REALIZATION OF A DIGITAL CHAOTIC OSCILLATOR BY USING A LOW COST MICROCONTROLLER

Ercan Köse[1*] – Aydın Mühürcü[2]

[1]Department of Mechatronics Engineering, Tarsus Technology Faculty, Mersin University, 33480-Tarsus, Mersin-Turkey
[2]Department of Electrical and Electronics Engineering, Engineering Faculty, Sakarya University, 54187 Sakarya-Turkey

ARTICLE INFO

***Article history*:**
Received: 06.06.2016.
Received in revised form: 16.08.2016.
Accepted: 30.08.2016.

Keywords:
Microcontroller
FPGA
Chaotic oscillator
Discrete-time
s and z-transform
Algorithm

Abstract:

This study addresses the in-detail steps to create a chaotic oscillator with continuous-time equations using microcontroller hardware with a little lower clock-frequency and narrower data bus, utilized at much lower hardware, software and algorithm development costs compared to chaotic oscillators developed using analog circuit components or a hardware-based software platform such as FPGA. For this purpose, a Lorenz chaotic oscillator with continuous-time nonlinear equations was selected. Lorenz t-domain equations were transformed into S-domain and Z-domain, respectively. After these transformations, a detailed flowchart was given to illustrate the steps required to implement the chaotic oscillator in the microcontroller. All the details derived were simulated by running simultaneous MATLAB-SIMULINK simulations. Besides, the performance of the discrete-time chaotic oscillator was executed in the PIC18F452 microcontroller, produced by the Microchip Technology Inc. and visualized in 1D and 2D graphs on an oscilloscope screen.

1 Introduction

Many events in nature cannot be explained by the theory of linear systems. Therefore, non-linear dynamical phenomena like chaos theory which describes the behavior of certain dynamical systems are extremely sensitive to the initial conditions [1]. Chaos theory is also now witnessing a lot of increased enthusiasm in interdisciplinary sciences [2], varying from weather to finance, from economics to hydraulics, from nanotechnology to medical and from turbulence to vibration [3]. Because chaos increases the working area, it has to be concentrated on the studies required for the production of chaotic signals.

There exist several ways how to practically realize an analog chaotic signal oscillator since the nonlinear behavior of a chaotic oscillator has shown too many difficulties for the parameters involved in the mathematical model of a chaotic oscillator [4]. Therefore, numerous chaotic oscillators have been proposed and realized with different kinds of electronic devices, trying to generate a little higher number of scrolls [5]. But, these oscillators have got non-autonomous circuits or the simplest autonomous circuits because of an intolerance value

[*] Corresponding author. Tel.:+90.324.6274804; fax: +90.3246274805
E-mail address: ekose@mersin.edu.tr

used for physical components such as resistance–inductance–capacitance- transistor- diode etc. [6]. To perform autonomous chaotic generators, the digital signal generator should be used instead of analog signals. Digital based chaotic generators have several/some advantages over analog based chaotic generators in terms of autonomy, minimization, synchronization, flexibility, usefulness, practicability and stability. Meanwhile, a digital based chaotic generator avoids the parameters mismatch among analog based chaotic generators.

Over the last decade, numerous digital chaotic generators such as Application Specific Integrated Circuits (ASIC), Digital Signal Processors (DSPs), and Field-Programmable Gate Arrays (FPGA) [7] have been proposed. FPGA realization was performed for chaotic frequency hopping sequences [8], Lorenz's chaotic generator for ciphering telecommunications [9], a new auto-switched chaotic system [10], a real time novel chaotic oscillator [7], and multi-scroll chaotic oscillators have been designed [4]. In FPGA applications, using distinct algorithms like, the Euler, Heun, Runga Kutta are used. Nowadays, a great number of applications of the chaotic systems have been suggested and carried out in the FPGA technology. Unfortunately, hardware characterized by trustworthy statistical properties cannot be easily applied [8]. The cost of the FPGA is higher than the microcontroller and the design of the algorithms that provided the realization of the chaotic oscillator in the FPGA proved to be a very difficult and complex task. Besides this, in FPGA, the speed of the mathematical operations used in the algorithms may be lower [11]. To get rid of these problems mentioned above, the implementation of digital chaotic oscillators is based on using a microcontroller.

On the other hand, the purpose of this study is to use a cost effective microcontroller instead of using many circuit elements for generating the chaos signal. Microcontroller codes that are compatible with mathematical models have been developed and installed to produce chaos signals. Microcontrollers can provide the chaos generator valuable features to be used easily in many areas. Additionally, the software embedded in the current mobile phone will allow us to encrypt communications easily.

2 Discretization process for continues time chaotic oscillators

In this study, continuous time Lorenz chaotic oscillator [12] has been chosen to be transformed and implemented as a discrete time chaotic oscillator applied with microprocessor hardware. Here, the chaotic oscillator of Lorenz has non-linear equations where x, y and z are state variables; a, b and c are positive constant parameters, Eq. 1.

$$\left.\begin{aligned} \frac{\partial x(t)}{\partial t} &= ay(t) - ax(t) \\ \frac{\partial y(t)}{\partial t} &= cx(t) - x(t)z(t) - y(t) \\ \frac{\partial z(t)}{\partial t} &= x(t)y(t) - bz(t) \end{aligned}\right\} \tag{1}$$

Using the Laplace transformation method [13], t-domain Lorenz equations are transformed to S-domain Lorenz equations as seen in Eq. 2.

$$\left.\begin{aligned} X(s) &= a\frac{1}{s}\left[Y(s) - X(s)\right] \\ Y(s) &= \frac{1}{s}\left[cX(s) - X(s)Z(s) - Y(s)\right] \\ Z(s) &= \frac{1}{s}\left[X(s)Y(s) - bZ(s)\right] \end{aligned}\right\} \tag{2}$$

After S-domain transformation, the Lorenz equations have been converted to a block diagram and visualized on Matlab-Simulink, Fig 1. Using a Matlab-Simulink S-domain model, the time series of the Lorenz chaotic oscillator of the x, y, z (1-D phases) and y-x, z-x, y-z (2-D phases) were plotted as shown in Fig. 2 and Fig. 3.

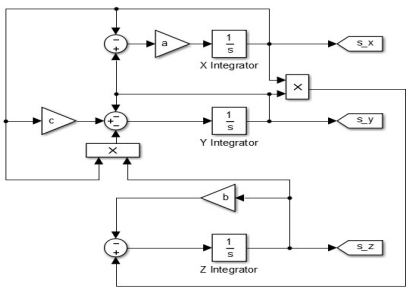

Figure 1. S-domain block diagram for Lorenz chaotic oscillator visualization on MATLAB-SIMULINK

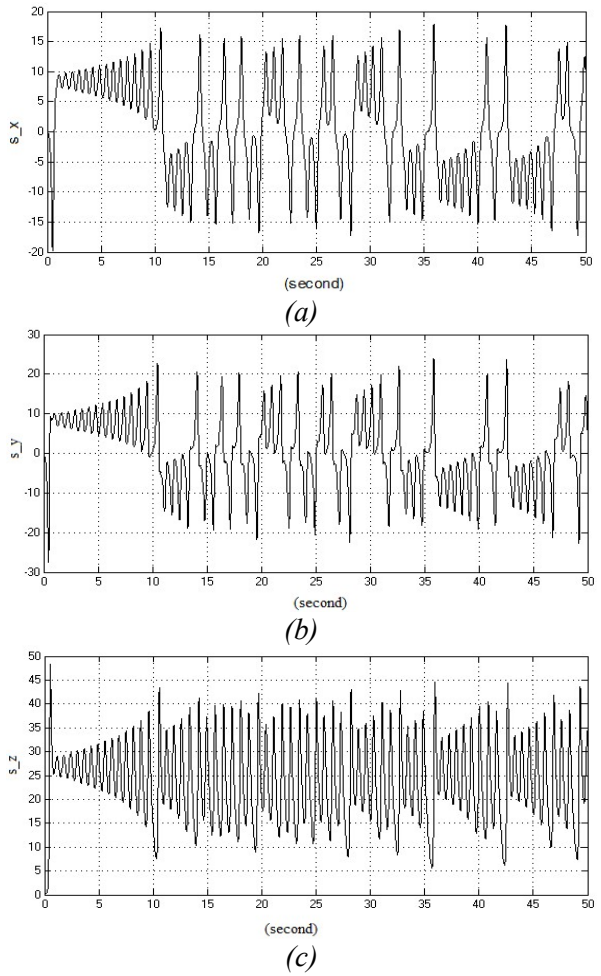

Figure 2. (a) x, (b) y and (c) z variables in the S-domain Lorenz chaotic system.

The Lorenz S-domain block diagram has been converted into Lorenz Z-domain (discrete time) block diagram shown in Fig. 4 using forward difference method [14], presented in Eq. 3.

$$s \approx \frac{1}{T}(z-1) \tag{3}$$

here T (named with tsample on Simulink), is the sample period, [14].

In this study, in order to select the T, X, Y and Z chaotic oscillation signals were observed the S-domain in order to analyze the change speed. As indicated in Fig. 2, T = 0.005s is calculated by multiplying two peak time, where the change is most intense with 0.01.

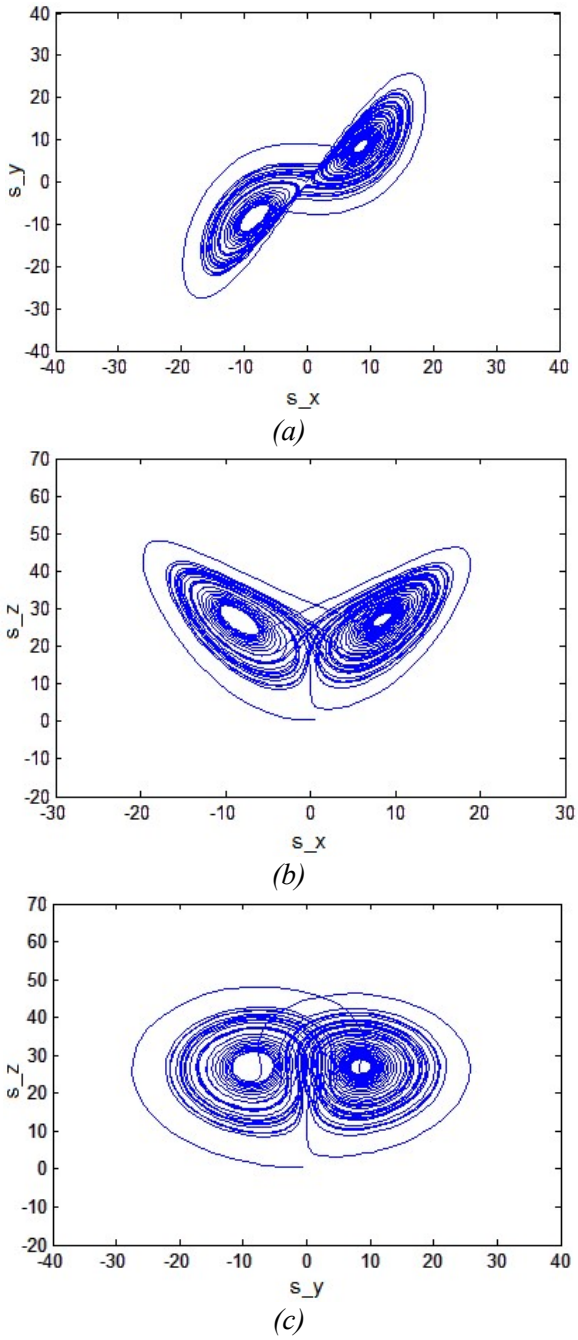

Figure 3. Phase portrait of the S-domain Lorenz Systems in the (a) xy, (b) xz, and (c) yz.

Using the direct programming method [15], the Z-domain integral transfer functions have been converted to Z-domain integral block diagram based on 1 sample delay, 1/z, blocks, Fig. 5.

T sampling parameters is used for the production of the discrete time chaotic signal. Selection sensitivity of the T parameter is an indicator which varies depending on the application areas sign of the chaotic. For example, if the chaotic signal is used to

change the momentum of a mechanical stirrer, T selection will have no need for precision [16]. However, if the chaotic signal is used for encryption, which will be produced in the discrete-time chaos sign encoder and decoder, the section should have absolutely the same sampling period T [17].

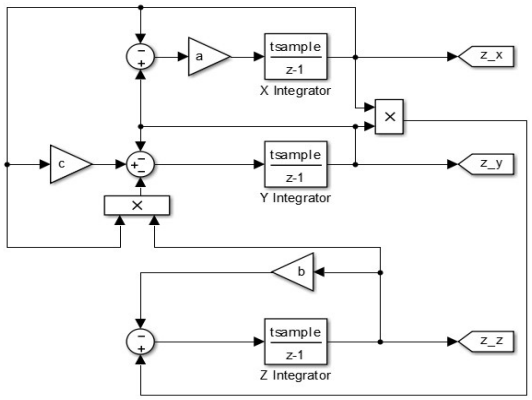

Figure 4. Z-domain block diagram for Lorenz Chaotic Oscillator visualization on MATLAB-SIMULINK.

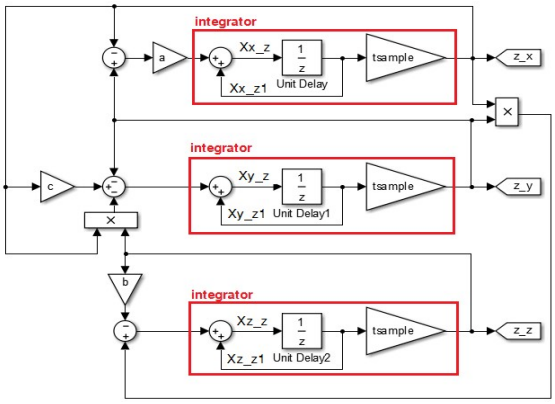

Figure 5. Lorenz block diagram implemented with discrete time integral block diagrams

The smallest difference will cause the deterioration of the chaos synchronization between the marks. The production of the Lorenz Signals using three different T is shown in the Fig. 6, $x = 1$, $y = -0.5$, $z = 1$ under the initial condition.

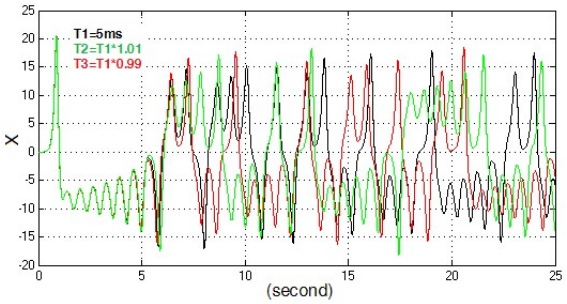

Figure 6: Effects on Lorenz X is a change of about 1% in the period T.

3 Running Lorenz in a coding system

The discrete time algorithm of Lorenz chaotic oscillator is adapted easily to the microprocessor by using discrete time Lorenz block diagram as shown in Fig. 5. Coding was performed considering the flow-chart diagram shown in Fig. 7.

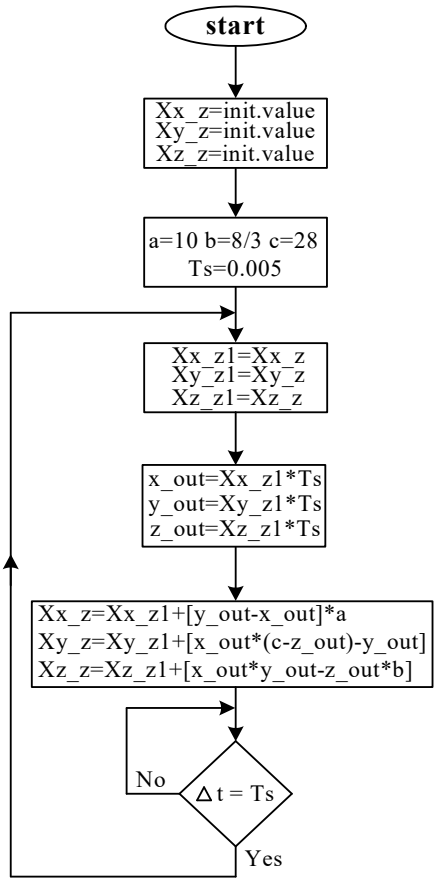

Figure 7. Flow-chart of discrete time Lorenz algorithm structure.

The flow diagram in Fig. 7 was performed using the discrete time Lorenz chaotic oscillator's block diagram represented in Fig. 5. At the beginning of the non-periodic part of the algorithm, the initial values and discrete time Lorenz chaotic oscillator parameters were assigned. A periodic procedure was repeated after each new T_S timeout. In our simulation and experimental studies $T_S = 4ms$ were selected and the initial conditions were chosen as $x = 1$, $y = -0.5$, $z = 1$.

4 Hardware experimentation

Usually, the simulation and hardware commands are used simultaneously in FPGA-based discrete-time chaotic oscillator algorithms. Hybrid code sequences are generated by this software approach [4, 7, 8, 9]. Although such software algorithms are called hardware-based, these are indeed a kind of simulation operation. The main reason for not being able to switch to a hardware-only platform is the inability to transfer simulation commands of the hybrid codes into the hardware [18]. And, missing codes in the hardware will result in either broken or erroneous hardware. The most concrete evidence, revealing whether these operations apply a hybrid command system, can be obtained by reflecting the results of the algorithm executed in a FPGA on an oscilloscope screen through a DAC (Digital-to-Analog Converter) integrated-circuit [19]. FPGA structure consists of logic gates, and this structure has no DAC circuit. Unless a DAC is used, the results of the algorithmic operation executed in FPGA cannot be transferred to the external environment in an analog signal form.

Chaotic signals can have bi-polar analog signals (Fig. 2). As shown in this study, the output variables of the algorithm may need to be fed into mono-polar DAC as an input signal through the processor. In this case, the chaotic signal must be offset before leaving the processor (Fig. 8). Thus, the output variables of the algorithm are not allowed to be negative in order to prevent losing negative analog signals (Fig. 9).

In this study, the discrete-time Lorenz chaotic oscillator algorithm was executed in cost effective microcontroller hardware. Running the algorithm in an 8-bit processor has some advantages compared to FPGA-based algorithms [4, 7, 8, 9]. The algorithm codes were transferred into a PIC18F452 microcontroller produced by the Microchip Technology Inc. This microcontroller has an 8-bit ALU (Arithmetic Logic Unit). Thanks to this ALU, the mathematical operations of the algorithm are processed by microcontroller in a fast and simple manner. In addition, the software development platform of the microcontroller, such as CCS-C, Micro C, etc., does not have any additional hardware or software cost for floating-point operations with real variables, contrary to Xilinx ISE, Altera Quartus, etc. in the FPGA platform. Hence, any function that requires floating-point operations can be coded and compiled in microcontrollers without any special effort as in FPGA [20, 21].

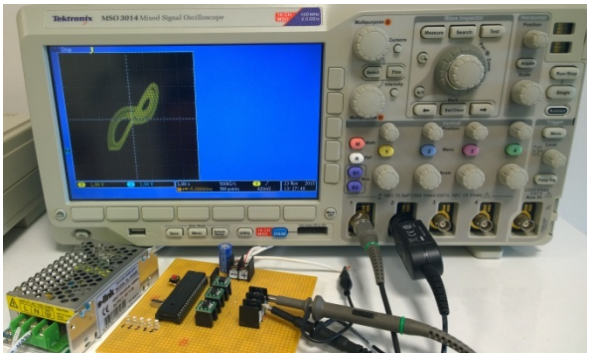

Figure 8. Lorenz chaotic oscillator appearance in the realization of digital platforms

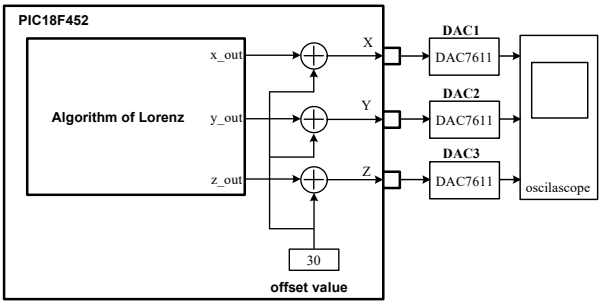

Figure 9. Microcontroller based chaotic oscillator hardware unit block diagram.

Figure 10. shows the oscilloscope images for the x, y, and z outputs of the hardware structure shown in Fig. 8. The 1D output waveform was visualized in 2D in y-x, z-x, and z-y forms by activating the XY-mode of the oscilloscope (Fig. 11).

The discrete-time Lorenz chaotic oscillator algorithm was executed in 8-bit PIC18F452 microcontroller with a crystal frequency of 10 MHz. Microcontroller and peripherals were supplied with

a 5V voltage source. The electrical power consumed by hardware was measured to be 225mW. The microcontroller program of the discrete-time Lorenz chaotic oscillator algorithm was coded using CCS-C compiler. In the software, 11 floating-point real variables (4 bytes each), 3 2-byte integer variables (int16) and 2 1-byte integer variables (int8) were defined. Using these variables, 3 float+float, 4 float-float, 8 float*float mathematical operations; 3 float-to-float and 3 float-to-int16 transfer operations were performed. The cost of discrete-time Lorenz chaotic oscillator algorithm was measured as 350µs per 1 run in a 10MIPS PIC18F452 microcontroller. And, the memory cost of the algorithm in the processor was found to be 6% RAM and 9% ROM (Fig. 12).

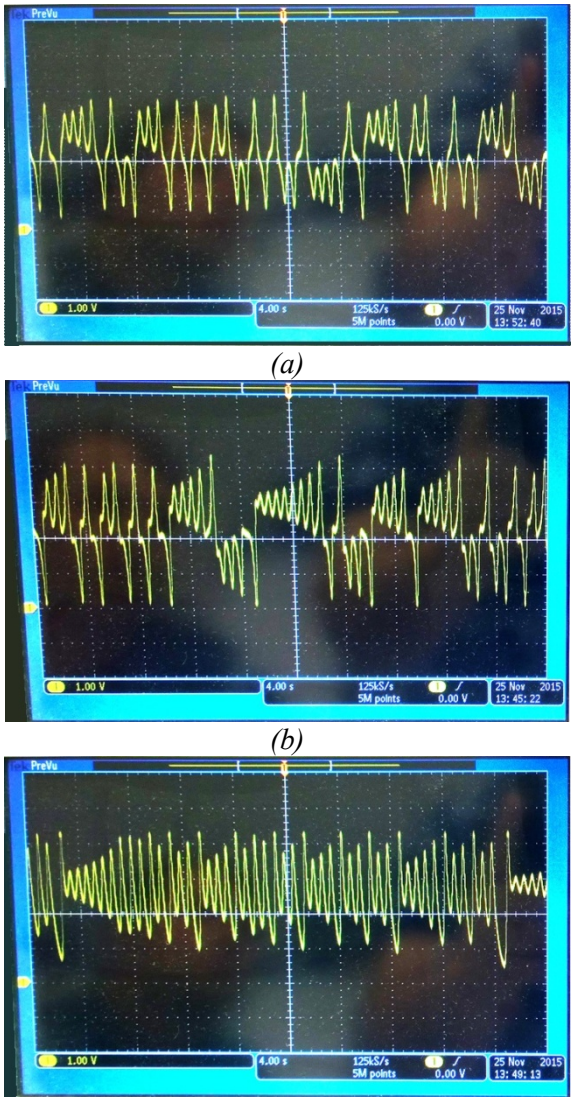

Figure 10. (a) x, (b) y and (c) z variables oscilloscope displays for Lorenz chaotic system.

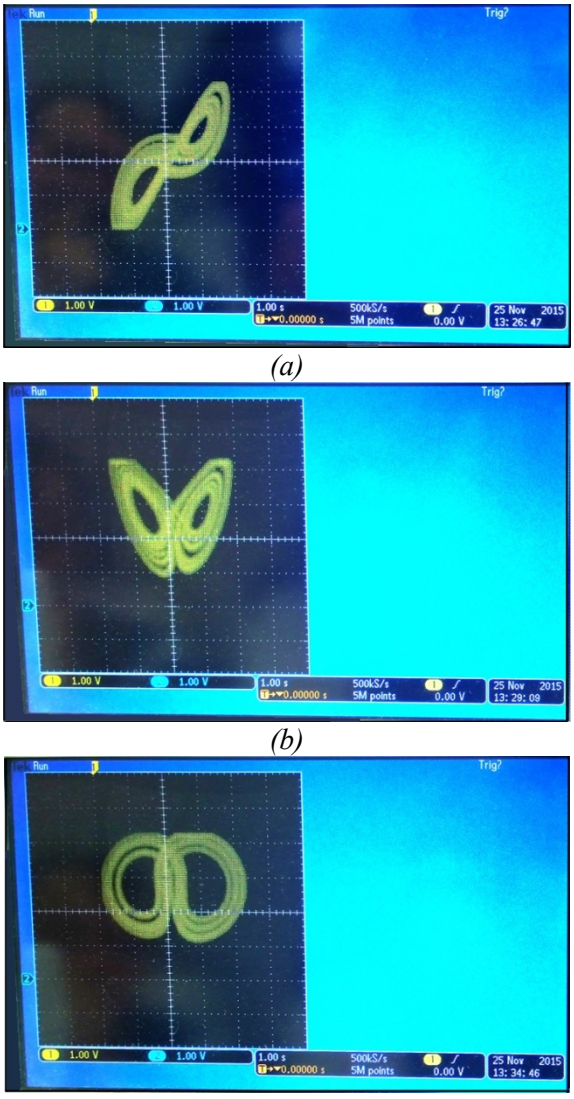

Figure 11. Phase portrait oscilloscope displays for the Lorenz oscillator in the (a) xy, (b) xz, and (c) yz.

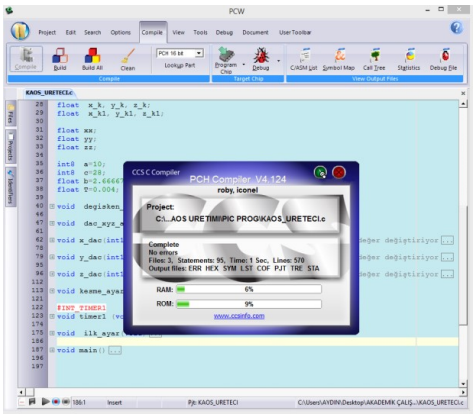

Figure 12. Used microcontroller's memory percentages.

5 Conclusion

Considering the 9% memory cost of the discrete-time Lorenz chaotic oscillator algorithm in this study, it is evident that numerous chaotic algorithms can be placed in a single microcontroller chip, without using complex processors. In the generation of chaotic signals, digital microcontroller platforms were proven to be reliable as seen in the 1D and 2D signal graphs, produced by the discrete-time chaotic oscillator in the microcontroller-based hardware platform developed by Laplace transform, Z-transform, and the sampling theorem. It was shown that the initial conditions and parameter values of the chaotic oscillators in the digital platform turn into a software-controlled scalar structure, which is not a random variable as it is in the analog-circuit chaotic oscillators. Consequently, it was understood that the microcontroller-based discrete-time chaotic generators can provide important contributions to minimization, autonomous, synchronization, flexibility, usefulness, practicability and stability studies, as well as establishing a significant ground for further studies. It was also observed that the software, hardware, and algorithm development costs of the microcontroller-based processors are much lower compared to FPGA-based complex processors.

References

[1] Chen, G., Wang, X., Li, X.: *Fundamentals of Complex Networks: Models, Structures and Dynamics*, Higher Education Press, Wiley. 73-74, 2015.

[2] Gandhi, G.: *Electronic Realizations of Chaotic Circuits: From Breadboard to Nanotechnology*, Theses of the Ph.D. Dissertation, P´azm´any P´eter Catholic University, Budapest, 4-5, 2008.

[3] Wei, H., Xu, J., Zhu, D., Wu, Y., Lu, J., Lu, K.: *Multi-objective optimization of parameters and location of passive vibration isolation system excited by clamped thin plate foundation*, Engineering Review, 36 (2016), 1, 19-27.

[4] Tlelo-Cuautle, E., Rangel-Magdaleno, J.J., Pano-Azucena, A.D., Obeso-Rodelo, P.J., Nunez-Perez, J.C.: *FPGA realization of multi-scroll chaotic oscillators*, Commun Nonlinear Sci Numer Simulat, 27 (2015), 1-3, 66-80.

[5] Munoz-Pacheco, J., Tlelo-Cuautle, E., Toxqui-Toxqui, I., Sanchez-Lopez, C., Trejo-Guerra, R.: *Frequency limitations in generating multi-scroll chaotic attractors using CFOAs*, Int J Electron. 101 (2014), 11, 1559–69.

[6] Günay, E.: *A new autonomous chaos generator from state controlled-cellular neural networks*, Int. J. Bifurcation Chaos, 22 (2012), 3, 1250069–1–10.

[7] Koyuncu, I., Ozcerit, A.T., Pehlivan, I.: *Implementation of FPGA-based real time novel chaotic oscillator*, Nonlinear Dyn. 77 (2014), 1–2, 49–59.

[8] Ling, C., Wu, X.: *Design and realization of an FPGA-based generator for chaotic frequency hopping sequences*, IEEE Trans Circ Syst I. 48 (2001), 521–32.

[9] Azzaz, M.S., Tanougast, C., Sadoudi, S., Dandache A.: *Real-time FPGA implementation of Lorenz's chaotic generator for ciphering telecommunications*, In: IEEE North–East workshop on circuits and systems and TAISA conference, 2009, 1–4.

[10] Azzaz, M.S., Tanougast, C., Sadoudi, S., Fellah, R., Dandache, A.: *A new auto-switched chaotic system and its FPGA implementation*, Commun Nonlinear Sci Numer Simul, 18 (2013), 7, 1792–804.

[11] Parnell, K., Bryner, R.: *Comparing and Contrasting FPGA and Microprocessor System Design and Development*, WP213. 1 (2004), 1, 1-32.

[12] Lorenz, E.N.: *Deterministic Nonperiodic Flow*, Journal of the Atmospheric Sciences, 20, 130-141, 1963.

[13] Tomas, B.Co.: *Methods of Applied Mathematics for Engineers and Scientists*, Cambridge University Press, 12.4, 2013.

[14] Fadali, M.S., Visioli, A.: *Digital Control Engineering,* Academic Press, 9-50, 2009.

[15] Stojković, N.V., Stanimirović, P.S.: *Two direct methods in linear programming*, European Journal of Operational Research, 131 (2001), 2, 417-439.

[16] Ye, S., Chau, K.T., Niu, S.: *Chaoization of a Single-Phase Induction Motor for Washing Machines*, IEEE Industry Applications Conference Forty-First IAS Annual Meeting, (2006) 2, 855-860.

[17] Kocarev, L., Amigo, J.M., Szczepanski, J.: *Chaos-based Cryptography: an overview*, International Symposium on Nonlinear Theory and its Applications (NOLTA2005) (2005), 453-456.
[18] Ziade, H., Ayoubi, R., Velazco, R.: *A Survey on Fault Injection Techniques*, The International Arab Journal of Information Technology, 1(2014), 2, 71-186.
[19] http://www.microsemi.com/document-portal /doc_view/133662-in-circuit-fpga-debug-challenges-and-solutions
[20] http://www.ti.com/lit/wp/spry113/spry113.pdf
[21] Sarra, S.A., Meador, C.: On the numerical solution of chaotic dynamical systems using extend precision floating point arithmetic and very high order numerical methods, Nonlinear Analysis: Modelling and Control, 16 (2011), 3, 340–352.