NATIVE TERNARY RANDOM NUMBERS GENERATION

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This disclosure is a continuation-in-part of the US patent application 2017/0046129.

TECHNICAL FIELD

[0002] The present disclosure relates to implementations of computing systems. Specifically, the disclosure describes implementations of physically unclonable functions (PUFs) and true random number generators.

BACKGROUND

[0003] Ternary computing uses trits, or a base three numeral system. For example (0, 1, 2) or balanced (-, 0, +), instead of the bits (0, 1) used in typical binary computing. Random numbers are needed for multiple security protocols, such as cryptography, computer security, or other applications where unpredictable results are important. One way to create ternary random numbers is to convert binary random numbers into decimal numbers, then to convert the decimal data stream back into ternary random numbers. Such a method adds complexity to the process, and can potential vulnerabilities that expose the random numbers to hackers. A direct generation of native ternary random numbers is therefore desirable as a more secure, simple solution.

BRIEF DESCRIPTION OF DRAWINGS

[0004] FIG. 1 depicts design of native ternary true random number generator.

[0005] FIG. 2 illustrates the marginal cells of a PUF.

[0006] FIG. 3 depicts three types of sorted cells.

[0007] FIG. 4 is a schematic of the modulo 3 adder.

[0008] FIG. 5 is a simplified model of the mod 3 adder with two cells.

[0009] FIG. 6 is a diagram showing impact of mod 3 adder with two cells on randomness.

[0010] FIG. 7 is a diagram showing 3 grouping of the mod 3 adder with 2 cells.

[0011] FIG. 8 is a diagram showing the mod 3 adder with four cells.

[0012] FIG. 9 is a table describing the level of randomness by experimental case.

[0013] FIG. 10 is a graph showing how the level of randomness changes after the mod 3.

DETAILED DESCRIPTION

[0014] This disclosure, its aspects and implementations, are not limited to the specific components, assembly procedures or method elements disclosed herein. Many additional components, assembly procedures and/or method elements known in the art consistent with the intended physically unclonable function (PUF) generating systems and related methods will become apparent for use with particular implementations from this disclosure. Accordingly, for example, although particular implementations are disclosed, such implementations and implementing components may comprise any shape, size, style, type, model, version, measurement, concentration, material, quantity, method element, step, and/or the like as is known in the art for such physically unclonable function generating systems and method implementations, and implementing components and methods, consistent with the intended operation and methods.

[0015] PUFs form a set of cryptographic primitives used in authentication/cryptographic methods. The underlying authentication mechanism in which PUFs are used is in generating Challenge-Response-Pairs/Patterns (CRP). PUFs that are unique to each device allow an authentication system to challenge a device seeking authentication, receive half of a CRP pair from the device, and then compare the received half with a stored CRP pair to verify whether a match takes place. Where the PUFs take advantage of the natural manufacturing variations unique to the device seeking authentication, a large number of Challenge (i.e. input) Response (i.e. output) pairs can be generated.

[0016] In this document, PUFs are derived from analyzing intrinsic manufacturing variations of semiconductor devices that are created during the fabrication of the devices. Non-limiting examples of such variations that alter device characteristics include variations in critical dimensions, doping level(s) of semiconducting materials, threshold voltages, variations in lot-to-lot processing, and variations in wafer-to-wafer processing. These variations make each device unique, and identifiable from every other device, like a device fingerprint or biological DNA.

[0017] Methods to create binary random number generators (RNGs) from PUFs have been described previously, and could utilize static random access memory (SRAM), dynamic random

access memory (DRAM), Flash RAMs, Resistive RAM (ReRAM), or spin transfer torque magnetic random access memory (STT MRAM). Intrinsic variations due to manufacturing variations in the memory arrays are exploited for binary random number generators. One of the generic methods to generate random numbers is to characterize a particular parameter $\mathcal{P}$ of the cells of the memory array with a "built-in-self-test" (BIST) module. Each cell being different, the value of parameter $\mathcal{P}$ varies cell to cell, and follows a distribution with a median value T. For challenge and response generations, all cells with $\mathcal{P}$ below T can be then considered as "0", all others as "1". The cells of memory arrays are segmented between the predictable cells which reliably produce a 1 or 0, and the X-cells of the array that are unstable and produce a value close to the threshold T. These characteristics enable the design of both solid PUFs, and ternary random number generators (TRNGs).

[0018] Each bit of a binary data stream of N bits of a perfect RNG should have precisely the same probability to be either a "1" or a "0", $P_{(X=1)} = P_{(X=0)} = 0.5$. The average deviation from perfection, denoted by λ, is given by: $\lambda = \frac{1}{2}(|P_{(X=1)} - 0.5| + |P_{(X=0)} - 0.5|)$.

[0019] In the case of ternary data streams with three possible data states consisting of "-", "0", or "+", λ is given by: $\lambda = 1/3\,(|P_{X=-} - 1/3| + |P_{X=0} - 1/3| + |P_{X=+} - 1/3|)$

[0020] In one embodiment, the length of a RNG data stream may be N=128, making the number of possible combinations $2^{128} = 3.4x10^{38}$ for binary streams, and $3^{128} = 1.2x10^{61}$ for ternary streams, which is considerably larger. A paper from the National Institute of Standards and Technology suggested in 1999 that a value for λ greater than 10^{-3} would not be acceptable for binary RNG, $\lambda < 10^{-5}$ is currently considered as an excellent target.

[0021] Some embodiments may use $\oplus$ Modulo 3 sum adders for additional randomization. With Modulo 3 sum adders, two input trits X, and Y are transformed into the trit $Z = X \oplus Y$, with the following truth table: ($Z=-$) if ($X=0$) and ($Y=-$), or ($X=-$) and ($Y=0$), or ($X=Y=+$); ($Z=0$) if ($X=+$) and ($Y=-$), or ($X=-$) and ($Y=+$), or ($X=Y=0$); ($Z=+$) if ($X=0$) and ($Y=+$), or ($X=+$) and ($Y=0$), or ($X=Y=-$).

[0022] The reason Modulo 3 sum adders increase randomness is that the knowledge of Z does not disclose the value of X and Y. For example, Z = -, Z = 0, or Z = + can all be the result of three different XY pairs with equal probability. If two trits X and Y are somewhat random, the third trit Z is even more random than either X or Y. A quantification of the effect of Modulo 3 sum adders

in enhancing the randomization of native ternary RNG is presented below, and a method for producing a ternary true random number generator.

[0023] In one embodiment, the ternary X-cells from a PUF may be sorted. The first step of the method may be the generation of a native ternary random number from a PUF as depicted in FIG. 1. Within the individual cells of a PUF memory array, it can be assumed that the distribution of a physical parameter $\mathcal{P}$ used to determine if a cell is a "-" or "+" follows a normal distribution, with a standard variation σ due to inherent cell-to-cell variations created during manufacturing or by other instabilities. The median of the distribution represents a threshold T_1 that creates a binary system: if $\mathcal{P}$ falls below the threshold of the distribution it will be assigned a "-", while values of $\mathcal{P}$ above the threshold may be assigned a "+".

[0024] It can also be assumed that every time a new measurement of $\mathcal{P}$ is taken on a cell, this measurement will also follow a normal distribution with a standard variation σ_{intr} responding to various measurement instabilities, noise, and environmental variations. The standard deviation of the measurement σ_{intr} should be much lower than the standard deviation σ of the population for high-quality memory arrays, such that "-", and "+" are clearly and consistently defined by memory cells. The cells close to the threshold, X-cells, are not stable. FIG. 2 illustrates a histogram distribution of the memory cells. Parameter $\mathcal{P}$ X-cell 203 is near the threshold 205 of the distrbution, making it unstable. To maximize the randomness of an X-cell, the distance between the measurement of parameter $\mathcal{P}$, and the median should then be lower than σ_{intr}, indicating the variations in $\mathcal{P}$ are due to measurement instabilities, maximizing the chance of a random flip between "-" and "+". Unstable X-cells, for which the parameter $\mathcal{P}$ does not consistently classify them as "-" or "+", may then be one of three values, "-", "+", or "0." In such a case the level of randomness can be high; however, only a small percentage of the cells are categorized as X-cells. In an example embodiment for a 256 trit RNG, as much as 10,000 cells of the array may be required for high randomness.

[0025] FIG. 3 depicts a normal distribution of the measurement parameter $\mathcal{P}$ for an example memory array. Cells that are "-" would fall under the distribution curve in area 310, whereas cells that are "+" would fall under the distribution curve in area 311. X-cells, with measurement parameter $\mathcal{P}$ near threshold T_1 307 would fall within the shaded region 308. The X-cells can be further segmented into three subgroups based on parameter $\mathcal{P}$, A-cells 302, B-cells 304, and C-cells 306. The X-cells that have a higher probability to be tested as "-" are called A-cells. They

have an average probability $P_{A=-}$ to be tested as "-" , $P_{A=0}$ to be tested as "0", and $P_{A=+}$ to be tested as "+".

[0026] The X-cells that have a higher probability to be tested as "0" are called B-cells. They have an average probability $P_{B=-}$ to be tested as "-" , $P_{B=0}$ to be tested as "0", and $P_{B=+}$ to be tested as "+".

[0027] The X-cells that have a higher probability to be tested as "+" are called C-cells. They have an average probability $P_{C=-}$ to be tested as "-" , $P_{C=0}$ to be tested as "0", and $P_{C=+}$ to be tested as "+".

[0028] Two additional transition points, T_2 303 and T_3 305 can be established in addition to the threshold T_1 307. The selection of the transition point between A-cells and B-cells, T_2, and the transition of parameter $\mathcal{P}$ between B-cells and C-cells,T_3, can be such that each group can have an equivalent number of cells. In this method, the initial level of randomness λ_i of the native ternary random number is given by:

[0029] $$\lambda_i = \frac{|1/9-P_{A=-}|+|1/9-P_{A=0}|+|+|1/9-P_{A=+}|}{9} + \frac{|1/9-P_{B=-}|+|1/9-P_{B=0}|+|+|1/9-P_{B=+}|}{9} + \frac{|1/9-P_{C=-}|+|1/9-P_{C=0}|+|+|1/9-P_{C=+}|}{9}$$

[0030] Some embodiments may use Modulo 3 sum adders to create a truly random number generator (TRNG). In one embodiment, let us assume that the stream of random trits generated by the X-cells of a memory PUF array is $\{a_1, a_2, \ldots, a_i, \ldots a_n\}$, as depicted in FIG. 4. This stream is grouped in chunks of f trits $\{a_{1j}, a_{2j}, \ldots, a_{ij}, \ldots, a_{jf}\}$ with $f<n$. For example, 1,280 random trits may be grouped in 128 chunks of 10 trits. The resulting stream of random trits obtained with Modulo 3 sum adders data $\{c_1, c_2, \ldots, c_j, \ldots, c_m\}$ may be defined as follows:

[0031] $c_i = a_{1j} \oplus a_{2j} \oplus \ldots \oplus a_{ij} \oplus \ldots \oplus a_{jf} \bmod 3$

[0032] The truth table of $\oplus$ is also shown in FIG. 4.

[0033] As shown below such a randomization when applied to a stream of random trits, generated by a PUF memory array is extremely effective to generate truly random trits and be a TRNG. Modulo 3 sum adders can be implemented at the software level, or in hardware with only a few logic gates. These gates can be inserted in the state machine of the PUF memory to directly feed secure processors, and crypto-processors with quality streams of random trits.

[0034] Some embodiments may use Modulo 3 sum adders with 2-cells. In order to develop a model that quantifies the level of randomness of the method, the impact on two cells is considered,

when two adjacent trits are added to modulo 3. FIG. 5 shows an illustrated depiction of such a scheme.

[0035] The X-cells are equally distributed into three type of cells (A, B, & C), each of them with a probability of occurrence of 1/3. Statistically the stream of trits $\{a_1, a_2, \ldots, a_i, \ldots a_n\}$ contain trits with equal probability to be "-", "0", or "+", also with a value of 1/3. The top table shown in FIG. 6 is an example of distribution of probability verifying such a distribution. For example, $P_{A=-}$ =1/9 + Δ.λ with Δ=1.8; $P_{B=0}$ =1/9 + Δ.λ with Δ=+0.9; $P_{C=+}$ =1/9 + Δ.λ with Δ=1.8. The initial randomness is $\lambda i = (1/9) \sum |\Delta \,.\, \lambda| = \lambda$.

[0036] In some embodiments, the individual cells of a memory array may be combined by pairs to be added mod 3, creating nine possible combinations with an equal probability of 1/9. These combinations of A-cells, B-cells, and C-cells includes AA, AB, AC, BA, BB, BC, CA, CB, CD. The overall probability to have trits with "-", "0", and "+" remain stable at 1/3 but increased number of possible combinations creates more randomness in the system.

[0037] For example, one pair of cells may comprise two A-cells. The resulting trit may be "-" when both A-cells are "+" (the probability is: $P_{A=+}\ P_{A=+}$), or the first A is "0" with the second A is "-" (the probability is: $P_{A=0}\ P_{A=-}$), or the first A is "-" with the second A is "0" (the probability is: $P_{A=-}\ P_{A=0}$).

[0038] All together the probability $P_{AA=-}$ that two cells AA can result in a trit at "-" is given by:

[0039] $P_{AA=-} = (P_{A=+} \times P_{A=+}) + (P_{A=0} \times P_{A=-}) + (P_{A=-} \times P_{A=0}) =$

[0040] $(1/9 - 1.35x\lambda)^2 + 2\,(1/9 + 1.8x\lambda)(1/9 - 0.45x\lambda) = 1/27 + 0.2x\lambda^2$

[0041] The bottom table in FIG. 6 is showing the probability to get "-", "0", or "1" after addition mod 3 for each pair ($P_{AA=-}$, $P_{AA=0}$, $P_{AA=+}$, $P_{AB=-}$, $P_{AB=0}$,…). The resulting deviation from randomness after addition is:

[0042] $\lambda' f = (1/27) \sum |\Delta \,.\, \lambda^2| \approx 2.27\, \lambda^2$.

[0043] For example, if the initial deviation from randomness is $\lambda = 2\ 10^{-2}$; the resulting deviation is:

[0044] $\lambda' f = 2.27 \times 2^2 \times 10^{-4} = 8.7\ 10^{-4}$

[0045] After addition, the nine possible configurations can be again combined into 3 type of cells A', B', and C' as shown in FIG.7. For example, the one that are mainly "-", the A'-cells consist of CC, AB, and BA pairs. The probability that A'-cell is "-" is: $P_{A'=-} = P_{CC=+} + P_{AB=-} + P_{AB=0} = 1/9 + 3.1\lambda^2$.

[0046] The method presented above can be extended to the addition mod 3 of four sequential trits to generate trits of higher randomness in other embodiments. This process is depicted in FIG. 8. The resulting deviation from randomness of the new stream of trits is given by:

[0047] $\lambda''f = (1/27) \sum |\Delta \, . \, \boldsymbol{\lambda^4}| \approx 11.3 \, \boldsymbol{\lambda^4}$; also: $\lambda''f = 2.27 \, (\lambda' f)^2 = 2.27 \, (2.27)^2 \, \boldsymbol{\lambda^4} \approx 11.3 \, \boldsymbol{\lambda^4}$

[0048] For example, if the initial deviation is $\lambda = 2x10^{-2}$; the resulting deviation is:

[0049] $\lambda''f = 11.3 \times 2^4 \times 10^{-8} = 1.81x10^{-6}$

[0050] By extension, after addition of 8 sequential trits, the deviation from randomness λ'''' will be: $\lambda''''f = 2.27 \, (\lambda''f)^2 \approx 290 \, \boldsymbol{\lambda^8}$

[0051] For example, if the initial deviation is $\lambda = 2 \, 10^{-2}$; the resulting deviation is: $\lambda''''f = 290 \times 2^8 x10^{-16} = 7.4x10^{-12}$, which is more than 10^{-9} times more randomness.

[0052] The parameter $\mathcal{P}$ that is analyzed is the V_{set} of the array of metal-oxide resistive RAM. The transition between the "-" state and the "+" state of the distribution is 2.1 Volts, and this distribution is considered a normal distribution.

[0053] In one embodiment, only *2%* of the cells are X-cells. For these cells, parameter $\mathcal{P}$ is very close to the transition of 2.1 Volt. $\lambda_i = 2x10^{-2}$.

[0054] In another embodiment, 4% of the cells are X-cells. $\lambda_i = 4x10^{-2}$.

[0055] In another embodiment, 7% of the cells are X-cells. $\lambda_i = 6x10^{-2}$.

[0056] In another embodiment, 11% of the cells are X-cells. $\lambda_i = 1x10^{-1}$.

[0057] The reason we are considering this range of options is to quantify the effectiveness of the addition mod 3 to scramble the trits, as a function of the initial randomness.

[0058] A probabilistic model is used to analyze the four experimental cases. FIG. 9 and FIG. 10 illustrate the impact of the addition mod 3 on the randomness of the resulting data streams of trits when the number of cells involved vary from 2 to 8.

[0059] In all cases the addition mod 3 when applied to a data stream of trits generated by a PUF is very efficient to enhance randomness. In one embodiment, the one with the highest level of initial randomness benefits the most from Modulo 3 sum adders. It is straight forward to use the model as a predictive tool. For example, as shown in FIG.10, the number cells necessary to get $\lambda_f < 5 \, 10^{-6}$ for case 1 is 4, it is 5 for case 2, it is 6 for case 3, and it is 8 for case 4.

[0060] The TRNG design can be generalized to other embodiments. As described previously herein, the random trits are generated from memory based physically unclonable functions. The

method can be extended to any PUF. It could be conceived that Flash NAND, NOR, or EEPROM memories may be used. In other embodiments it is conceived that DRAM, SDRAM, DDR-RAM memories may be used. Additionally, other PUF hardware devices that have been contemplated are ReRAM, CBRAM, Memristor, SRAM, and buffer memories. The TRNG system may be generalized to ring oscillators, gate-delay based PUFs, and optical or sensor PUFs.

[0061] In all cases the physical component generates a data stream of trits with a deviation from absolute randomness λ_{in}, the addition mod 3 can enhance the stream to meet a particular λ_{out} objective.

[0062] The method can be extended to n-value logic, for example quaternary logic (4 bits), pentagonal logic (5 bits), or hexagonal logic (6 bits). In such cases the X-cells are divided in n-different type of cells, and the addition of chunk of n-bits is done modulo "n".

[0063] The system and method presented here presents some distinct advantages over the prior art. The ability to directly generate a stream of random trits without having to convert binary data streams into ternary data streams. The combination of the use X-cells from a memory array, and the Modulo 3 sum adders enhance the randomness of the ternary TRNG. It is possible to model with the probabilistic model the minimum size of chunk of trits f that need to be added to reach a minimum λ_f.

[0064] The new method minimizes sensitivity to parameter drifts such as temperature or biasing conditions. The hardware implementation can use known circuitry in commercial CMOS for the modular 3 adders. The method can reach NIST expectations in term of deviation from pure randomness of the TRNG, even if the randomness created by the PUF is weak.

CLAIMS

We claim:

1. A method for generating random numbers, comprising:

generating a parameter value for each cell of a memory array, the parameter value following a normal distribution,

computing a threshold, the threshold corresponding to the mean of the normal distribution of parameter values,

computing an instability region within a certain standard deviation of the threshold,

assigning cells with parameter values above the instability region a value of one,

assigning cells with parameter values below instability region a value of zero,

assigning cells with parameter values in the threshold region as instable;

generating a data stream of parameter values for cells assigned as instable;

segmenting the data stream into specific groups of trits;

submitting all trits contained in each specific group to a mod 3 addition operation; and

generating a new stream of trits with a higher level of randomness than the initial data stream.

2. The method of claim 1, wherein measuring a parameter value for each cell is repeated every time a new random number is requested.

3. The system of claim 1, wherein the length of the chunk of trits are as low as two bits, or very long.

4. The method of claim 1, wherein trits may take a balanced subset of values including (-, 0, +), or (-1, 0, +1), or un-balanced values such as (0, 1, 2), or (0, 1, X).

5. The method of claim 1, wherein the memory array is a resistive random access memory, conductive bridge random access memory, Memristor, flash memory, electrically erasable programmable read-only memory, Dynamic random-access memory, synchronous dynamic random-access memory, double data rate synchronous dynamic random-access memory, Static random-access memory, a phase change memory, magnetoresistive random-access memory, a magnetic memory, or a hard disk of any type.

6. The method of claim 1, wherein generating the parameter value may include measuring a physical element, a physical unclonable function, or a nanocomponent of any type such as gate delay based PUF, ring oscillator based PUF, optical PUF, or sensor PUF.

7. The method of claim 1, wherein the length of specific groups of trits is increased when the level of randomness of the initial data stream is low, or reduced when the level of randomness of the initial data stream high.

8. The method of claim 1, wherein the length of specific groups of trits is increased when the expectation in the level of randomness of the final data stream is higher, or reduced when the level of randomness of the final data stream lower.

9. The method of claim 1, is expanded to create random numbers of native n-value logic such as quaternary (4 bits), pentagonal logic (5 bits), hexagonal logic (6 bits) by submitting all specific groups of trits to a mod-n addition function.

ABSTRACT:

[0066] This disclosure describes ways to design true random number generators (TRNG) from memory products with ternary states, and enhance the level of randomness with a XOR data compiler. In this disclosure, a method to directly generate random trits from PUF memory arrays is described, as well as ways to enhance randomness with modulo 3 sum adders. The deviation from absolute randomness of the resulting ternary random numbers is lowered by an approximate factor of 10^{-4} to 10^{-12} by grouping and adding the trits (modulo 3) located every chunk of 4 cells to 8 cells.

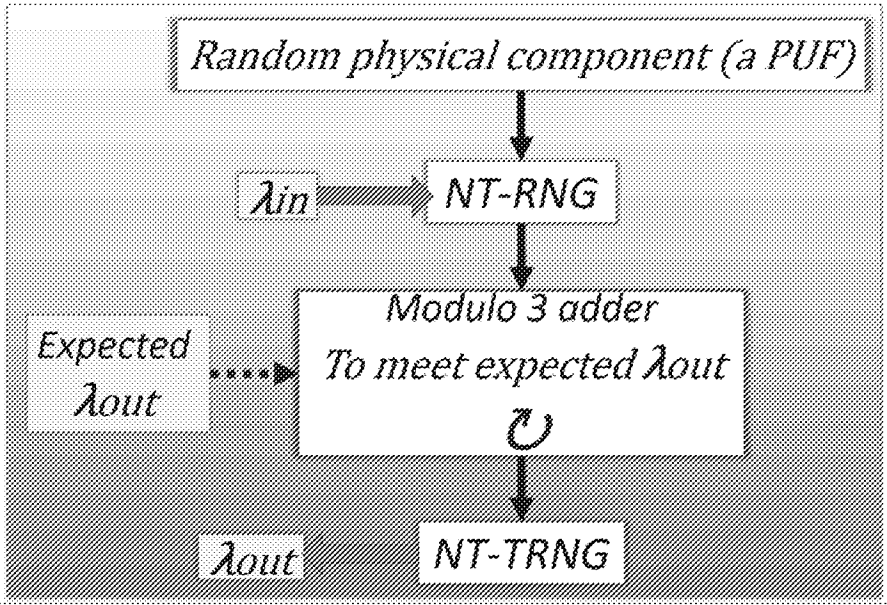


Fig. 1: Design of Native Ternary **T**rue **R**andom **N**umber **G**enerator

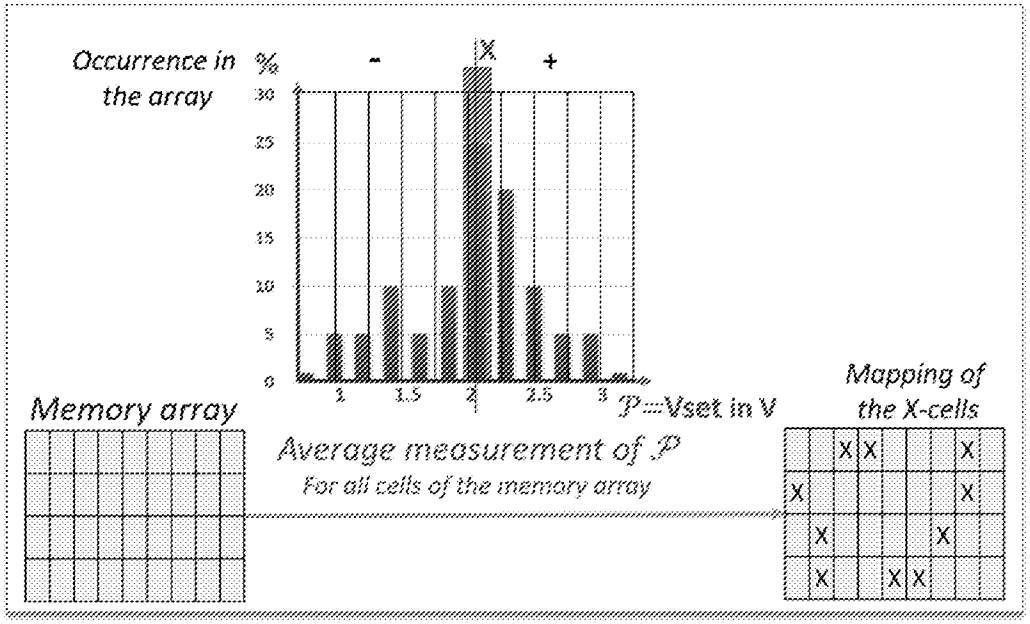


Fig. 2: X-cells - the marginal cells of a PUF

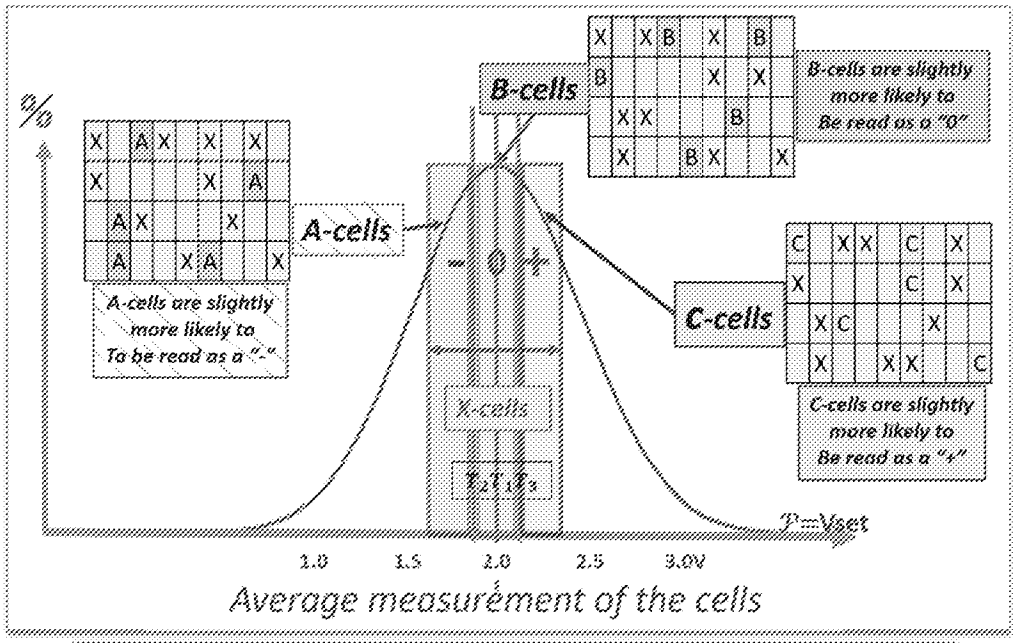
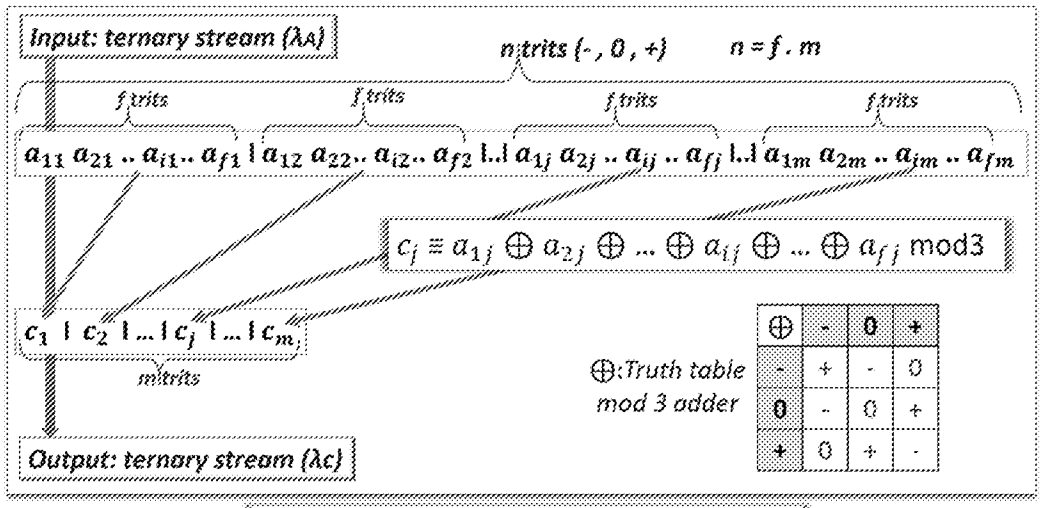


Fig. 3: X-cells sorted into three types: A (-); B(0) ; C(+)



⊕	-	0	+
-	+	-	0
0	-	0	+
+	0	+	-

Fig.4: Description of the Modulo 3 adder

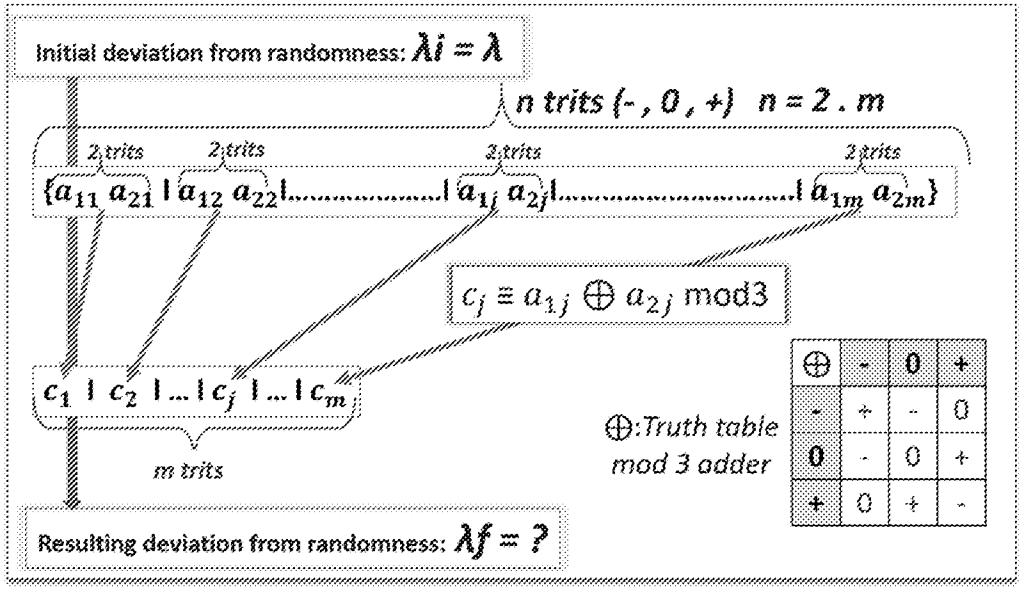


Fig.5: Simplified model - The mod 3 adder with 2 cells

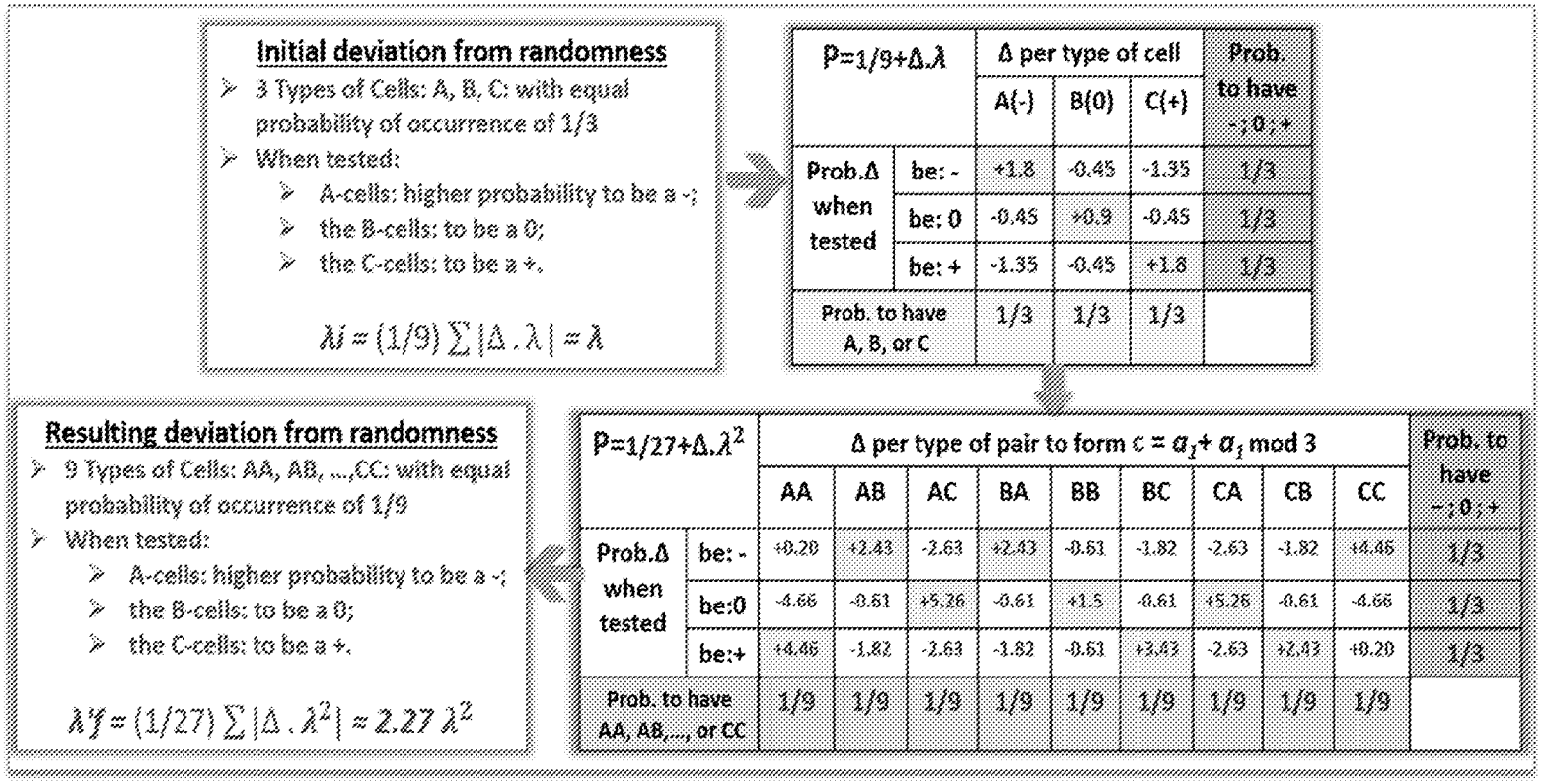


Fig.6: The mod 3 adder with 2 cells: impact on randomness

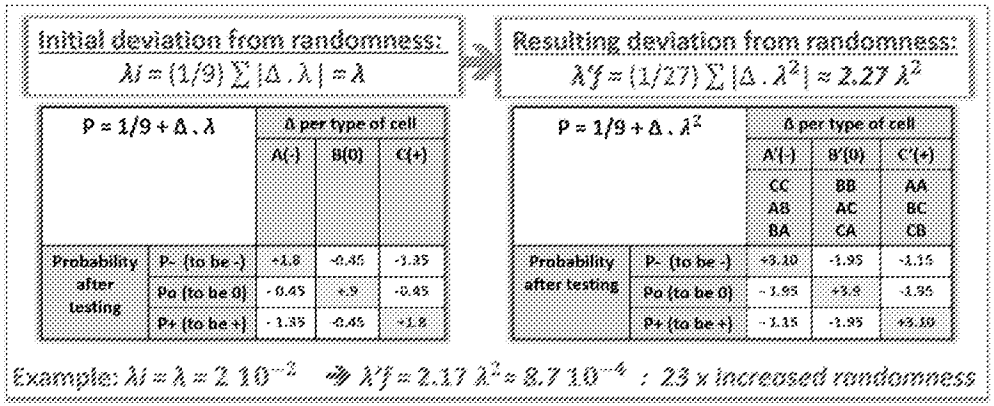


Fig. 7: The mod 3 adder with 2 cells: Grouping again in 3 types

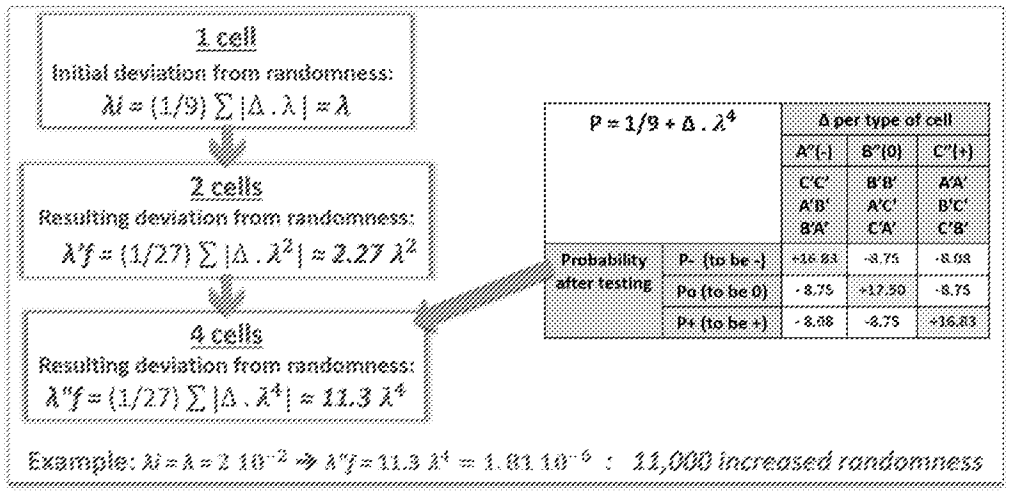


Fig.8: The mod 3 adder with 4 cells

Case	% of X-cells	$\lambda i = \lambda$	$\lambda' f \approx 2.27\ \lambda^2$ 2 cells	$\lambda'' f \approx 11.3\ \lambda^4$ 4 cells	$\lambda''' f \approx 290\ \lambda^8$ 8 cells
1	2%	$2\ 10^{-2}$	$8.7\ 10^{-4}$	$1.81\ 10^{-6}$	$7.4\ 10^{-12}$
2	4%	$4\ 10^{-2}$	$3.6\ 10^{-3}$	$2.9\ 10^{-5}$	$1.9\ 10^{-9}$
3	7%	$6\ 10^{-2}$	$8\ 10^{-3}$	$1.46\ 10^{-4}$	$4.87\ 10^{-8}$
4	11%	$1\ 10^{-1}$	$2.3\ 10^{-2}$	$1.1\ 10^{-3}$	$2.9\ 10^{-6}$

Fig.9: Level of randomness λ by experimental case

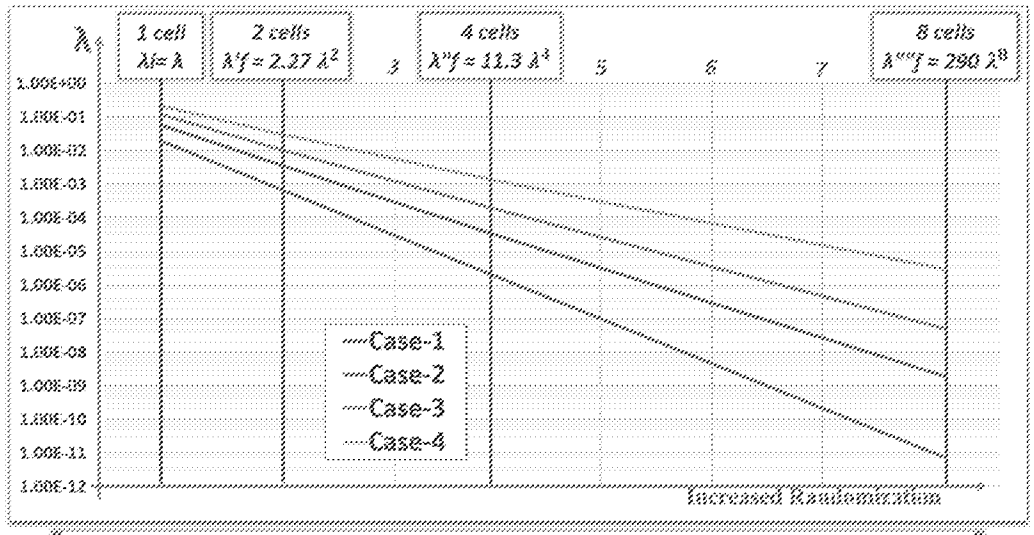


Fig. 10: Level of randomness **λ'f** , **λ"f** , and **λ""f** after mod3 addition