American Journal of Engineering Research (AJER)
e-ISSN: 2320-0847 p-ISSN : 2320-0936
Volume-12, Issue-5, pp-01-12
www.ajer.org
Research Paper Open Access

A High Frequency Memristor Model Applied to Hardware Security

J. Balaam Alarcón-Angulo[1] & A. Sarmiento-Reyes[1]
[1](INAOE, Puebla, México
jbalan@inaoep.mx & jarocho@inaoep.mx

ABSTRACT : *This paper proposes a novel mathematical memristive model that can operate at high-frequency signals and has been used to develop an RO-PUF system. For this purpose, a memristive ring-oscillator has been designed to generate the signal used for the creation of the physical unclonable function. Some of the current models used in electrical simulations of memristive systems are rigid in the scalability of the operating frequency, also are highly complex, which causes a higher use of computational resources at the moment of the simulation. The model used in this work allows frequency scalability and efficient use of computational resources.*

Date of Submission: 22-04-2023 Date of acceptance: 05-05-2023

I. INTRODUCTION

Since the conceptual postulation by Chua in the '70s [1] [2], followed by its manufacturing as an actual device by HP Labs [3] [4] in 2008, the memristor has attracted considerable attention for its use in different applications, such as; high-density memory, oscillators, chaotic systems, programmable analog circuits, and synapses in neuromemristive circuits. Unfortunately, the currents simulators do not include in their libraries a memristive system that can be used as a device, so it is necessary to develop memristive models that are compatible with the numerical analysis used by the simulators.

Through the years, several memristor models have been developed and along with them some circuit applications. These prototypes span from macromodels as introduced by Biolek in 2009 [5] to mathematical models like the Affan [6], and circuits emulators of memristance as implemented by Vista [7].

The disciplines in which memristor can be used have different areas of study, one of those areas is hardware security and the elaboration of physical unclonable function (PUF). Other works have explored the profits of using memory devices in PUF structure, Herder C. et al [8] explain the use of random-access memory (SRAM) in mobile systems for storage a secret key. Rose, G. [9] shows a write-time-based memristive PUF that leverages variability in the SET time of the memristor, and Gao, Y. demonstrate the use of memristor in PUF structure improves it, resulting in a reconfigurable PUF [10].

This work introduces a mathematical memristive model that can operate at high frequencies and is obtained by substituting the symbolic state-variable solution from HP's memristor into Strukov's coupled resistor model [3], giving rise to the memristor model that was used to build up a PUF based on a memristive ring-oscillator (RO).

The rest of the paper is organized as follows. Section II shows the development and evaluation of the mathematical memristive model. Section III presents an analysis to compare the generated RO-PUF with others PUFs. Finally, some conclusions are drawn in Section IV.

II. DEVELOPMENT AND VALIDATION OF THE MATHEMATICAL MEMRISTIVE MODEL

The mathematical memristive model has been developed starting from the definition of memristive system [2], which are nonlinear dynamical systems, defined by its realization in state space:

$$\dot{x} = f(x,u,t) \qquad (1)$$

$$y = g(x,u,t) \qquad (2)$$

where u and y are the input and output of the system, respectively, and x is the state-variable of the system. The HP memristor includes an equation of the state space [3] that meets the guidelines of Equation (1), and is defined by:

$$\dot{x} = \frac{\mu R_{on}}{\Delta^2} i(t) f_w(x) \qquad (3)$$

where Δ stands for the full length of the semiconductor material and $x(t)$ is the normalized state-variable $\left(x = {}^{\omega}/_{\Delta}\right)$. Besides, μ is the mobility of the charges, R_{on} is the ON-state resistance, and $f_w(x)$ is a window function that bounds the state-variable $x(t)$ and satisfies that $f_w(0) = f_w(1) = 0$ to guarantee no drift at the boundaries. The current is a stimulus function that in this case is a sinusoidal signal, given as $i(t) = Ap\sin(\omega t)$ where Ap is the amplitude, and ω is the angular frequency.

The $x(t)$ value is found by solving Equation (3) and using the homotopy perturbation method [11], [12], and [13], obtaining a fully symbolic solution to $x(t)$. The result was put in the electrical equivalent of memristance, proposed by Strukov [3], it consists of a series connection of two coupled resistors, the ON-state resistor R_{on}, and the OFF-state resistor R_{off}.

$$M(t) = R_{on}x(t) + R_{off}\big(1 - x(t)\big) \qquad (4)$$

Equation (4) gives the memristance at any time and depends on the $x(t)$ value. The result of the substitution of the state-variable is a fully symbolic mathematical memristor expression:

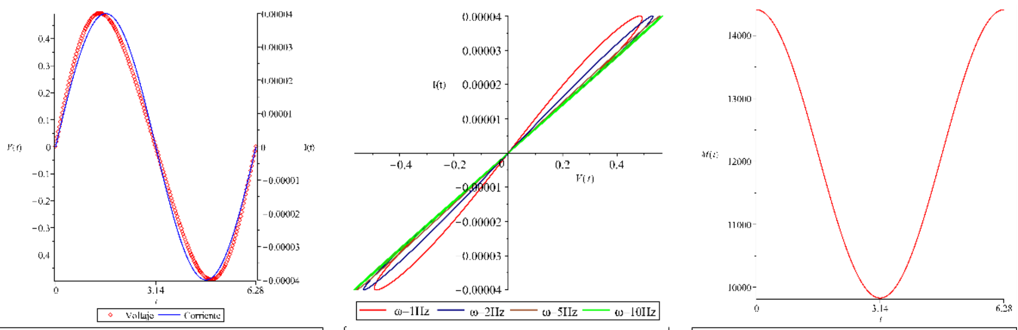
$$\begin{aligned} M_{O1} &= R_{on}^2 f_w \gamma(\alpha - 1)[-1 + \cos(\omega t)] + R_{init} \\ f_w &= f_w(X_o)|_{k=n} = 1 - (2X_o - 1)^{2k} \\ R_{init} &= [X_o + \alpha(1 - X_o)]R_{on} \\ \gamma &= \frac{\mu Ap}{\Delta^2 \omega} \end{aligned} \qquad (5)$$

where f_w is the Joglekar window function [14], R_{init} is the initial memristor resistance, with X_o as the initial condition of the state-variable, and γ is a constant that comes from Equation (3). **Table *1*** shows the typical values used in Equation (5). In previous work [15], the development of M_{O1} equation is explained in detail.

Table 1. Numeric values for M_{O1} equation.

$\mu\ \left({}^{cm^2}/_{sV}\right)$	Ap	$\Delta(m)$	$R_{on}\ (\Omega)$	X_o	k	α
10^{-10}	40×10^{-6}	10×10^{-9}	100	0.1	1	160

Equation (5) has been analyzed in Maple using the values of **Table *1*** and four values for the frequency to ensure that the model obtained using homotopy method complies with the fingerprints of the memristive systems [16]. **Fig. 1** shows the curves generated by the model.



(a) Current signal shape (blue line) vs voltage signal shape (red points).

(b) Pinched Hysteresis Loop generates by M_{O1} equation at different frequencies

(c) The memristance as a function time.

Fig. 1. Response of M_{O1} to the stimulus signal $Ap\,sin(\omega t)$.

Fig. 1(a) shows the voltage generated by the mathematical model (M_{O1}) when a stimulus signal $i(t)$ is applied, giving a current-controlled memristive model, the voltage value is obtained using the expression $v_M(t) = M_{O1}i(t)$ and satisfying the form of Equation (2), where $i(t) = Ap\sin(\omega t)$ acting as the stimulus signal, the memristor device M_{O1} is modeled by Equation (5), and the product will be the voltage $v_M(t)$ in any instant of time. Both signals coincide at the zero-crossing, the voltage (red points) has a deformation (in comparison to the current signal) causing the peak value between voltage and current to be at different time instant, which is a typical performance of a memristor.

One of the characteristics of the memristive systems in the time-domain is that their pinched hysteresis loop must pass through the intersection (0,0) of the $v - i$ characteristic curve, for any possible amplitude Ap and frequency ω, **Fig. 1**(b) shows the behavior of Equation (5) at different frequencies, is possible to see that the area of the waveform will decrease in size as the frequency increase, so when $\omega = \infty$ the memristor will change his behavior to resistive conduct, this kind of performance is present in memristive devices.

Finally, **Fig. 1**(c) shows the numerical value of M_{O1} as a function of time, the maximum value ($14.41K\Omega$) is reached when $t = 0$ ($t = {}^{2\pi}/_{\omega}$) causing that M_{O1} to depend only on R_{init} due to $\cos\left(t_o, {}^{2\pi}/_{\omega}\right) = 1$, and in the middle of the period ($t = {}^{\pi}/_{\omega}$), M_{O1} reaches its minimum value ($9.38K\Omega$), as the frequency increase the difference between both values is reduced. **Fig. 1** shows that the proposed model complies with the fingerprints mentioned in [16], and therefore Equation (5) is a memristive model.

A. High-Frequency HPM memristor model

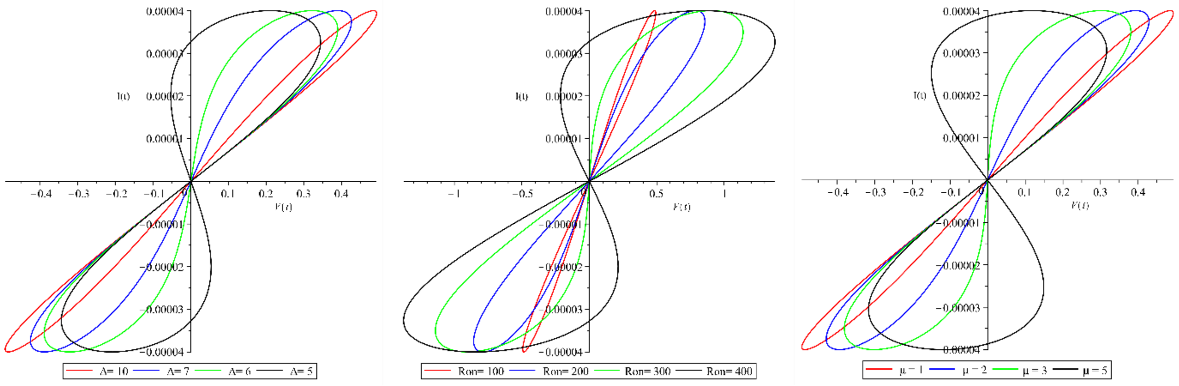
In this section, a methodology for scaling the operating frequency of the memristive model is proposed. It is known that the area of the PHL in inversely proportional to the operating frequency [16], such a relationship can be observed if the integral of the product between voltage and current is obtained.

$$A = \int_0^{\pi} v(t)i(t)dt \tag{6}$$

Solving the above equation for a half period of the input signal $i(t)$ and with $v(t) = v_M$, a symbolic expression for the area is obtained:

$$A = \frac{R_{on}Ap^2((\alpha-1)\mu Ap(-1+(2X_o-1)^{2k})R_{on} - ((X_o-1)\alpha - X_o)\Delta^2\omega\pi}{2\Delta^2\omega^2} \tag{7}$$

The result is a symbolic expression for the area of the PHL and depends on the same variables contained in the memristive model, it can also be seen that the area of the PHL will be reduced in proportion to ω^2, therefore, a variable to counteract the effect of ω will be selected. The variables Ap and X_o are not selected because they are: the amplitude of the stimulus signal and the initial condition of the state variable $x(t)$, with the latter in mind, the variables Δ, R_{on}, and μ are selected to change their values and see the model's response as a second step to scale the operating frequency.



(a) PHL when Δ changes, each value is by 10^{-9}.

(b) PHL when R_{on} changes.

(c) PHL when μ changes, each value

Fig. 2. Modification of the numerical value of the variables Δ (a), R_{on} (b), and μ (c), in Equation (5) at a fixed frequency of 1 Hz, resulting in an increase of the PHL area.

The result of changing the value of the variables Δ, R_{on}, and μ are shown in **Fig. 2**, is possible to see how the area of the PHL grows when the variables increase. The values of Δ, R_{on}, and μ are show in **Table *2***. The black PHL that appears in **Fig. 2** shows that at these values the model no longer behaves as a memristive system because it no longer behaves as a memristive system because it no longer meets the passivity criterion [16], when that happens it is possible to increase the operation frequency, causing the PHL area to reduce and when it reduces sufficiently it again meets the passivity criterion, scaling the operating frequency of the memristive system modeled by M_{O1}.

Table 2. Values of Δ, R_{on}, and μ, the maximum and minimum memristance and Equation (7) evaluated.

	Max.	Min.	Area
$\Delta_{red} = 10 \times 10^{-9}$	$14.41K\Omega$	$9.38K\Omega$	3.0461×10^{-5}
$\Delta_{blue} = 10 \times 10^{-9}$	$14.41K\Omega$	$5.06K\Omega$	2.4473×10^{-5}
$\Delta_{green} = 6 \times 10^{-9}$	$14.41K\Omega$	$1.69K\Omega$	2.0231×10^{-5}
$\Delta_{black} = 5 \times 10^{-9}$	$14.41K\Omega$	$-3.9K\Omega$	1.3199×10^{-5}
$R_{on_{red}} = 100$	$14.41K\Omega$	$9.38K\Omega$	3.0461×10^{-5}
$R_{on_{blue}} = 200$	$28.82K\Omega$	$10.5K\Omega$	2.4473×10^{-5}
$R_{on_{green}} = 300$	$43.23K\Omega$	$2.01K\Omega$	2.0231×10^{-5}
$R_{on_{black}} = 400$	$57.64K\Omega$	$-15.63K\Omega$	1.3199×10^{-5}
$\mu_{red} = 1 \times 10^{-14}$	$14.41K\Omega$	$9.83K\Omega$	3.0461×10^{-5}
$\mu_{blue} = 2 \times 10^{-14}$	$14.41K\Omega$	$5.25K\Omega$	2.4473×10^{-5}
$\mu_{green} = 3 \times 10^{-14}$	$14.41K\Omega$	672.4Ω	2.0231×10^{-5}
$\mu_{black} = 5 \times 10^{-14}$	$14.41K\Omega$	$-8.49K\Omega$	1.3199×10^{-5}

The first column of **Table 2** shows the variable with a color name as a subindex, this is to indicate the relation between the value of the variable and the PHL of ***Fig.*** *2*, the second column shows the value of the maximum memristance reached by Equation *(5)*, which occurs when $t = 0$, another observation is that the value of the maximum memristance remains constant despite the change in the value of the variables Δ and μ, the third column of **Table 2** shows the minimum memristance reached by the M_{O1} model, which occurs when $t = {}^{\pi}/_{\omega}$, in these case, the value of minimum memristance reached by the model is decreasing for almost all cases, except when R_{on}=200, this behavior is because the of minimum memristance in any time is given by ${d(PHL(t))}/_{dt}$, the minimum value is when $t = {}^{\pi}/_{\omega}$, being this case one with the opposite slope to the rest of the PHL curves, this is reflected in the green curve of *Fig. 2(b)*. Finally, the fourth column shows each value of Equation *(7)* evaluated, showing that in effect, the area of the PHL generated by M_{O1} model, increases when Δ and R_{on} increases, and when μ decreases the area still increases.

B. HFHPM vs other models

After verifying that Equation **(5)** (from now on called HFHPM) works as a memristive system, it is introduced in a subcircuit in HSpice to ensure the correct function in electric simulation, and compare its performance against other models, **Fig.** 3 shows the circuit used to simulate the models.

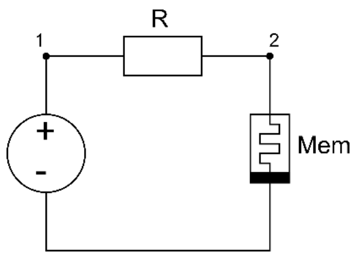


Fig. 3. Voltage divider used to obtain electric waveforms from a memristive model.

The models were simulated into the voltage divider in **Fig. 3**, in [17] explains that the most general-purpose computer simulation programs are made up of five main stages: an input stage, a device model retrieval and replacement stage, an equilibrium formulation stage, a numerical solution stage, and an output stage. When a memristor model includes integrals (Biolek), square root function (Affan), or, piecewise (HPMQ), the equilibrium equation formulation stage includes discretized non-linear nodal and hybrid equations, non-linear state equations, and tableau equations making to solve the equations takes more processing time, against to linear equations that can be solved using complex Gaussian elimination.

After doing the simulations, HSpice show three values: $CPU\ time$, $Elapsed\ time$, and $Peak\ Memory$. The simulation values for each model are shown in **Table** ***3***, the second column shows the processing time that takes to simulator do the simulation of **Fig. 3**, is possible appreciate that the HFHPM model has the lowest $CPU\ time$ value at moment to do the simulation due to is easier to evaluates Equation (5) in the Numerical Solution Stage.

Table 3: Computational resources used by the proposed model during the simulation (Fig. 3), compared against the Biolek *[5]*, **Affan** *[6]*, **MOS-Mem** *[7]*, **and HPMQ** *[18]***models.**

$\mu({}^{cm^2}/{}_{sV})$	Ap	$\Delta(m)$	$R_{on}(\Omega)$	X_o	k	α
10^{-10}	40×10^{-6}	10×10^{-9}	100	0.1	1	160

Since the memristor is a no-lineal device the simulator can´t use Complex Gaussian elimination, therefore the performance of model is evaluated using the number of nodes, as is well known, in a circuit with n nodes there are $n - 1$ linearly independent equations to be solved, the argument is derived from Kirchhoff's law. When the number of nodes increases, the equations to be solved increase so the computing time increases, for this reason, a memristor model that introduces more than two nodes (Biolek and MOS-Mem models as an example) in the simulation makes the $CPU\ time$ value rise. Due to this relation, the next function is proposed as a first step to the normalization of the simulation time ($CPU\ time$).

$$\Psi = \frac{\phi}{(\varsigma - \varrho) + 1} \qquad (8)$$

where ϕ is the value contained in the second column of Table 3, ϱ is the degree of the MNA matrix obtained from the original topology (Fig. 3), and ς is the degree of the MNA matrix after introducing the model into the voltage divider. Equation (8), is used to show the ratio between the simulation time and a memristive device model when is implemented in an electrical simulator.

Now Equation (8) is used in the practical case for obtaining the electric waveforms in the simulation of a memristive model. The circuit used is shown in **Fig. 3**. The value of $\varrho = 2$, due to the MNA dimension being 2×2, the value of ς will change depending on the model used. Having all the values of the variables present in Equation (8) we can evaluate it. **Table *4*** shows the function Ψ evaluated for all models and the value of ς for each model.

Table 4: Evaluation of *Ψ* for the memristor models mentioned in the paper.

Model	ς	Ψ (seconds)
Biolek	6	0.008
Affan	2	0.04
MOS-Mem	5	0.2
HPMQ	2	0.17
HFHPM	2	0.03

Table 4 shows the ratio time (Ψ) between the number of nodes in the simulated circuit and the processing time ϕ, if ς=ϱ then $\Psi = \phi$, this is because these models do not add nodes to the simulation. When the memristor model adds nodes to the simulation, the processing time rises, altering the value of ϕ. Table 4 shows that the Biolek model has the greatest value modification, and the reason is that it is the model that adds more nodes in the simulation, Equation (8) shows quantitatively the change that a memristor model generates in an electrical simulation. The second column in Table 4 shows the ς value for each model.

Further on, when the simulation time will be normalized, the disadvantages of a memristive model that adds nodes in the electrical simulation are explained.

C. Memristive Ring-Oscillator

The Ring-Oscillator used in this work was elaborated using five logic inverters connected in cascade, **Fig. *4*** shows the structure of the circuit, inverter is an NMOS memristive load inverter.

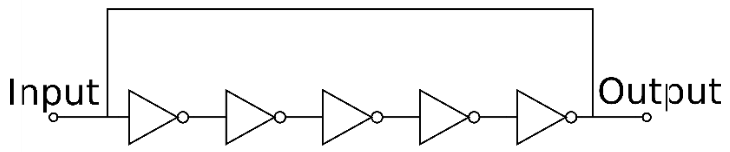


Fig. 4. Five Stage Ring-Oscillator structure.

The topology of the NMOS memristive load inverter is shown in **Fig. 5** where the input voltage (V_{in}) is equal to the gate to source voltage (V_{GS}) of NMOS and the output voltage (V_{out}) is equal to the drain to source voltage (V_{DS}) of the transistor. Here, enhancement type NMOS act as the driver transistor. The load consists of a memristor device, the power supply of the circuit is V_{dd} and the drain current i_{Drain} is equal to the memristor current $i_{memristor}$.

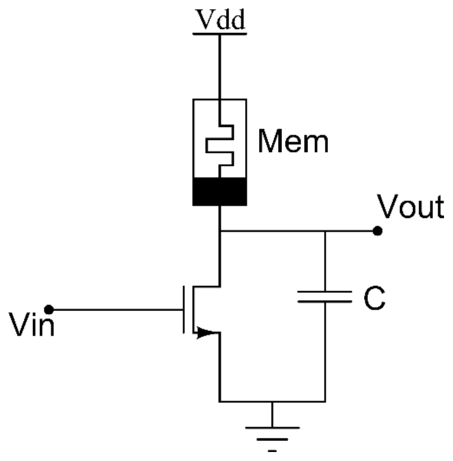


Fig. 5. Inverter used in the Five Stage Ring-Oscillator.

The value of the capacitor **C** that appears in **Fig. *5*** is $100fF$, the transistor ratio $W/_L = 10$, and the simulation was made using the fives models that appear in **Table *4***.

Continuing with the methodology of Section II-B, five simulations are made, one for each memristor model. **Table *5*** shows the computational resources that need each RO structure when a model is uploaded in the simulation of **Fig. *4***.

Table 5: Computational resources used in the RO simulation for each memristor model.

Model	CPU time (seconds)	Elapsed time (seconds)	Peak Memory (megabytes)
Biolek	1864.9	1922.05	210.99
Affan	1851.115	1909.78	209.98
MOS-Mem	0.56	1.15	211.97
HPMQ	712.04	744.09	212.98
HFHPM	0.17	0.63	210.97

Table *5* shows the computational resources used in each simulation, the second column shows the total time spent by the CPU resources on a server (CPU time), the third column displays the total time taken by the SQL server and the last column shows the memory used by the server. In **Table *3***, the CPU time values between the models differed by one order of magnitude at most, but when the number of memristor devices increases, some models need more simulation time, and therefore the CPU time values in **Table *5*** increase.

One of the things shown in **Table *5*** is that the models of Affan, Biolek, and HPMQ have large difference between their CPU time values, instead, the HFHPM model and the emulator don't show a difference of more than one order of magnitude between their CPU times values.

As can be seen, the CPU times values between both simulations (**Table *3*** and **Table *5***) increase considerably for almost all models, this is due to the increase in the MNA matrix or because of the set of non-linear equations introduced by the memristor model. Since, the simulation time significantly changes between models and circuits, for at leas two reasons (increase the size of the MNA matrix and/or the complexity of the equation), the following benchmark FOM is proposed:

$$FOM = \frac{\Psi_b}{\Psi_a} \qquad (9)$$

where Ψ_a and Ψ_b comes from Equation (8), with the difference that the values used in Ψ_b were taken from the RO simulation (**Table *6***) and the values used in Ψ_a are taken from **Table *4***. The second column in **Table *6*** shows the values of ς_b that were used in Ψ_b, and the last column shows the evaluation of Equation (9).

Table 6: Benchmark FOM values for each memristor model.

Model	ς_b	Frequency variation	FOM
Ideal	7	N/A	1
HFHPM	7	Yes	1.133
HPMQ	7	Yes	8376.9412
Affan	7	No	9255.75
MOS-Mem	27	Yes	0.1663
Biolek	32	No	10788.9589

The fourth column in **Table *6*** shows the FOM evaluated for six cases, the ideal case is when $\Psi_b \approx \Psi_a$, so Equation (9) can be approximated to 1, this is considered due to that ideally any model should increase the size of the MNA matrix. Therefore, if the value of the FOM for the memristive model deviates from the ideal value, this model has a high complexity for the simulator, and that is a disadvantage when a circuit design contains several memristors.

Now the difference between the simulation time of each model is more easily noticed. The HFHPM model requires less simulation time thanks to being created from the symbolic solution given by the homotopy method and other models find the solution of integrals or square roots, once for each memristive device and for that reason the proposed model beats the others in simulation time, therefore the rest of the work the HFHPM model is used.

One of the bases for the PUF design is to use the process variation present at the moment of fabrication. The HFHPM model allows emulating these variations when the parameter X_o in Equation (5) is modified.

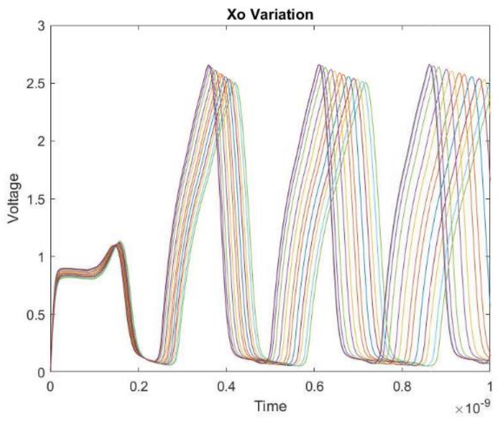


Fig. 6: Frequency variation generated in the RO changing the X_o values.

The RO-PUF works by comparing two signals from two RO circuits designed exactly alike, but due to process variation, its oscillation frequency is slightly different. **Fig. *6*** shows 13 different signals from five stage ring-oscillator using the HFHPM model and changing the X_o value in the model, the frequency changes slightly. The values for X_o, are shown in

Table 7: X_o values used in the HFHPM model, generating different output signals from the five-stage ring-oscillator.

X_o	0.05	0.07	0.1	0.12	0.15	0.18	0.2	0.23	0.25	0.28	0.3	0.312	0.315

The values that X_o takes, make that the model star whit different memristance value, causing that the frequency of the RO change slightly, is important to said that before of 0.05, and after of 0.315 the RO doesn't work correctly.

III. RO-PUF Implementation

A physical unclonable function (PUF) is a system that for a given signal and condition (usually called challenge), provides a physically defined "digital fingerprint" output (called response) that serves as a unique identifier. **Fig.** *7* shows a general scheme of the operation of a PUF. Exploiting the process variation present in N RO designed with the same specification generating cryptographic keys, and device validation.

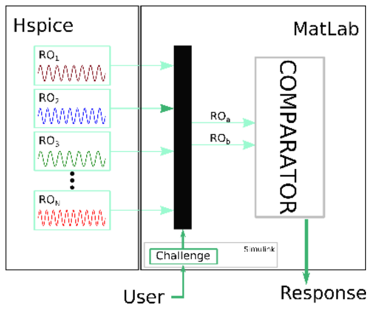


Fig. 7. RO-PUF scheme.

Fig. *7* shows that the process starts with N RO signals, the user selects a challenge and depending on the selection the system chose and processes two signals using a comparator, generating the response to the selected challenge, this methodology is called challenge-response pair.

The challenge-response pair method consists of the user choosing from a finite number of options, those options are programmed by the system designer, and depending in the selected challenge, the system gives a response (all the system is the PUF), the number of responses that the PUF can give also is finite and includes two RO signals, which are pseudo-randomly combined [19].

For this case of study, the challenge function creates an n-bit array, this array will be used to accommodate the result of the comparison between both signals of the ring-oscillator, **Fig.** *8* shows the scheme of the challenge function.

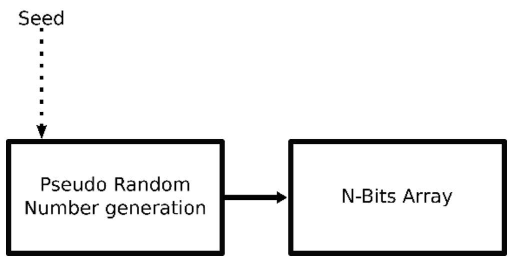


Fig. 8. Structure of the challenge function.

The idea of the challenge is to use a seed to create a pseudo-random number between 1 and N, where N is the number of bits calculated by the PUF, the generated number is collocated into the array, and this action is repeated n times until the array is filled, the internal order of the number in the array is different.

When the challenge function is ready, the RO-PUF is created, taking the files *TR0* that were generated in the electrical simulation, and uploaded into MATLAB code, here the program selects two of the thirteen files, one for each value of X_o given in **Table 7**, and passes through a counter, with it the combination of the two signals is made giving, as a result, an RO-PUF of 45 bits.

Each response generated is a unique arrangement of 1's y 0's, which are obtained by comparing the frequency of two oscillators. The output (response) generated by the challenge selection must be evaluated to guarantee that each challenge has only one response. The parameters to be evaluated are Bit-Aliasing, Uniqueness, and Uniformity.

Uniformity is used to measure the 1's and 0's distribution in the response. Ideally, the value must be 50% for a truly random PUF response.

$$Unifomity_i = \frac{1}{n}\sum_{i=1}^{m} R_{i,m} \times 100\% = 50\% \qquad (10)$$

where $R_{i,m}$ is the m-th bit of an m-bit response, Equation (10) is used to show that the response generated by the challenge has the same amount of 1's and 0's. **Fig. 9** shows the uniformity of the 10 responses obtained from the 10 challenges.

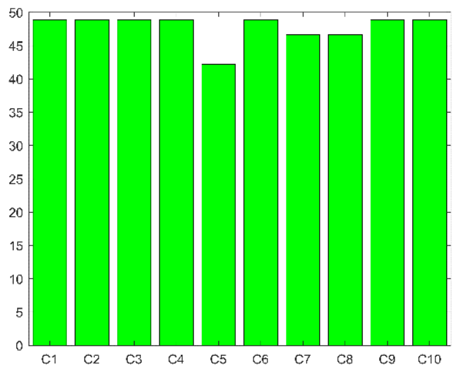


Fig. 9. Uniformity of each challenge-response pair.

Uniqueness represents the ability of a PUF to distinguish between a particular device among group of devices of the same characteristics. The Hamming Distance (HD) between a pair of responses is used to evaluate uniqueness. If two responses, *i* and *j* ($i \neq j$), have m-bit string R_i and R_j respectably for the challenge, the average HD among *k* ROs is defined as:

$$Uniqueness = \frac{2}{k(k-1)}\sum_{i=1}^{k-1}\sum_{j=i+1}^{k}\frac{HD[R_i, R_j]}{n} \backslash times\ 100\% \qquad (11)$$

this parameter indicates how much the bit m from the response R_i match with the bit m from the response R_j.

Bit-Aliasing is a measure of biasness in the response, a few bits if the response are stuck to '0' if biased to zero or stuck to '1' if biased to one. If Bit-Aliasing happens, a different device may produce a nearly identical

PUF response which is an undesirable effect. The Bit-Aliasing is estimated by taking the m-th bit in the PUF identifier as the percentage Hamming Weight (HW) of the m-th bit across *k* devices.

$$Bit - Aliasing = \frac{1}{k}\sum_{i=1}^{k} R_{i,m} \times 100\% \qquad (12)$$

where $R_{i,m}$ is the m-th binary bit of an n-bit response from device *i*.

Each response given is related to the challenge selected by the user, here it is easier to see the influence of the challenge in the response, the order of each bit of the response depends on the order of the number in the array generated by the challenge. As explained above it is necessary to evaluate the PUF mathematically to ensure that the distribution of 1's and 0's is uniform, **Table *8*** shows the result of the evaluation.

Table 8: Evaluation of RO-PUF system using the HFHPM model against the proposal of Sahoo *[20]* **and Maiti** *[21]***.**

	CMOS CRO-PUF	FTL CRO-PUF	Mixed Logic PUF	RO-PUF	AP-PUF	This work
Uniformity	46.75%	48.12%	47.5%	50.56%	55.69%	48.32%
Uniqueness	40.76%	44.15%	43.14%	47.24%	7.20%	52.56%
Bit-Aliasing	46.75%	48.12%	47.5%	50.56%	19.57%	48.32%

An ideal PUF system has a value for the three parameters of 50% the reason is that evaluation gives the probability of obtaining an equal number of 1's y 0's in each array. **Table *8*** shows the evaluation of the RO-PUF created, compared with the other models, this work presents a near value of 50%, checking in this way that the proposed model has good performance in the creation of PUF.

IV. CONCLUSION

We present a new mathematical memristive model that can scale in frequency, also allows electrical simulation, and consumes less computational resources than the common models.

The HFHPM model is compatible with the steps for the design of a resistive load inverter, allowing the option to redesign the inverter, if necessary, of if the user wants to optimize the inverter.

The inverter was used to make five-stage ring-oscillator, due to the HFHPM model allowing to modify of a parameter, and with this emulating the process variation, the oscillator is viable for PUF use.

Also, we create a challenge function in Simulink and thus we programmed the response in MATLAB, and with this, the evaluation of the RO-PUF was made, obtaining a near value to the ideal.

REFERENCES

[1]. L. O. Chua, "Memristor - The Missing Circuit Element," IEEE Transactions on Circuits Theory, vol. 18, no. 5, pp. 507-519, 1971.
[2]. L. O. Chua and S. M. Kang, "Memristive devices and systems," Proceedings of the IEEE, vol. 64, no. 2, pp. 209-223, 1976.
[3]. D. B. Strukov, G. S. Snider, D. R. Stewar and R. S. Williams, "The missing memtitor found," Nature, vol. 453, pp. 80-83, 2008.
[4]. M. Hopkin, "Found: the missing circuit element," Nature, 2008.
[5]. Z. Biolek, D. Biolek and V. Biolkova, "Spice model of memristor with nonlinear dopant drift.," Radioengineering, vol. 18, no. 2, 2009.
[6]. A. Z. Mohammed, Memristor Circuits and Systems., King Abdullah University, 2015.
[7]. J. Vista and A. Ranjan, "A simple floating mos-memristor for high-frequency applications.," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 27, no. 5, pp. 1186-1195, 2019.
[8]. C. Herder, M.-D. M. Yu, F. Koushanfar and S. Devadas, "Physical unclonable functions and applications: A tutorial.," Proceedings of the IEEE, vol. 102, no. 8, pp. 1126-1141, 2014.
[9]. G. S. Rose, N. McDonald, L.-K. Yan and B. Wysocki, "A write-time based memristive puf for hardware security applications.," In 2013 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 830-833, 2013.
[10]. Y. Gao, D. C. Ranasinghe, S. F. Al-Sarawi, O. Kavehei and D. Abbot, "Memristive crypto primitive for building highly secure physical unclonable functions.," Scientific Reports, vol. 5, no. 1, 2015.

[11]. J. H. He, "Homotopy perturbation method: a new nonlinear analytical technique.," Applied Mathematics and computation, vol. 135, no. 1, pp. 73-79, 2003.
[12]. A. Sarmiento-Reyes, L. Hernández-Martínez, H. Vázquez-Leal, C. Hernández-Mejia and G. U. D. Arango, "A fully symbolic homotopy-based memristor model for applications to circuit simulation.," Analog Integrated Circuits and Signal Processing, vol. 85, no. 1, pp. 65-80, 2015.
[13]. H. Vazquez-Leal, "Generalized homotopy method for solving nonlinear differential equations.," Computational and Applied Mathematics, vol. 33, no. 1, pp. 275-288, 2014.
[14]. Y. N. Joglekar and S. J. Wolf, "The elusive memristor: properties of basic electrical circuits.," European Journal of Physics, vol. 30, no. 4, pp. 661-675, 2009.
[15]. A. Sarmiento-Reyes and J. B. Alarcón-Angulo, "Nullor-Based Negative-Feedback Memristive Amplifiers: Symbolic-Oriented Modelling and Designs," in Pathological elements in analog circuit design, Springer International Publishing, 2018, pp. 329-360.
[16]. S. P. Adhikari, M. P. Sah, H. Kim and L. O. Chua, "Three fingerprints of memristor," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 60, no. 11, pp. 3008-3021, 2013.
[17]. L. O. Chua and P.-M. Lin, computer-aided analysis of electronic circuits algorithms and computational techniques, Prentice-Hall., 1975.
[18]. J. M. Germán-Martínez, Memory circuits for hardware security applications. [Maste'rs thesis], Puebla, México: Instituto Nacional de Astrofísica Óptica y Electrónica, 2020.
[19]. C. A. v. T. Henk and S. Jajodia, Encyclopedia of Cryptography and Security, New York: Springer New York, 2011.
[20]. R. S. Sauvagya, K. Sudeendra and K. Mahapatra, "A modified sonfigurable ro puf with improved security metrics.," 2015 IEEE International Symposium on Nanoelectronic and Information Systems, pp. 320-324, 2015.
[21]. A. Maiti, J. Casarona, L. McHale and P. Schaumont, "A large scale characterization of ro-puf," 2010 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST), pp. 94-99, 2010.