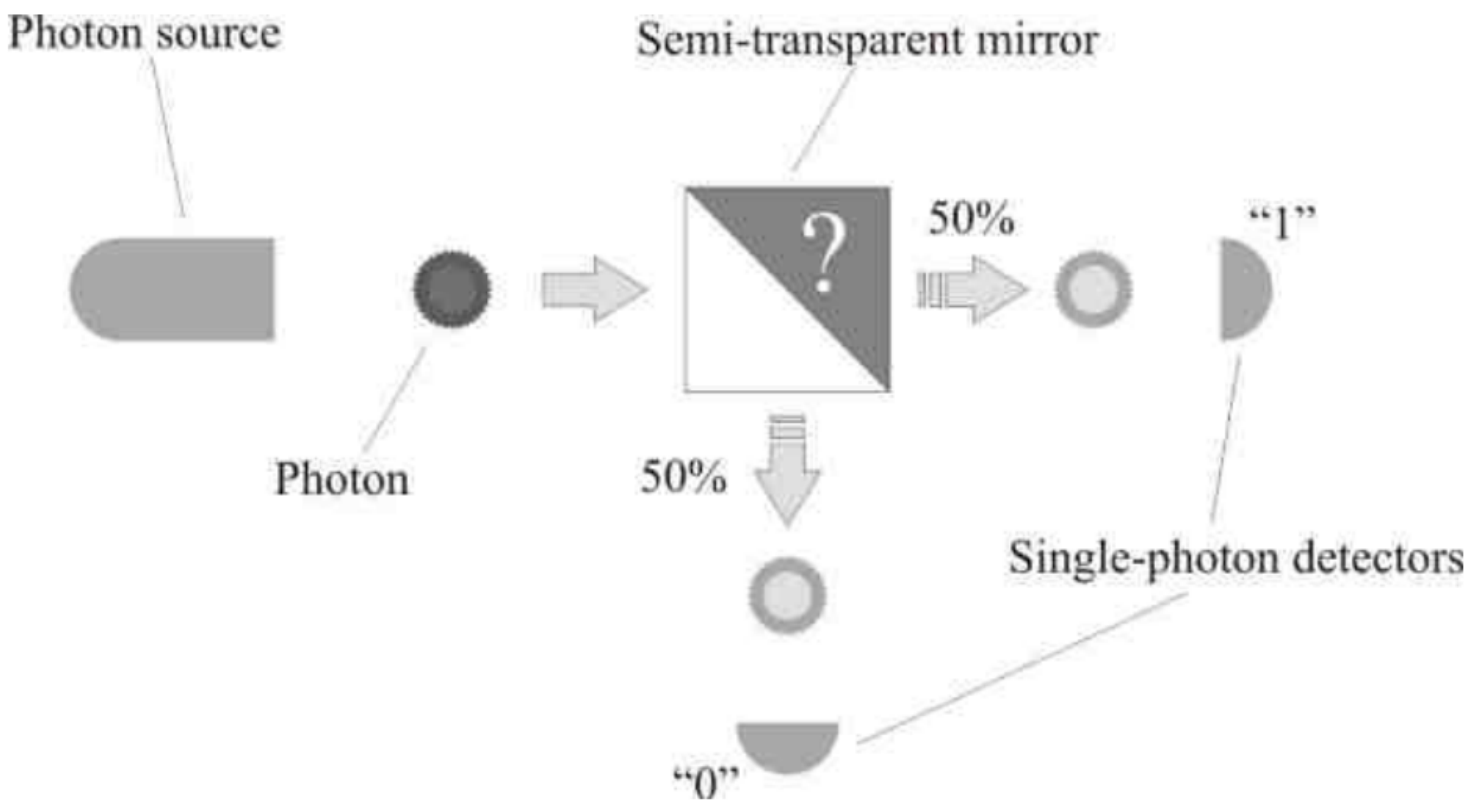
Random bit rate[1]	4 Mbit/s ± 10% (Quantis-PCIe-4M)
	16 Mbit/s ± 10% (Quantis-PCIe-16M)
Thermal noise contribution	< 1% (Fraction of random bits arising from thermal noise)
Storage temperature	- 25 to + 85°C
Dimensions	120 mm x 64.4 mm (Quantis-PCIe-4M)
	167.7 mm x 106.7 mm (Quantis-PCIe-16M)
PCI Express specification	PCI Express Base 1.0a compliant
Requirement	PCI with available PCIe slot

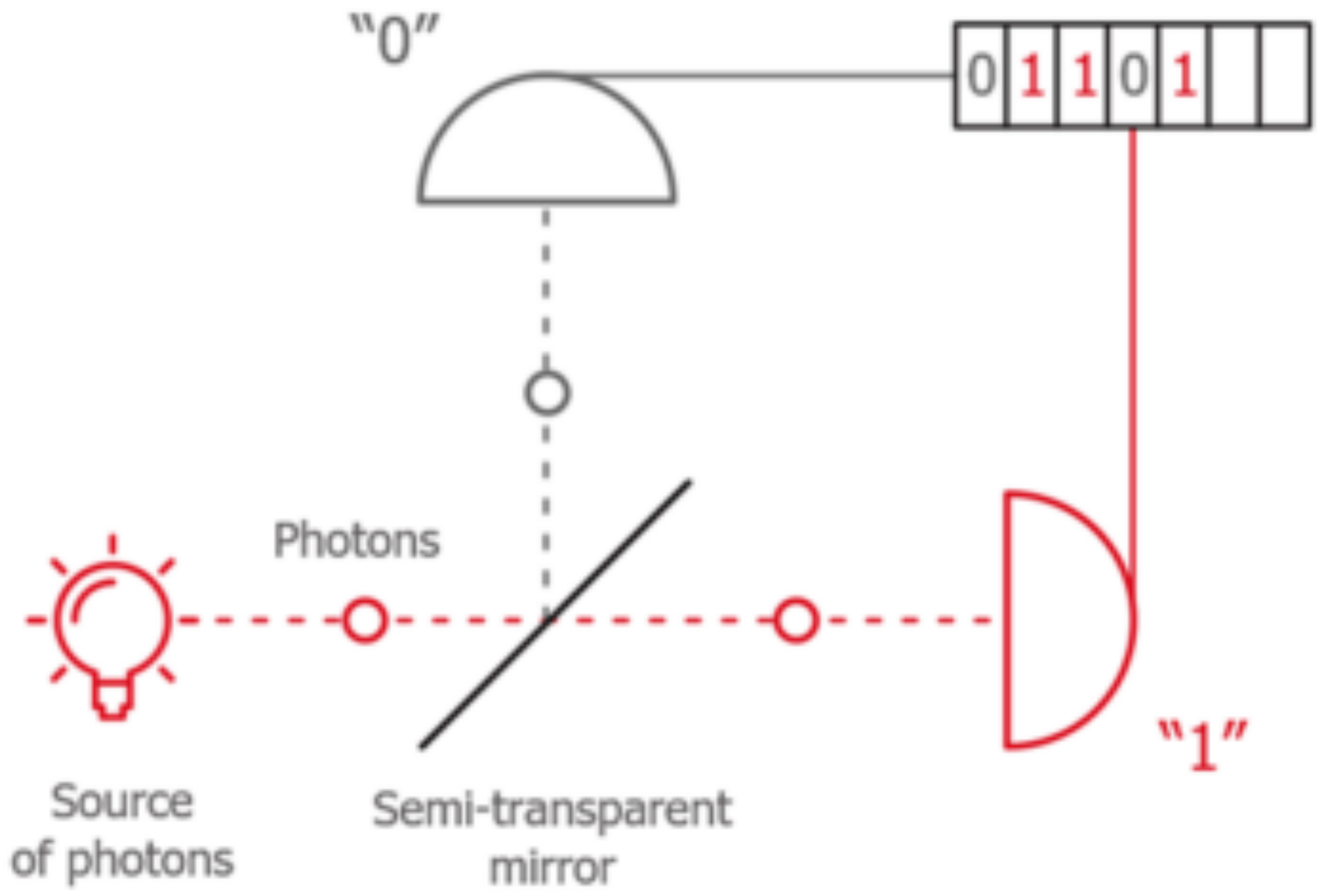
Quantum True Random Bit Generator

How it works?

Based on Quantum Physics :

Photons - light particles - are sent one by one onto a semi-transparent mirror and detected. The exclusive events (reflection - transmission) are associated to « 0 » - « 1 » bit values.





Quantum True Random Bit Generator

Are they really random?

Test name	Mean of p-value	Variance	Conclusion
Approximate Entropy Test	0.489	0.088	SUCCESS
Frequency Test within a Block	0.506	0.081	SUCCESS
Cumulative Sums Test	0.499	0.081	SUCCESS
Discrete Fourier Transform (Spectral) Test	0.493	0.079	SUCCESS
Binary Matrix Rank Test	0.498	0.084	SUCCESS
Run Test	0.497	0.081	SUCCESS
Serial Test	0.495	0.078	SUCCESS
Maurer's Universal Statistical Test	0.493	0.081	SUCCESS
Linear Complexity Test	0.499	0.083	SUCCESS
Test for the Longest Run of Ones in a Block	0.503	0.087	SUCCESS
Non-overlapping Template Matching Test	0.499	0.082	SUCCESS
Overlapping Template Matching Test	0.490	0.081	SUCCESS
Frequency (Monobit) Test	0.505	0.084	SUCCESS
Lempel-Ziv Compression Test	0.480	0.080	SUCCESS
Random Excursions Test	0.503	0.083	SUCCESS
Random Excursions Variant Test	0.502	0.082	SUCCESS

Quantum True Random Bit Generator

Are they really random?

A. NIST : SP800-22 Test Suite Compliance

The National Institute for Standards and Technology (NIST) is the US agency dedicated to setting new standards in every technological field: bioscience / energy / communication / etc. Regarding IT security, the NIST standards are designed for the American federal agency security level, which makes them highly trustworthy.

The Quantis has been submitted to the NIST Special Publication 800-22 named "**A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications**" and successfully passed all the test suites.

CSPRBG

Cryptographically secure PRBG

What does wikipedia have to say about it?

Cryptographically secure pseudorandom number generator

From Wikipedia, the free encyclopedia

A **cryptographically secure pseudorandom number generator** (**CSPRNG**) or **cryptographic pseudorandom number generator** (**CPRNG**)[1] is a pseudorandom number generator (PRNG) with properties that make it suitable for use in cryptography. It is also loosely known as a **cryptographic random number generator** (**CRNG**) (see Random number generation § "True" vs. pseudo-random numbers).[2][3]

Cryptographically secure PRBG

FORTUNA generator - HW

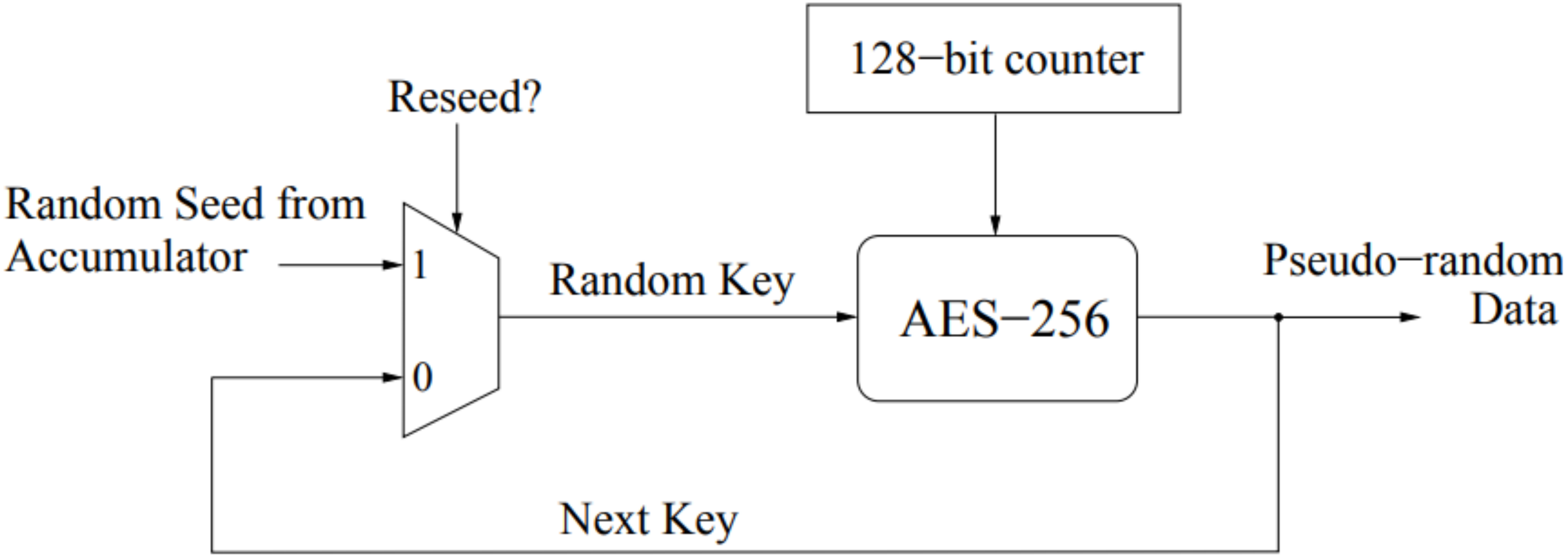


Fig. 2: Fortuna's Generator Core

Source: McEvoy, Robert, et al. "Fortuna: cryptographically secure pseudo-random number generation in software and hardware." *2006 IET Irish Signals and Systems Conference*. IET, 2006.

Cryptographically secure PRBG

FORTUNA generator - seed procedure

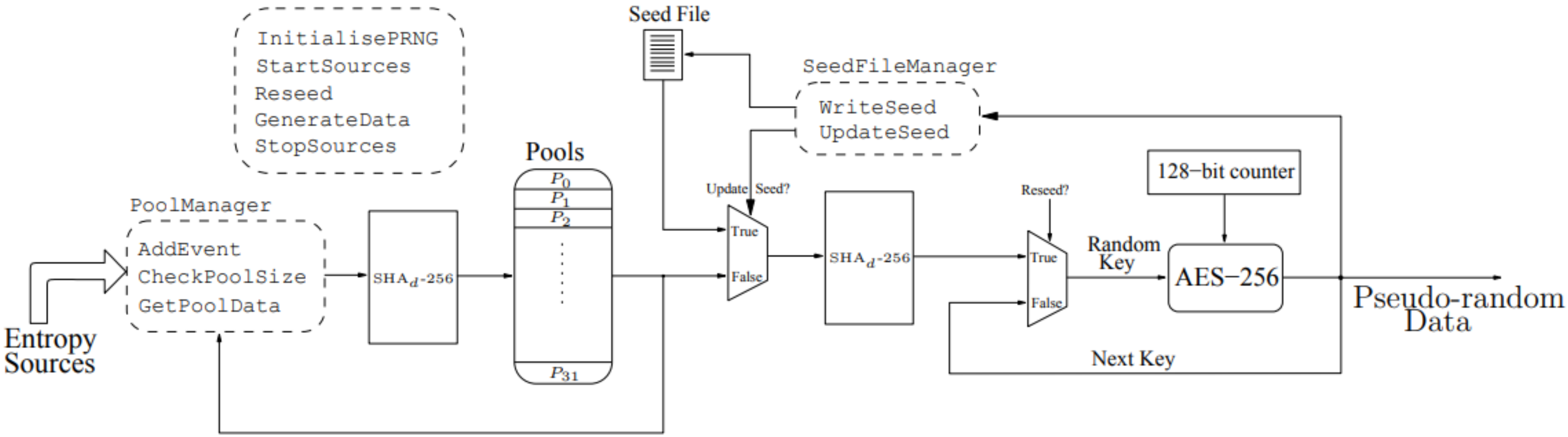


Fig. 4: Software Flow Diagram of Fortuna Implementation

Source: McEvoy, Robert, et al. "Fortuna: cryptographically secure pseudo-random number generation in software and hardware." *2006 IET Irish Signals and Systems Conference*. IET, 2006.

Tools for evaluating randomness

Ent

A Pseudorandom Number Sequence Test Program

This page describes a program, **ent**, which applies various tests to sequences of bytes stored in files and reports the results of those tests. The program is useful for evaluating pseudorandom number generators for encryption and statistical sampling applications, compression algorithms, and other applications where the information density of a file is of interest.

```
Entropy = 7.980627 bits per character.

Optimum compression would reduce the size
of this 51768 character file by 0 percent.

Chi square distribution for 51768 samples is 1542.26, and randomly
would exceed this value less than 0.01 percent of the times.

Arithmetic mean value of data bytes is 125.93 (127.5 = random).
Monte Carlo value for Pi is 3.169834647 (error 0.90 percent).
Serial correlation coefficient is 0.004249 (totally uncorrelated = 0.0).
```

NIST test for randomness

NIST
National Institute of
Standards and Technology
Technology Administration
U.S. Department of Commerce

Special Publication 800-22
Revision 1a

A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications

Andrew Rukhin, Juan Soto, James Nechvatal, Miles Smid, Elaine Barker, Stefan Leigh, Mark Levenson, Mark Vangel, David Banks, Alan Heckert, James Dray, San Vo

Revised: April 2010
Lawrence E Bassham III

NIST test for randomness

Statistical tests

Table 1. The recommended size n of the bitstream for each particular test (Some tests are parameterised by a second parameter m, M, respectively. The table shows meaningful settings for the second parameter and the number of sub-tests executed by each particular test.)

Test #	Test name	n	m or M	# sub-tests
1.	Frequency	$n \geq 100$	-	1
2.	Frequency within a Block	$n \geq 100$	$20 \leq M \leq n/100$	1
3.	Runs	$n \geq 100$	-	1
4.	Longest run of ones	$n \geq 128$		1
5.	Rank	$n > 38\,912$	-	1
6.	Spectral	$n \geq 1000$	-	1
7.	Non-overlapping T. M.	$n \geq 8m - 8$	$2 \leq m \leq 21$	148*
8.	Overlapping T.M.	$n \geq 10^6$		1
9.	Maurer's Universal	$n > 387\,840$		1
10.	Linear complexity	$n > 10^6$	$500 \leq M \leq 5000$	1
11.	Serial		$2 < m < \lfloor \log_2 n \rfloor - 2$	2
12.	Approximate Entropy		$m < \lfloor \log_2 n \rfloor - 5$	1
13.	Cumulative sums	$n \geq 100$		2
14.	Random Excursions	$n \geq 10^6$		8
15.	Random Excursions Variant	$n \geq 10^6$		18

NIST test for randomness

Statistical tests

	Test name	Short description
1	Frequency (Monobit) test	Tests proportion of zeros and ones
2	Frequency test within a block	Tests proportion of ones within M-bit blocks
3	Runs test	Tests total number of sequences of identical bits
4	Test for the longest run of ones in a block	Searches for the longest run on ones within M-bit blocks
5	Binary matrix rank test	Tests the rank of disjoint sub-matrices
6	Discrete Fourier Transform (spectral) test	Observes peak heights in the DFT
7	Non-overlapping template matching test	Counts number of occurrences of the pre-specified target strings
8	Overlapping template matching test	Test number of occurrences of the pre-specified target strings
9	Maurers 'Universal statistical' test	Detects whether the sequence can be significantly compressed
10	Linear complexity test	Determines if the sequence is complex enough or not
11	Serial Test	Tests the frequency of all possible overlapping m-bit patterns
12	Approximate entropy test	Compares the frequency of overlapping blocks
13	Cumulative sums (Cusum) test	Tests the cumulative sum of the partial sequences
14	Random excursions test	Tests the number of visits to a particular state within a cycle
15	Random excursions variant test	Counts total number of times that a particular state occurs

Dieharder: A Random Number Test Suite

dieharder

by
Robert G. Brown
Duke University Physics Department
Durham, NC 27708-0305

Abstract

Dieharder: A Random Number Test Suite

Version 3.31.1

Robert G. Brown (rgb)

Dirk Eddelbuettel

David Bauer

Contents

Document Type	Size (K)	Last Modified
HTML (page)	N/A	06/19/17
dieharder/dieharder-2.24.1-0.i386.rpm	94	06/19/17
dieharder/dieharder-2.24.1-0.src.rpm	539	06/19/17
dieharder/dieharder-2.24.1-1.i386.rpm	97	06/19/17
dieharder/dieharder-2.24.1-1.src.rpm	427	06/19/17
dieharder/dieharder-2.24.2-0.i386.rpm	96	06/19/17
dieharder/dieharder-2.24.2-0.src.rpm	427	06/19/17
dieharder/dieharder-2.24.3-0.i386.rpm	96	06/19/17
dieharder/dieharder-2.24.3-0.src.rpm	519	06/19/17
dieharder/dieharder-2.24.4-0.i386.rpm	97	06/19/17
dieharder/dieharder-2.24.4-0.src.rpm	778	06/19/17
dieharder/dieharder-2.24.7-0.i386.rpm	96	06/19/17
dieharder/dieharder-2.24.7-0.src.rpm	886	06/19/17
dieharder/dieharder-2.24.7-1.i386.rpm	96	06/19/17
dieharder/dieharder-2.24.7-1.src.rpm	897	06/19/17
dieharder/dieharder-2.27.10-1.src.rpm	1382	06/19/17
dieharder/dieharder-2.27.10-1.x86_64.rpm	102	06/19/17

EC and Random Bit Generators

Whiteboard exercise with students in the classroom

Imagine that we are designing an edge system, which should include a random number generator.

What absolutely should be paid attention to when selecting such a generator or designing it?