Random Number Generation using Chaotic Circuits

Hrishi Olickel
Yale-NUS College

In this paper, the use of a chaotic circuit - namely, Chua's Circuit - is explored as a possible method of random number generation. Using the chaotic nature of the output and the sensitive dependence on initial conditions, random binary bits were generated and compared against the output of known state-of-the-art Pseudo-Random-Number-Generators (PRNGs) and with those of a True-Random-Number-Generator (TRNG) using known statistical tests for randomness. The results were used understand the randomness of the values generated, and to draw conclusions for the possible application of Chua Circuits as RNGs for encryption.

A dynamical system can be considered chaotic if it satisfies the following criteria:

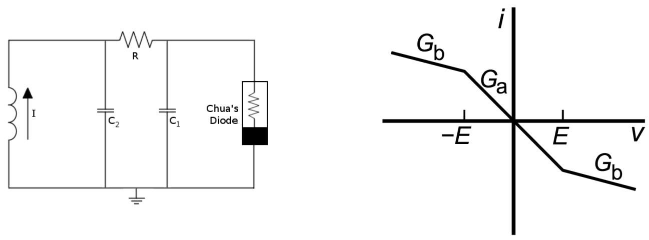
- It is densely periodic in its orbits.
- It is topologically mixed, i.e. the orbits frequently oscillate between topological states.
- It is sensitive to initial conditions.

Here, we exploit the first and the third characteristics in order to generate random numbers from an electrical circuit modelling a chaos attractor. In the background, we will be looking at how the circuit works, along with details specific to the implementation used here. We then consider the transformations applied to the raw data collected, along with mathematical reasons for these transformations. Finally, we perform three separate tests of randomness on the data collected, and compare the results with those of a known PRNG (Pseudo-Random Number Generator) and TRNG (True Random Number Generator) each.

Background

Chua's Circuit (Matsumoto, 1984) was conceptualised by Leon Chua in 1983. Brought to operational status by Takashi Matsumoto, the circuit is one of the most resilient and easily constructed examples for chaos in electronics. As we can see from figure 1 (a), the circuit contains four linear, passive elements and a non-linear element called the Chua diode.

Figure 1. (a) Simplified Schematic (b) IV Characteristic



The Chua Diode has a non-linear Voltage-Current characteristic, which is most commonly modelled as seen in figure 1 (b). The inductors, capacitors and resistance used are of standard values. The Chua Diode is the only active element in this circuit, i.e. it is the only element capable to injecting energy into it.

The circuit can be modelled by a three-dimensional system of Ordinary Differential Equations (ODEs). Considering g(x) to be the piecewise function that models the IV characteristic of the Chua Diode, we get the following:

$$g = \begin{cases} G_a V_1 + (G_a - G_b)E_1, & \text{if } V_1 \leqslant -E_1 \\ G_b V_1, & \text{if } -E_1 < V_1 < E_1 \\ G_a V_1 + (G_b - G_a)E_1, & \text{if } E_1 \leqslant V_1 \end{cases} \tag{1}$$

Using this equation for g, we can describe the circuit in terms of the voltages across capacitors 1 and 2 (figure 1 (a)) and the current across the inductor using Kirchhoff's Laws:

$$V_1 = \frac{V_2 - V_1 - Rg}{RC_1} \tag{2}$$

$$V_2 = \frac{V_1 - V_2 + Ri_L}{RC_2} \tag{3}$$

$$i_L = \frac{-V_2}{L} \tag{4}$$

We will soon model these equations in MATLAB and consider the output of the circuit from a mathematical perspective, once the details of implementation are made clear.

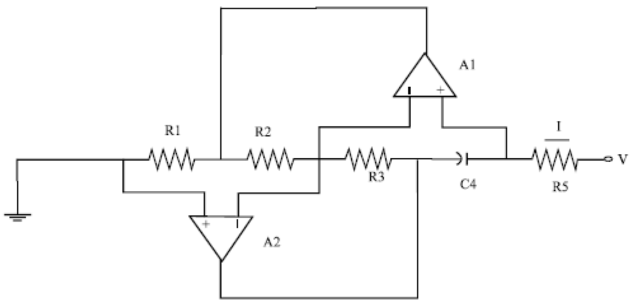
Implementation

It was decided that an Antoniou-Inductance Simulation Circuit - using an op-amp - would be the best method of implementing the inductor in the Chua circuit. The unstable nature of the circuit calls for a variable inductor that can be tuned. The sensitive nature of the circuit also means that the inductor must be immune to external noise and interference. While it was possible to conduct the experiments in a Faraday cage, the simulator circuit was chosen for the following properties:

- The simulator models an ideal inductor (Siderskiy, 2000), which brings the implementation closer to the ideal case.
- The circuit is completely solid-state. Moreover, it could be incorporated into the rest of the circuit such that the entire circuit only makes use of a single chip, reducing point of failure.
- The simulator is immune to noise and interference.
- The value of the inductor can be tuned using a variable resistor, and this cases a linear change in output inductance.
- The simulation circuit is significantly cheaper than a variable inductor of similar precision.

The schematic for the simulator is presented in figure 2 (a).

Figure 2. Antonio-Inductance Simulation Schematic



The simulated inductor can be considered equivalent to an inductor of inductance L, the relationship to which is expressed below:

$$L = \frac{C_1 R_1 R_3}{R_2} \tag{5}$$

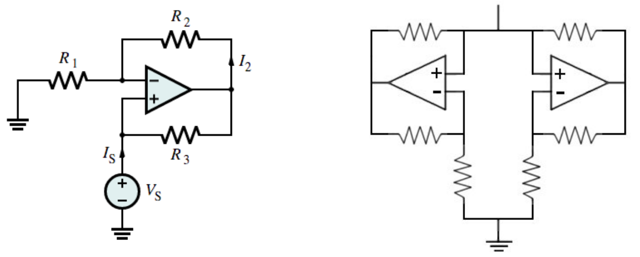
The complete explanation for this can be found in (Siderskiy, 2000), but the simple explanation is that the operational amplifiers in the circuit maintain a current I such that the voltage drop between R1 and R2 combined is zero, same as the drop between R3 and C4. This leads to a virtual ground appearing at these points, which leads to behaviour similar to an ideal inductor with respect to change in V.

Regarding the implementation of the Chua Diode, let us consider the IV Characteristic in figure 1 (b) again. You will notice that the element has negative resistance. The simplest way of modelling this is to use an operational amplifier (op-amp) in a negative impedance converter (NIC) (W.K, 2003) configuration, which will inject energy into the circuit w.r.t a change in voltage. Figure 3 (a) shows one possible configuration where the op-amp acts as a negative impedance converter. In this configuration, the negative resistance is modelled using the following equation:

$$-I_s = V_s \frac{\frac{R_1}{R_2}}{R_3} \tag{6}$$

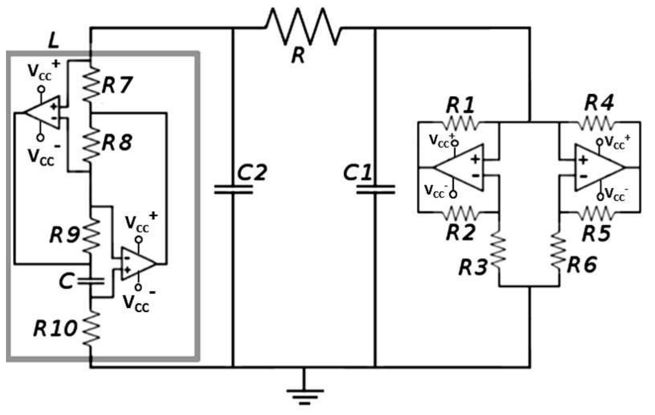
It is possible to connect two such circuits in parallel which are tuned such that the parallel sum of the two negative resistances provide the non-linear characteristic of the Chua Diode, as shown in figure 3 (b). You will notice no external components, merely that the NICs are upside down. The internal dynamics of the NICs are quite complex and significantly long, and for the purposes of this report we will assume that the IV characteristic is equivalent to the ideal case displayed in figure 1 (b).

Figure 3. (a) NIC Schematic (b) Chua Diode Implementation



The rest of the components are sufficiently simple that we can implement them using off-the-shelf, sometimes low precision components. The completed schematic can be seen in Figure 4. There are two variable resistors we use for tuning. The first is R10, which controls the inductance value of the simulated inductor. The second is R and controls the chaotic properties of the circuit.

Figure 4. Complete Circuit Schematic



We can see that the completed schematic maintains the simplicity of the circuit while gaining precision and reliability on conventional components. It is also interesting to note that all components in this circuit can be etched in silicon, making for a completely integrated circuit.

Figure 5 and 6 show the completed and functional circuit. It is worth noting that the circuit is battery powered to increase reliability and prevent transients from mains power supplies. New batteries were used for each test case to prevent the voltage discharge curve of the batteries from affecting the circuit.

Figure 5. Chua Circuit Top View

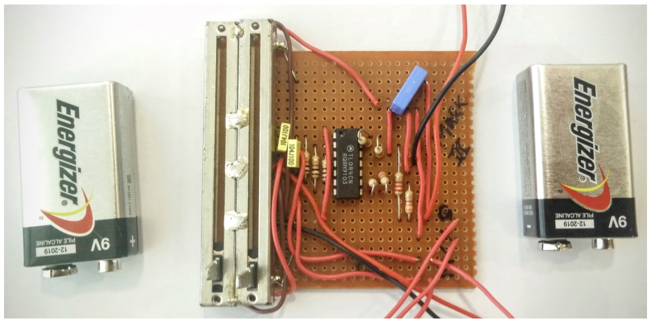
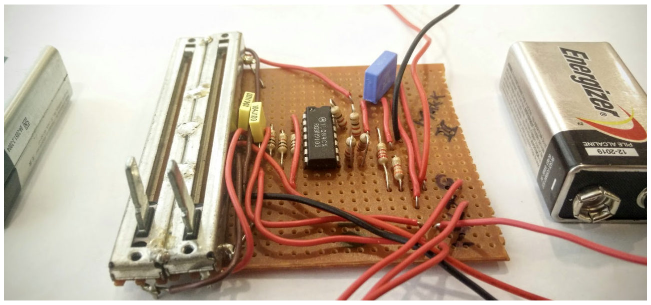


Figure 6. Chua Circuit Lateral View

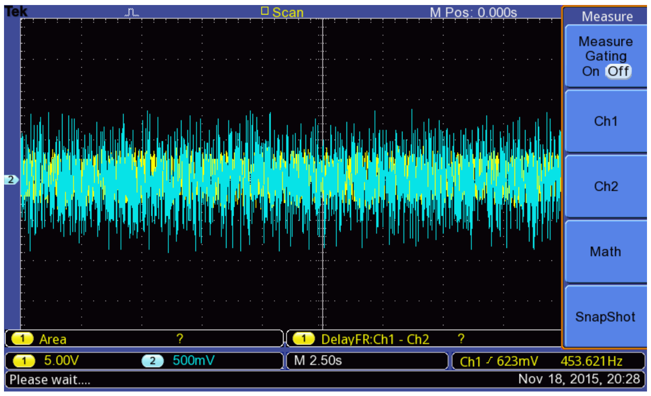


In order to collect data for the X, Y and Z axes, the voltage across C_1, C_2, and the voltage at the point between R_7 and R_8 was used, respectively.

Data Collection

Once the circuit was completed, data was collected from the X and Y axes, through the use of an oscilloscope[1]. The collected waveform - before it was exported - can be seen in figure 7.

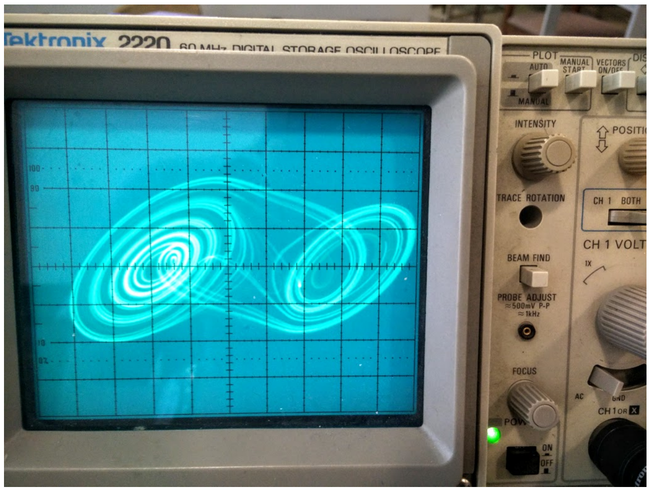
Figure 7. Oscilloscope Screen Capture



An analog oscilloscope was used to visualise the double scroll attractor, and this can be seen in Figure 8. A graph (Figure 9 (a)) was plotted from collected values, to check the integrity of the numbers, and the same scroll was found, with the same parameters.

Next, we compare the observed values to a mathematical model generated using equations (2), (3), and (4)[2]. Figure 9 (b) demonstrates that the underlying model is still the same. We can now perform the final calibration using the observed and expected operational parameters (span, position of attractors etc). Having confirmed observational values from two independent oscilloscopes and the mathematical model, we continue on to the analysis of the data.

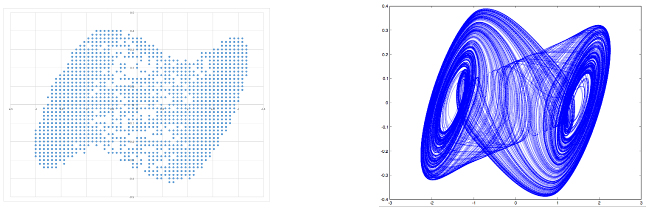
Figure 8. Visualisation of the Lorentz attractor



[1]The collected data can be found at https://github.com/hrishioa/IS2EE/tree/master/Final%20Project/Ra

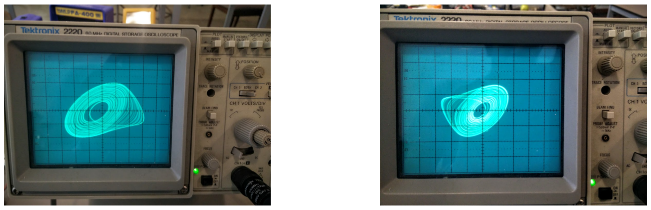
[2]The code for the MATLAB simulation can be found at https://github.com/hrishioa/IS2EE/blob/master/Final%20Project/ch

Figure 9. (a) Graph of Y signal against X signal (b) Computer Generated Graph of Y against X



Some interesting features were observed here. Since the circuit contains variable components, it was possible to adjust the position of the variable resistors to witness the onset of chaotic behaviour. Before this occurs, the system remains in a closed attractor, either skewed to the left or to the right. Both cases can be seen in Figure 10. When we decrease the inductance such that the system falls out of chaos, it unpredictably ends up in either left or right configuration. This is an indication of the random behaviour that we will seek to exploit soon.

Figure 10. (a) Left attractor (b) Right Attractor



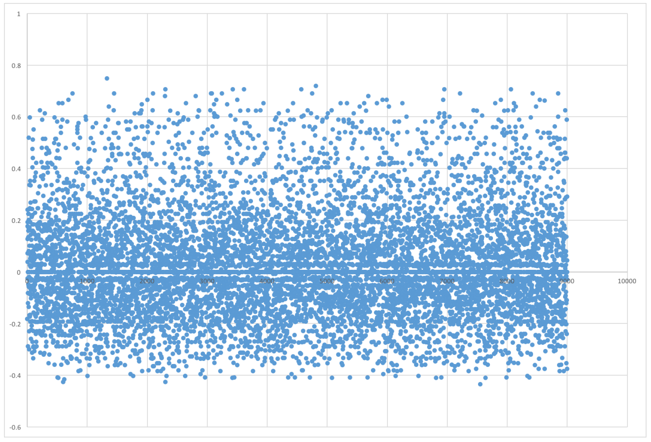
Data Analysis

At this point, we have a stream of numbers collected from the Chua Circuit using the oscilloscope. The measurement parameters for the data are as follows:

- Total Points of Data: 100000
- Points per second : 50
- Significant Figures : 3

Figure 11 shows a graph of *XY* against time. This was done to visually assess the distribution of data present. From the graph we can see that the X and Y components themselves do not look very random. The distribution heavily favours zero and is decidedly non-random, hence we can skip any kind of testing at this stage.

Figure 11. Graph of XY against Time



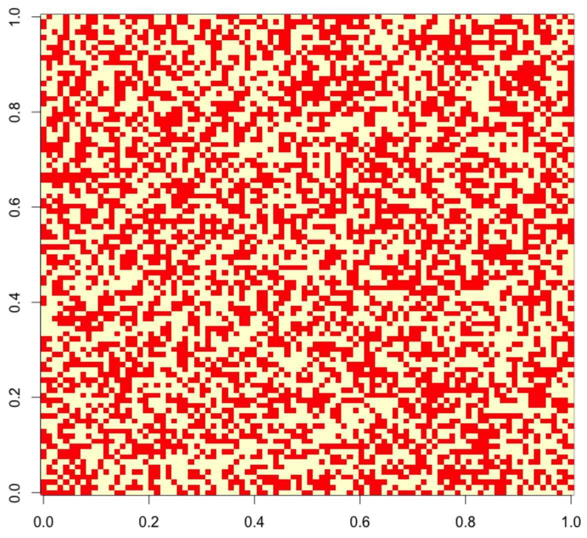
Analysing the attractor itself, we can see that the closer the signal gets, the denser the lines get. This means that there is a higher probability that it will be found closer to an attractor than in a position away from it. Moreover, the system is closed, and any approach that considers it open will simply fail to be random. From our experiments with the circuit before, we know that due to the sensitive dependence on initial conditions it cannot be determined in which attractor the point will be for any given time. In a suitably balanced graph - i.e. one with left and right basins of the same size - we can assume that the probability of finding the point in either will be 0.5. This is of course assuming that there are no points left in the system that we haven't assigned to either the left or the right attractors.

The demarcation of left or right is binary. If the stream is reduced down to a single bit per data point - which contains the information about which basin the point is currently at - we can make a preliminary assumption that the data so obtained will be sufficiently random. Since the two attractors are separated about the X axis, we can simply use a piecewise transformation on the X axis data to obtain our binary bit stream. Consider the incoming points from the X signal to be part of the function f(x). We define function r(x) such that

$$r(x) = \begin{cases} 1, & \text{if } f(x) > 0 \\ 0, & \text{if } f(x) \leqslant 0 \end{cases} \tag{7}$$

Figure 12 shows the output of r(x) transformed into a binary matrix of equal dimensions and displayed as an image. We can see that there are no discernible patterns. This is a legitimate and often used method of checking for randomness, as the human brain exhibits an uncanny skill in pattern recognition that we are yet to replicate in code. This will suffice as a starting point for our further tests on randomness.

Figure 12. Image of Random Data Generated



Tests for Randomness

It was interesting for me to find in my research that randomness in a data stream can never be proven or unproven. Even if the data looks decidedly not random, it is possible that you may have recorded the data stream at a point where a pattern was expressed as a mathematical inevitability. Therefore, at the end of this report we cannot say if the stream we have collected is definitively better or worse than another. We will however, through a number of tests, be able to express our confidence in these numbers being random.

For comparison, we have selected two bitstreams of the same size. The first stream is generated from a PRNG in the programming language R. R uses a mathematical function called the Mersenne-Twister. The Mersenne-Twister (Makotor & Takuji, 1998) is one of the most widely used PRNGs in the world. For the scope of this paper, we can assume that it will provide numbers of a certain quality, surpassing which we can consider our generator to be of PRNG quality. The second stream is from the TRNG used by http://random.org. The website uses a TRNG of reasonable reputation which makes use of atmospheric noise captured around the world using radio telescopes. We will use this as our stream of true random numbers as well as a benchmark against which the generator will be tested.

Testing Methodology

We will use three different tests to compare the streams of data which we will call TR, PR and R. R will refer to the bitstream we generated using the circuit, TR to the numbers from the TRNG, and PR to the numbers from the PRNG. We will not explore in detail how the tests work, but we will move forward with the assumption that a similar score in the tests refers to a similar degree of confidence in the randomness of the numbers. The three tests are as follows:

- Bartel's Test for Randomness - this is an implementation of the Ranked Version of the Von Neumann test for Randomness. This is a statistical method for testing randomness that is part of the state-of-the-art random test suite diehard (Bartels, 1982).
- Shannon Entropy Test - As the name suggests, the Shannon Entropy level is a measure of the randomness present in a bitstream w.r.t compressibility (Shannon, 2001). The lower this number, the more compressible the stream. Simply put, it is the expected value of the information present in a stream.
- Visual Test of Randomness - Similar to what was done before, we will visually compare the three sources for patterns and randomness. While this is not useful for comparing the randomness of highly random data, we will nevertheless be able to look for patterns and signs that the data is decidedly not random.

Bartel's test revealed - for the alternate hypothesis of non-randomness - the following scores:

- R : statistic = -0.1133, p-value = 0.9098
- PR: statistic = -0.2327, p-value = 0.816
- TR: statistic = 0.7518, p-value = 0.4522

The statistic measures the probability of a trend or oscillation in consecutive numbers. The difference in the statistic values arise from the nature of the underlying processes that generate the data being used. For the scope of this paper, we will simply consider the p-values to evaluate whether there is enough evidence to reject the null-hypothesis that the numbers are random. Bartel's test does not identify either data stream as nonrandom, since the p-values are considerably high. It can be noted that the bitstream performs better than the two generators here.

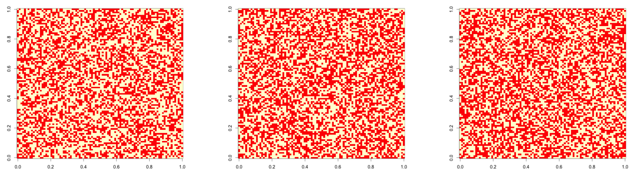
Next, considering Shannon Entropy we look for the entropy scores of the data streams:

- R : entropy = 8.466952
- PR: entropy = 8.397283
- TR: entropy = 8.384804

The entropy test classifies the three samples as having a high amount of entropy, with the bit stream R having the highest amount. The PRNG output scores higher than those from the TRNG, but this can be explained since pseudo-random number generators have been designed to imitate high entropy output. We are more interested in the comparison between the TRNG and the circuit-based generator, where the generator succeeds in creating a stream of higher entropy than the TRNG as well as the PRNG.

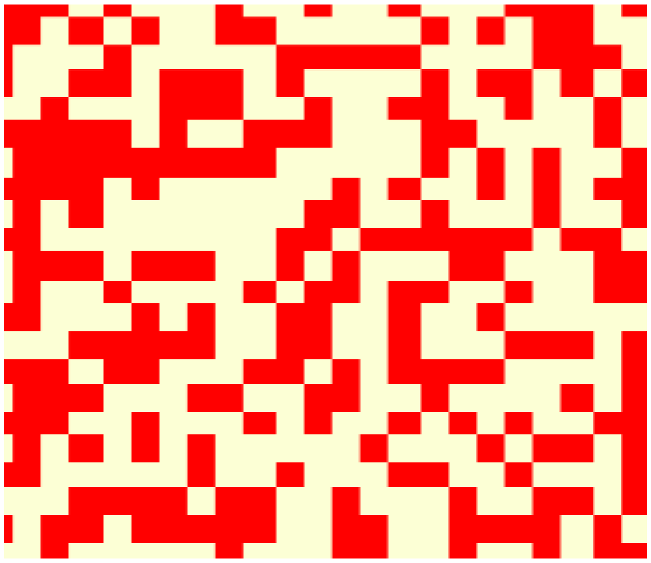
Finally, we plot images of the three data streams for visual comparison (figure 13).

Figure 13. (a) Data from Chua Circuit (b) PRNG Data (c) TRNG Data



The astute observer (perhaps the ability to zoom-and-enhance) will notice that there are some visual patterns present in (b), but (a) and (c) contain no discernible patterns. A close-up of these patterns can be seen in Figure 14, although it must be pointed out that these are not recurring patterns. They are simply features in the data that are easily identified, and their presence - while diminishing the randomness in the output - does not make it predictable enough to be considered nonrandom.

Figure 14. 4x zoom of PRNG output



Summary

The tests we have performed are by no means conclusive, and neither are the samples collected. Proper exploration of the randomness of this generator - perhaps for commercial use - needs to be done after extensive data collection for a period of months, and substantial mathematical testing. However, the three tests we have used are state-of-the-art in testing for randomness, and they are in agreement that randomness is present in the data, generated to a level of confidence at par with or beyond that of the True Random Number Generator.

Passing further tests, the circuit and the accompanying generator have huge potential in security related applications where a source of high-quality random numbers are required, with a considerable number of advantages over existing systems. The entire circuit can be etched in Silicon, provides shorter start-up times, is relatively immune to noise and malicious interference, and - most importantly - does not rely on the obscurity of its model for unpredictability. As we have seen in many cases (Knuth, 1973), this kind of security through obscurity is eventually bound to fail.

As for further research, I was curious to consider the Lyapunov exponent (Dieci, Russel, & Vleck, 1997) of the system to measure the highest possible bitrate that can be achieved, and how an increase in sampling frequency affects the randomness of the bits so obtained. It would be interesting to explore the nature of the Lyapunov exponent, and how this relates to the physical implementation of the circuit.

References

Bartels, R. (1982). The rank version of von neumann's ratio test for randomness. , *77*(377), 40–46. Retrieved 2015-11-24, from `http://www.jstor.org/stable/2287767` doi: 10.2307/2287767

Dieci, L., Russel, R. D., & Vleck, E. S. V. (1997). On the computation of lyapunov exponents for continuous dynamical systems. , *34*(1), 402–423.

Knuth, D. E. (1973). Super-random number generator. In *The art of computer programming* (pp. 100–150).

Makotor, M., & Takuji, N. (1998). Mersenne twister: A 623-dimensionally equidistributed uniform pseudorandom number generator. , *8*(1), 3–30.

Matsumoto, T. (1984). A chaotic attractor from chua's circuit. , *31*(12), 1055–1058.

Shannon, C. E. (2001). A mathematical theory of communication. , *5*(1), 3–55. Retrieved 2015-11-24, from `http://doi.acm.org/10.1145/584091.584093` doi: 10.1145/584091.584093

Siderskiy, V. (2000). *The antoniou inductance-simulation circuit derivation.* Chua Circuits (website). Retrieved 2015-11-24, from `http://www.chuacircuits.com/PDFs/Antoniou Inductance-Simulation Circuit.pdf`

W.K, C. (2003). Negative impedance conversion. In *The circuits and filters handbook* (pp. 396–397). CRC Press.