Hardware Implementation of FPIC Controllers for Discrete-Time Chaotic Systems Using LabVIEW-FPGA

Implementación en Hardware de Controladores FPIC para Sistemas Caóticos en Tiempo Discreto Usando LabVIEW-FPGA

Heiner Castro Gutiérrez[1], Carlos Robles-Algarín[2*], John Taborda Giraldo[3]

1. PhD., Purdue University, West Lafayette, United States. https://orcid.org/0000-0002-9218-7540
2*. PhD., Universidad del Magdalena, Santa Marta, Colombia. https://orcid.org/0000-0002-5879-5243, croblesa@unimagdalena.edu.co
3. PhD., Universidad del Magdalena, Santa Marta, Colombia. https://orcid.org/0000-0002-6090-1711

Cite this article as: H. Castro Gutiérrez, C. Robles-Algarín, J. Taborda Giraldo
"Hardware Implementation of FPIC Controllers for
Discrete-Time Chaotic Systems Using LabVIEW-FPGA",
Prospectiva, Vol. 23 N° 2 2025

Recibido: 03/05/2025 / Aceptado: 11/07/2025

http://doi.org/ 10.15665/rp.v23i2.3767

ABSTRACT

This paper presents the hardware implementation of control algorithms based on the Fixed-Point Inducting Control (FPIC) technique applied to chaotic systems. Three discrete-time chaotic maps—Logistic, Fold, and Flip—were analyzed in both coupled and uncoupled configurations. For each system, equilibrium points were analytically determined. The control strategy was implemented using LabVIEW-FPGA to generate and stabilize chaotic behaviors in real time, and the results were validated against MATLAB simulations. Bifurcation diagrams were generated to identify parameter regions that lead to chaotic or stable behaviors. All systems were implemented using 16-bit fixed-point arithmetic, demonstrating the feasibility of FPGA-based realization of FPIC-controlled chaotic systems. The proposed prototyping setup provides a valuable platform for rapid testing of chaos-based control strategies and their potential applications in secure communications and nonlinear systems. The results demonstrate that FPIC effectively stabilized chaotic behavior in all systems, with convergence to the analytical fixed point occurring near gs ≈ 0.5 for the Logistic Map and gs ≈ 0.748 for the Fold Map. Experimental FPGA results closely matched MATLAB simulations, confirming the accuracy and viability of the proposed implementation.

Key Words: chaotic systems, FPIC control, LabVIEW, FPGA, bifurcation diagrams, discrete nonlinear maps.

RESUMEN

Este artículo de investigación científica presenta la implementación en hardware de algoritmos de control basados en la técnica de Control por Inducción en Punto Fijo (FPIC) aplicada a sistemas caóticos. Se analizaron tres mapas caóticos en tiempo discreto—Logístico, Fold y Flip—tanto en configuraciones acopladas como no acopladas. Para cada sistema, se determinaron analíticamente los puntos de equilibrio. La estrategia de control fue implementada en tiempo real utilizando LabVIEW-FPGA con aritmética en punto fijo de 16 bits, y los resultados fueron validados mediante simulaciones en MATLAB. Se generaron diagramas de bifurcación para identificar las regiones de parámetros que conducen a comportamientos caóticos o estables. El entorno de prototipado desarrollado constituye una herramienta valiosa para la evaluación rápida de estrategias de control basadas en el caos y sus aplicaciones potenciales en comunicaciones seguras y sistemas no lineales. Los resultados demuestran que la técnica FPIC logró estabilizar efectivamente el comportamiento caótico en todos los sistemas, con convergencia al punto fijo analítico alrededor de gs ≈ 0.5 para el Mapa Logístico y gs ≈ 0.748 para el Mapa Fold. Los resultados experimentales en FPGA coincidieron estrechamente con las simulaciones en MATLAB, confirmando la precisión y viabilidad de la implementación propuesta.

Palabras clave: sistemas caóticos, control FPIC, LabVIEW, FPGA, diagramas de bifurcación, mapas discretos no lineales.

1. INTRODUCTION

Chaotic systems are nonlinear dynamical systems that exhibit complex, aperiodic behavior and extreme sensitivity to initial conditions. Despite being deterministic, their unpredictable evolution makes them both a challenge and an opportunity in control engineering [1], [2]. In recent years, chaos theory has garnered increasing attention in areas such as secure communications, encryption, modeling biological systems, random number generation, and the control of power converters [3], [4]. However, controlling chaotic dynamics remains a significant challenge due to the inherent instability and sensitivity of such systems.

Several control techniques have been proposed to stabilize chaotic systems around desired trajectories or equilibrium points [5], [6]. Traditional methods include feedback linearization, sliding mode control, and time-delayed feedback, among others [7], [8]. More recently, researchers have explored strategies that exploit the discrete-time structure of many chaotic maps, leading to simpler and faster control implementations suitable for digital hardware. Among them, the Fixed-Point Inducting Control (FPIC) technique has emerged as a promising approach due to its simplicity, robustness, and compatibility with fixed-point arithmetic [9], [10].

FPIC represents an advanced approach to managing chaotic systems, utilizing techniques from control theory to stabilize and manipulate unpredictable behaviors. The FPIC methodology focuses on integrating feedback mechanisms that adaptively govern chaotic dynamics, particularly in systems susceptible to instability and sensitivity to initial conditions. The principle underlying FPIC is its capability to induce stability by ensuring that the average of the system's behavior over time approaches a desired fixed point. This concept is closely related to Zero Average Dynamics (ZAD), another control strategy that aids in maintaining system stability. The combination of these two strategies creates a robust control framework wherein the FPIC technique can stabilize chaotic orbits prevalent in such systems [11], [12]. This property is especially beneficial when applied to power electronics, where chaotic behavior can significantly impact performance and reliability [9]. Particularly, the application of FPIC has been verified in controlling chaotic dynamics within power converters, improving system performance and stability under challenging conditions [13], [14].

At the same time, Field-Programmable Gate Arrays (FPGAs) have become increasingly popular for implementing real-time control systems, thanks to their parallelism, low latency, and energy efficiency. Compared to microcontrollers or DSPs, FPGAs offer better performance for systems requiring deterministic response and fast execution cycles—features critical when implementing real-time controllers for chaotic systems [15], [16]. Furthermore, high-level platforms such as LabVIEW-FPGA allow rapid prototyping without requiring low-level HDL programming, significantly reducing development time and complexity [17], [18].

Despite these advances, few works have explored the direct implementation of chaos control strategies—especially FPIC—on FPGA hardware using fixed-point arithmetic. Most studies remain limited to software simulations or focus on floating-point architectures that are not suitable for low-power or embedded applications. This represents a significant research gap, particularly in the context of real-time systems, where hardware resource efficiency is crucial.
Real-time chaos control is particularly relevant in embedded systems and hardware-constrained environments, where unpredictable behavior can compromise stability or security. The use of discrete-time chaotic maps, such as the Logistic, Fold, and Flip Maps, provides a simplified yet representative modeling framework for studying complex nonlinear dynamics. These maps are widely used due to their analytical tractability and ease of implementation in digital hardware, making them ideal candidates for validating control strategies under limited computational resources [19].

Moreover, the combination of FPIC with FPGA implementation offers a compelling solution for applications requiring low latency and high reliability. The simplicity of FPIC, which avoids the need for system identification or heavy computation, aligns well with the hardware constraints of real-time systems. By leveraging fixed-point arithmetic and modular design in LabVIEW-FPGA, the approach presented here enables rapid iteration, scalability to multidimensional chaotic networks, and direct integration into low-power devices [20], [21]. This flexibility supports a wide range of use cases, from chaos-based encryption to real-time control in nonlinear systems [22], [23].

In this work, we address this gap by presenting the design, implementation, and experimental evaluation of FPIC controllers for three one-dimensional discrete chaotic maps: the Logistic Map, the Fold Map, and the Flip Map. Both standalone and linearly coupled versions of these maps are considered. The control strategy is implemented using LabVIEW-FPGA, enabling real-time generation and stabilization of chaotic behavior with 16-bit fixed-point arithmetic. Bifurcation diagrams are obtained to analyze system dynamics under varying control parameters, and the FPGA results are validated against MATLAB simulations. The proposed system offers a flexible and reusable platform for the rapid prototyping of chaos-based control strategies, with potential applications in embedded systems, nonlinear control, and secure communications.

2. METHODOLOGY

The methodology adopted in this work involves design, hardware implementation, and experimental validation of FPIC controllers applied to chaotic systems using the LabVIEW-FPGA platform. The process is structured into five key stages: mathematical modeling of chaotic systems, formulation of the FPIC controller, hardware implementation using an FPGA, simulation validation, and experimental testing.

2.1. Chaotic Systems in Nonlinear Dynamics

A chaotic system is a nonlinear dynamic system that exhibits disordered behavior despite being governed by deterministic laws. One of its defining characteristics is extreme sensitivity to initial conditions, where even minimal variations in the starting point can lead to vastly different trajectories over time [24]. This

property, often referred to as the butterfly effect, is a hallmark of chaos and has been extensively studied in systems across physics, biology, and engineering.

In this study, three one-dimensional discrete-time chaotic maps were selected for analysis and implementation: The Logistic Map, the Fold Map, and the Flip Map [25]. These systems were chosen due to their well-known nonlinear dynamics and frequent use in studies of chaos theory. Their mathematical formulations are given by equations (1), (2), and (3), respectively:

$$x(k+1) = rx(k)\big(1 - x(k)\big) \tag{1}$$

$$x(k+1) = r + x(k) + x^2(k) \tag{2}$$

$$x(k+1) = -(1+r)x(k) + x^3(k) \tag{3}$$

To explore higher-dimensional dynamics and more complex behaviors, coupled versions of these maps were also implemented. A coupled chaotic system can be constructed by linking two identical chaotic maps, either linearly or nonlinearly [21]. In this work, only linear coupling was considered, with formulations as shown in equations (4) and (5):

$$\begin{pmatrix} x(k+1) \\ y(k+1) \end{pmatrix} = \begin{pmatrix} f_x(k) + \epsilon\big(y(k) - x(k)\big) \\ f_y(k) + \epsilon\big(x(k) - y(k)\big) \end{pmatrix} \tag{4}$$

$$\begin{pmatrix} x(k+1) \\ y(k+1) \end{pmatrix} = \begin{pmatrix} f_x(k) + \epsilon\left(f_y(k) - f_x(k)\right) \\ f_y(k) + \epsilon\left(f_x(k) - f_y(k)\right) \end{pmatrix} \tag{5}$$

Here, f_x and f_y represent the chaotic functions corresponding to the right-hand sides of equations (1)–(3). The linear coupling approach allows the creation of symmetrical systems where each map influences the behavior of the other through a shared coupling term. These configurations offer a stepping stone toward more complex systems that involve multiple chaotic maps with heterogeneous dynamics or nonlinear interactions. While this work focuses on linearly coupled maps, future efforts may extend this approach to nonlinear coupling schemes or networks of multiple chaotic generators interacting through various topologies. Such systems are particularly relevant in modeling complex phenomena in nature and in engineering applications, such as secure communications, where the synchronization of chaotic signals plays a crucial role.

2.2. Fixed-Point Inducting Control Theory

Fixed-Point Inducting Control (FPIC) is a control technique designed to stabilize the behavior of chaotic systems by driving their trajectories toward a fixed point. Since chaotic systems are inherently unstable and extremely sensitive to initial conditions, a control mechanism is necessary to enforce predictable and stable dynamics. FPIC was first introduced approximately a decade ago and has since been applied in various domains, including power electronics and nonlinear mechanical systems [10].

The FPIC strategy is grounded in a theoretical result that allows the stabilization of a fixed point in discrete-time chaotic systems [14]. Consider a system described by a difference equation of the form:

$$x(k+1) = f\big(x(k)\big) \tag{6}$$

Where $f(x(k))$ is a nonlinear, potentially chaotic function. A point x^* is said to be a fixed point if $f(x^*)=x^*$. To determine whether this fixed point can be stabilized, the system is linearized around x^*, and the Jacobian matrix J_0 is evaluated at that point. If the eigenvalues of J_0 satisfy the following condition:

$$|\lambda_i(J_0)| = \left|\lambda_i\left(\frac{\partial y}{\partial x}\middle|x^*\right)\right| < 1\ \forall_i \tag{7}$$

Then there exists a control signal:

$$\hat{f}(k) = \frac{f\big(x(k)\big) + Nx^*}{N+1} \tag{8}$$

That ensures stabilization of the system at the fixed point x^*, provided that the scalar gain N is positive and sufficiently large. In this context, the control law in equation (8) can be rewritten as:

$$\hat{f}(k) = g_m f_x(k) + g_s x^* \tag{9}$$

Where g_m is referred to as the master parameter and g_s as the slave parameter. These coefficients satisfy:

$$g_m = \frac{1}{N+1} \tag{10}$$

$$g_s = \frac{N}{N+1} \tag{11}$$

Ensuring that $g_m+g_s=1$.

The roles of the master and slave parameters are fundamental in shaping the system's response. The master parameter g_m controls the influence of the original (chaotic) system on the overall dynamics, while the slave parameter g_s dictates the extent to which the control input nudges the system toward the fixed point x^*. When g_s is close to 1 (and hence g_m close to 0), the system exhibits strong convergence toward x^*; conversely, when g_m dominates, chaotic behavior re-emerges. Thus, to ensure stabilization, N must be chosen sufficiently large, as required by the FPIC theorem.

This theoretical formulation serves as the basis for implementing FPIC in various types of chaotic maps, as detailed in the following sections. Each system is stabilized using the control law in equation (9), with fixed-point arithmetic tailored for real-time hardware execution.

2.3. Controller Design

The implementation of FPIC control begins with the mathematical reformulation of the original chaotic system by incorporating the control input. Starting from equation (6), the controlled system can be expressed as:

$$x(k+1) = g_m f\big(x(k)\big) + g_s x^* \tag{12}$$

where x^* is the desired fixed point of the system. Substituting the definitions of g_m and g_s from equations (10) and (11), the equation becomes:

$$x(k+1) = \frac{1}{1+N} f\big(x(k)\big) + \frac{N}{1+N} x^* \tag{13}$$

In this form, the control law blends the original chaotic function with a constant input derived from the desired fixed point. The effectiveness of the controller relies on selecting a value of N (and hence g_s) that ensures convergence toward x^*.

To determine the fixed point for each chaotic system, the steady-state condition is applied, assuming the system reaches equilibrium when $x(k+1)=x(k)=x^*$. This leads to:

$$x^* = f(x^*) \tag{14}$$

Solving equation (14) for each chaotic map yields the corresponding fixed points [25]:

- Logistic Map

$$x^* = 1 - \frac{1}{r} \tag{15}$$

- Fold Map

$$x^* = \pm\sqrt{r} \tag{16}$$

- Flip Map

$$x^* = 0, \qquad x^* = \pm\sqrt{-r} \tag{17}$$

These expressions define the fixed points at which control is applied. Notably, while the Logistic Map has a single stable fixed point, the Fold and Flip Maps may yield two or more possible fixed points depending on the sign and magnitude of the gain parameter r.

An essential conceptual insight from equation (13) is the interpretation of g_m as the weighting of the original (unstable) system dynamics, while g_s governs the strength of the control action. Since $g_m+g_s=1$, increasing g_s enhances stability but reduces system responsiveness to chaotic behavior, and vice versa. Therefore, a proper selection of N is crucial to ensure a balance between stability and dynamical performance.

By substituting the expressions of the chaotic maps (1), (2), and (3) into equation (13), the controlled versions of the Logistic, Fold, and Flip systems are obtained. Additionally, for linearly coupled systems, the FPIC-controlled equations are derived in the form:

$$x(k+1) = g_m f_x\big(x(k)\big) + g_s x^* \quad (18)$$

$$y(k+1) = g_m f_y\big(y(k)\big) + g_s y^* \quad (19)$$

Where f_x and f_y denote the chaotic maps, and x^*, y^* are the respective fixed points. In this configuration, both subsystems are synchronized in their gain and control parameters, resulting in symmetrical controlled dynamics. This design methodology enables the practical implementation of chaos control through simple arithmetic operations—additions and multiplications—which are efficiently handled using fixed-point representation in FPGA hardware.

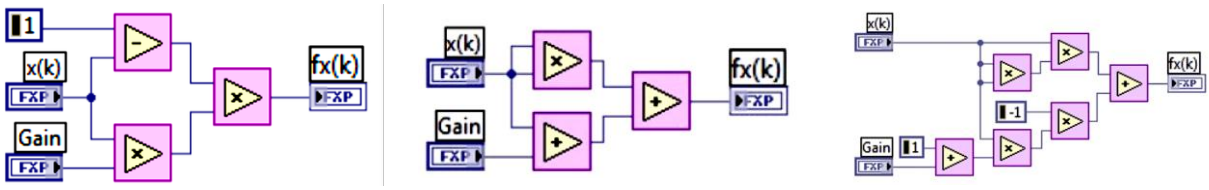
2.4. FPIC Control Implementation in Chaotic Systems

The experimental implementation of chaotic systems and their respective FPIC controllers was conducted according to the methodology described in [25]. LabVIEW-FPGA was used to translate the control algorithms into hardware and deploy them on a Virtex II FPGA. This platform enables graphical programming and real-time deployment on reconfigurable hardware (FPGAs), facilitating rapid prototyping and efficient use of hardware resources [17], [26]. For software validation, MATLAB was employed due to its robust numerical capabilities, especially in simulating nonlinear dynamic systems and generating bifurcation diagrams [19], [27].

All calculations were performed using 16-bit fixed-point representation, with 5 bits assigned to the integer part and 11 bits to the fractional part. This precision level was selected as a trade-off between computational accuracy and hardware efficiency. Figure 1 illustrates the block diagrams for the three chaotic functions implemented in LabVIEW FPGA.

These diagrams directly correspond to the mathematical definitions in equations (1), (2), and (3). For simplicity, feedback paths (used to store the previous value *x(k)*) are not shown in the figure. In practice, feedback was implemented by initializing registers with the initial condition *x(0)* before execution.

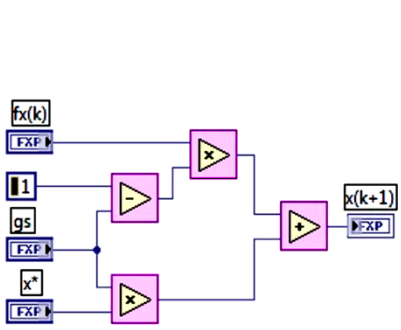
Figure 1. Implemented chaotic functions: (a) Logistic, (b) Fold, (c) Flip. All have fixed-point (FXP) inputs *x(k)*, *Gain*, and the constant 1; the output is fixed-point $f_x(k)$, and the boxes represent the computations.
Figura 1. Funciones caóticas utilizadas: (a) Logística, (b) Pliegue, (c) Volteo. Todas las funciones tienen entradas de punto-fijo *x(k)*, *Gain*, y la constante 1, la salida es de punto-fijo $f_x(k)$, y las cajas representan los cálculos realizados.



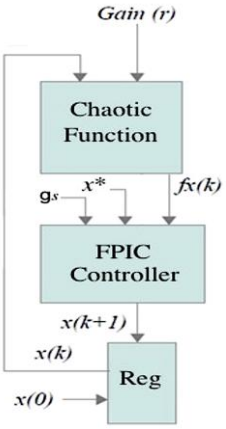
(a) (b) (c)

The FPIC control implementation is illustrated in Figure 2. In this setup, the slave parameter g_s is defined as a configurable input, while the master parameter is computed internally as $g_m=1-g_s$. The user can also input the gain parameter r and the fixed point x^*. The system then performs a sweep over a specified range of g_s values to study the effect of control strength on the system dynamics.

Figure 2. FPIC control implementation: (a) FPIC controller, (b) General connection diagram. Inputs $f_x(k)$, 1, g_s, and x*, and output x(k+1) are formatted in fixed-point (FXP) representation. Boxes represent the calculations.
Figura 2. Implementación del control FPIC: (a) Controlador FPIC, (b) Diagrama general de conexión. Las entradas $f_x(k)$, 1, g_s, and x*, y la salida x(k+1) tienen formato de punto-fijo (FXP). Las cajas representan las operaciones realizadas.



(a)



(b)

It is important to note that the analytical calculation of the fixed point was not performed in hardware for the Fold and Flip maps, due to the square root operations involved. Square root calculations typically require sequential hardware circuits that span multiple clock cycles, reducing system performance. In order to maintain one-clock-cycle combinatorial logic for all arithmetic operations, fixed-point values of x^* were precomputed externally and loaded into the system. In contrast, the Logistic Map allows for the direct computation of x^* from a simple expression (equation 15), which can be implemented directly in the FPGA logic. Figure 2(b) shows the complete control diagram, where the chaotic function block is connected to the FPIC control logic and the parameter sweep module.

To collect and visualize system behavior under different control configurations, a testbench was developed using a two-level loop structure in the LabVIEW-FPGA environment. The outer loop (green) controls the incremental variation of the slave parameter g_s, based on initial and final values defined by the user. For each value of g_s, the inner loop (yellow) executes the chaotic system and stores the output values in on-chip memory using a Direct Memory Access (DMA) module.

During each iteration, the chaotic system is allowed to evolve over 4000 time steps. To avoid transient effects, only the final 128 output values are saved in memory. Each output sample is paired with the corresponding g_s value, resulting in 128 × 128 = 16,384 saved data points. The saved values are 16 bits wide, stored as pairs (g_s, $x(k)$) in 32-bit memory addresses. This data is later extracted for post-processing and visualization in MATLAB or other analysis tools.

This setup enables the real-time generation of bifurcation diagrams and the dynamic analysis of controlled chaotic systems with fine-grained resolution, providing a valuable platform for testing and validating FPIC-based chaos controllers in hardware.

2.5. Coupled Chaotic Systems

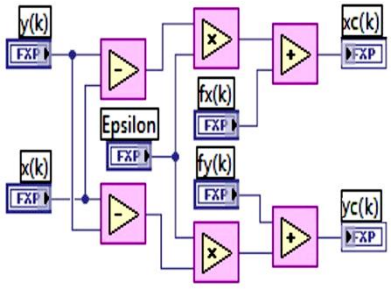
The implementation of coupled chaotic systems was conducted using the structure shown in Figure 3. In this configuration, two identical chaotic maps are connected through a linear coupling mechanism, enabling the analysis of higher-dimensional chaotic behaviors while maintaining simplicity in hardware implementation.

In the controlled coupled system, the FPIC control strategy is applied simultaneously to both subsystems. The same gain parameter r, fixed point $x^*=y^*$, and control parameters g_s and g_m are used for each chaotic map. This symmetry ensures that the dynamic behavior of both systems remains synchronized, simplifying the overall control logic.

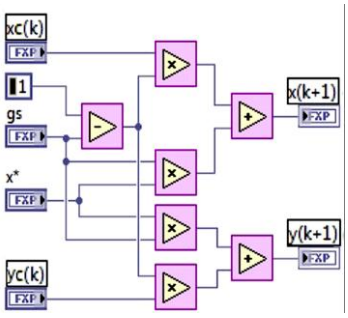
As illustrated in Figure 3(b), each chaotic subsystem is controlled individually using the FPIC approach described in previous sections. The linear coupling is introduced by allowing each subsystem to influence the other through a weighted term. However, in this work, the coupling is purely through shared parameters, rather than direct cross-feedback. Figure 3(c) shows the general connection diagram for the coupled implementation.

By using the same control gains and fixed points in both chaotic maps, the system maintains balance and exhibits predictable behavior under control. This setup also facilitates scalability to more complex configurations, including asymmetric or heterogeneous coupling, which may be explored in future work. The controlled coupled system provides a framework for evaluating synchronization and stabilization strategies in multidimensional chaotic environments, with potential applications in secure communication networks and distributed nonlinear systems.

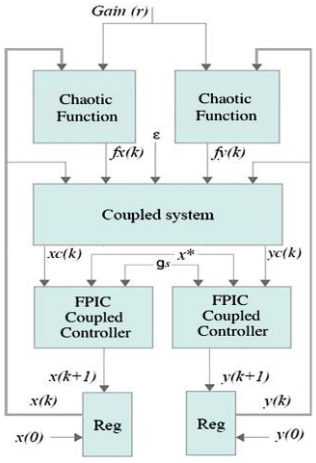
Figure 3. Coupled implementation: (a) Coupled system, (b) Coupled controller, (c) System connection diagram. All inputs and outputs are formatted in fixed-point (FXP) representation, and the boxes represent the performed calculations by each block.
Figura 3. Implementación del sistema acoplado: (a) Sistema acoplado, (b) Controlador acoplado, (c) Diagrama de interconexión del sistema. Todas las entradas y salidas están representadas en punto-fijo (FXP), y las cajas representan los cálculos realizados por cada bloque.



(a)



(b)



(c)

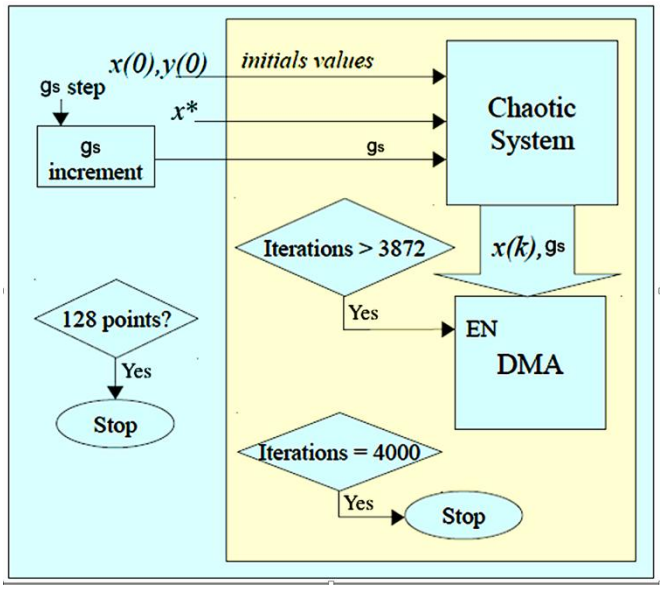
2.6. Test System for Controlled Chaotic Maps

Figure 4 presents the flow diagram of the test system developed for analyzing the controlled chaotic maps on an FPGA. The architecture consists of two nested loops: an outer loop responsible for regulating the variation of the slave parameter g_s, and an inner loop that executes the chaotic system under test and manages data acquisition. The outer loop (shown in green) iteratively increases g_s from a user-defined initial value to a final value, computing the step size accordingly. This loop determines the number of distinct values of g_s that will be evaluated. For this project, the number of steps was fixed to 128, meaning that 128 different values of g_s are tested in each experiment.

The inner loop (shown in yellow) simulates the evolution of the chaotic system for each value of g_s. It performs 4000 iterations to allow the system to reach a steady-state behavior, thereby minimizing the influence of initial transients. However, only the final 128 output samples are stored for analysis, as these represent the system's long-term response. To record the results, a Direct Memory Access (DMA) module was used to store output values in the FPGA's internal RAM. Each entry in memory corresponds to a pair (g_s,$x(k)$), stored as two 16-bit values in a 32-bit memory word. A total of 128 output values are recorded for each of the 128 g_s steps, resulting in 16,384 saved data points per test run.

It is essential to note that the initial conditions, $x(0)$ and $y(0)$, remain constant throughout the experiment, as they are hard-coded into the register's initialization. Changing these values requires a new synthesis of the FPGA design. If a single (non-coupled) system is tested, the second initial condition $y(0)$ is ignored. This test system enables high-resolution visualization of the system's response across a range of control strengths, allowing for the generation of bifurcation diagrams and precise characterization of stable and chaotic regimes. The results obtained from this framework are presented and discussed in the following section.

Figure 4. General flow diagram of the FPGA VI (Virtual Instruments)
Figura 4. Diagrama general de flujo del Instrumento Virtual (VI) en FPGA

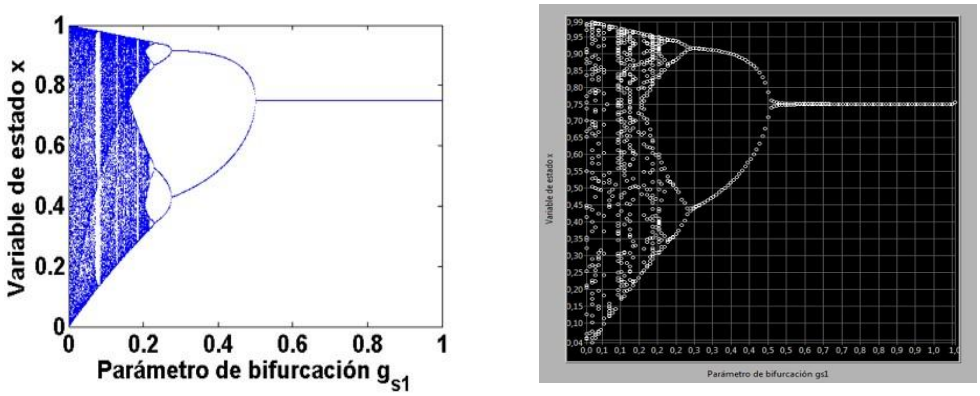


3. RESULTS AND DISCUSSION

3.1. Logistic Map Control Experiments

The first set of experiments focused on the Logistic Map, both in its uncontrolled and FPIC-controlled versions. Initially, the system was tested without control, and the results were compared to Matlab simulations to validate the accuracy of the FPGA implementation. The uncontrolled system showed good agreement with simulation results, as previously reported in [25]. Subsequently, the FPIC control strategy was applied, and the system behavior was evaluated through bifurcation diagrams. Figure 5 presents the results for the controlled Logistic Map with initial condition $x(0) = 0.4$, gain parameter $r = 4$, and fixed point $x^* = 0.75$, as computed from equation (15). The value $r=4$ was chosen because, in the absence of control, the system exhibits chaotic behavior at this gain level, thus providing a suitable test case for stabilization using FPIC.

Figure 5. Controlled logistic map, $x(0)=0.4$, $r=4$, $x^*=0.75$, $g_s=[0,1]$: (a) Matlab simulation, (b) LabVIEW-FPGA Response
Figura 5. Mapa logístico controlado, $x(0)=0.4$, $r=4$, $x=0.75$, $g_s=[0,1]$: (a) Simulación en Matlab, (b) Respuesta en LabVIEW-FPGA



(a) (b)

Two notable bifurcation points are visible in the diagram. The first is a flip bifurcation to a period-2 orbit occurring around $g_s \approx 0.276$, and the second is a flip bifurcation to a period-1 orbit near $g_s \approx 0.5$. For low values of g_s, the system remains chaotic, while increasing leads to greater stability. For $g_s > 0.5$, the system becomes fully stabilized around the fixed point $x=0.75$, as expected from the theory. Although both Matlab and FPGA implementations exhibit similar bifurcation structures, some minor differences were observed. In the MATLAB simulation, the system demonstrates strong convergence to a single output value beyond $gs = 0.5$. However, in the FPGA response, minor oscillations are visible near the fixed point around $g_s=0.5$. For instance, the output slightly alternates between two values close to 0.75, indicating a weaker stability margin in this region.

This discrepancy prompted a fixed-point simulation in MATLAB using the same 16-bit precision as in the FPGA. The simulation confirmed that hardware errors did not cause the oscillations but rather are inherent to the reduced numerical resolution. When using floating-point precision, the oscillations disappear, further validating the impact of fixed-point representation on system dynamics. The coupled Logistic Map implementation yielded results similar to the single system case. The bifurcation structure and stability transition points were preserved, although minor discrepancies were observed around the transition

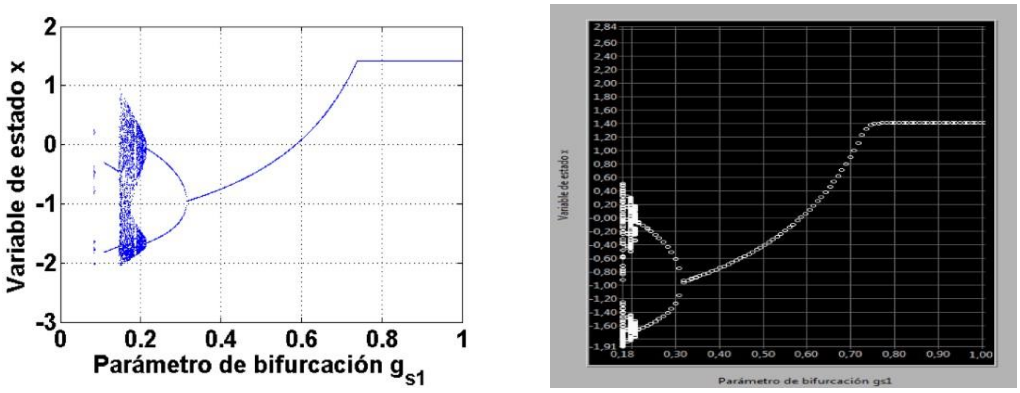
thresholds in the results between MATLAB and FPGA. These are consistent with the behavior described above and are attributable to the fixed-point arithmetic in the hardware implementation.

3.2. Fold Map Experiments

The next set of experiments focused on the Fold Map. Both the standalone and the coupled versions of this chaotic system were evaluated under FPIC control. The initial condition for the single system was set to $x(0) = 0$, with a gain parameter $r = -2$, resulting in a fixed point $x^* = \pm 1.4142$, as per equation (16). This value of r was selected because it induces instability in the uncontrolled system, providing a good scenario to evaluate the controller's effectiveness.

Figure 6 displays the bifurcation diagrams obtained from the MATLAB simulation and the FPGA implementation of the controlled Fold Map. The system reaches a stable regime for $g_s > 0.325$; however, it does not converge to the analytically computed fixed point x^* until $g_s > 0.748$. For intermediate values of g_s, the system stabilizes to other values depending on the strength of the control, highlighting the progressive influence of the FPIC mechanism. The differences between the MATLAB and FPGA results in this case are minimal and can be considered negligible. As with the Logistic Map, a slight discrepancy appears around the transition to stability, but it does not significantly impact the interpretation of the system's behavior.

Figure 6. Controlled Fold Map $x(0)=0$, $r=-2$, $x^*=1.4142$, $g_s=[0,1]$. (a) Matlab Simulation, (b) LabVIEW-FPGA Response
Figura 6. Mapa Fold controlado con $x(0)=0$, $r=-2$, $x^*=1.4142$, $g_s=[0,1]$. (a) Simulación en Matlab, (b) Respuesta en LabVIEW-FPGA



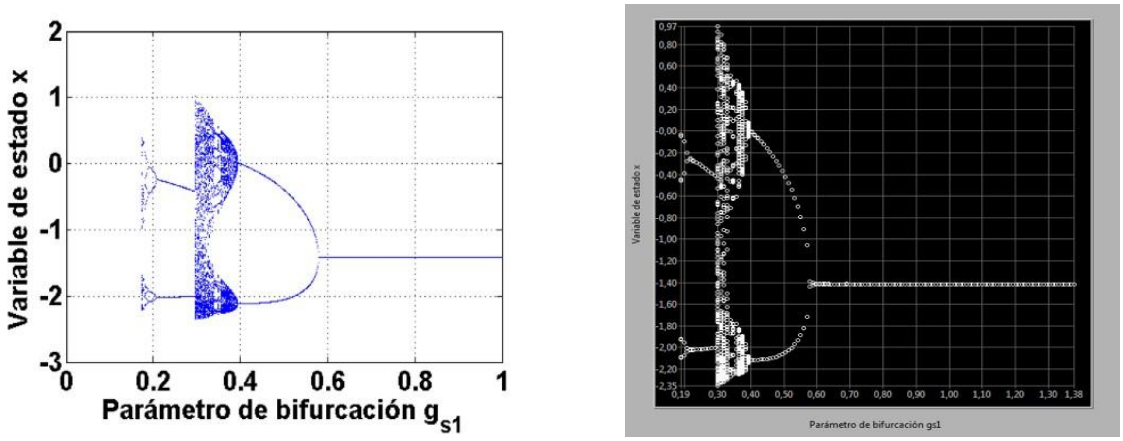
(a) (b)

Figure 7 shows the response of the controlled coupled Fold Map, where initial conditions were set to $x(0)=0.4$, $y(0)=0.3$, with gain $r=-2$, coupling coefficient $\alpha=0.275$, and fixed points $x=y^*=-1.4142$. The system becomes stable for $g_s>0.582$, with convergence to the fixed point visible beyond $g_s\approx 0.748$. As in the previous case, for intermediate values of g_s, the system reaches a stable regime but not necessarily at the fixed point.

Again, the differences between the Matlab simulation and FPGA implementation are present but limited. These discrepancies are primarily observed near the stability threshold and can be attributed to fixed-point resolution constraints, as previously discussed. Additional experiments with other values of x^*, including the negative branch of the solution $x^* = -\sqrt{2}$, produced consistent results. The system exhibited similar

stabilization behavior and convergence characteristics under FPIC control, confirming the reproducibility and reliability of the proposed hardware implementation.

Figure 7. Controlled Coupled Fold Map $x(0)=0.4$, $y(0)=0.3$, $r=-2$, $\varepsilon=0.275$. (a) $x^*=y^*=-1.4142$, $g_s=[0,1]$. (a) Matlab Simulation, (b) LabVIEW-FPGA Response
Figura 7. Mapa Fold acoplado y controlado con $x(0)=0.4$, $y(0)=0.3$, $r=-2$, $\varepsilon=0.275$. (a) $x^*=y^*=-1.4142$, $g_s=[0,1]$. (a) Simulación en Matlab, (b) Respuesta en LabVIEW-FPGA



(a) (b)

3.3. Flip Map Experiments

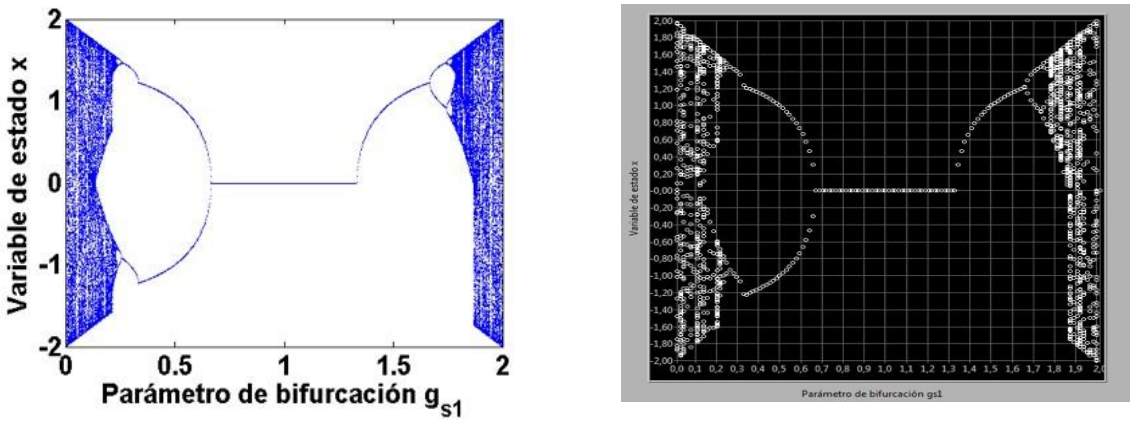
The final set of experiments evaluated the Flip Map under FPIC control. As with the previous systems, both standalone and coupled configurations were tested to examine the behavior of the control strategy in different settings. Figure 8 shows the results for the controlled Flip Map with initial condition $x(0) = 4$, gain parameter $r = 2$, and fixed point $x^* = 0$, in accordance with equation (17). In this case, the slave parameter g_s was varied over the interval [0, 2], allowing for the observation of the system's behavior under both weak and strong control influences. The system exhibits stable behavior around the fixed point $x^*=0$ for the interval $0.67 < g_s < 1.34$. Interestingly, this experiment shows that the system can remain stable even for $g_s > 1$, although stability is eventually lost when g_s exceeds approximately 1.65. This highlights the importance of proper parameter selection in FPIC: excessive control input may lead to destabilization rather than convergence.

Figure 9 presents the response of the controlled coupled Flip Map, using initial conditions $x(0)=4$, $y(0)=0.3$, gain $r=2$, coupling factor $\alpha=0.275$, and fixed points $x^*=y^*=-2$. In this configuration, the system reaches a stable regime for $g_s>0.6$; however, convergence to the fixed point $x^*=y^*=-2$ is only observed beyond $g_s \approx 0.88$. These results are consistent with the theoretical behavior of the FPIC-controlled Flip Map. As with previous systems, intermediate values lead to stabilization in regions close to, but not precisely at, the fixed point. This phenomenon is expected due to the gradual influence of the control signal and the nonlinear dynamics of the map.

Additional experiments were conducted for both the standalone and coupled Flip Map using different fixed points, including $x^*=2$ and $x^*=-2$, confirming the robustness of the FPIC strategy. The system consistently transitioned from chaotic behavior to stable convergence as the gain (g_s) increased, and the observed behavior aligned closely with both MATLAB simulations and FPGA implementations. Minor deviations,

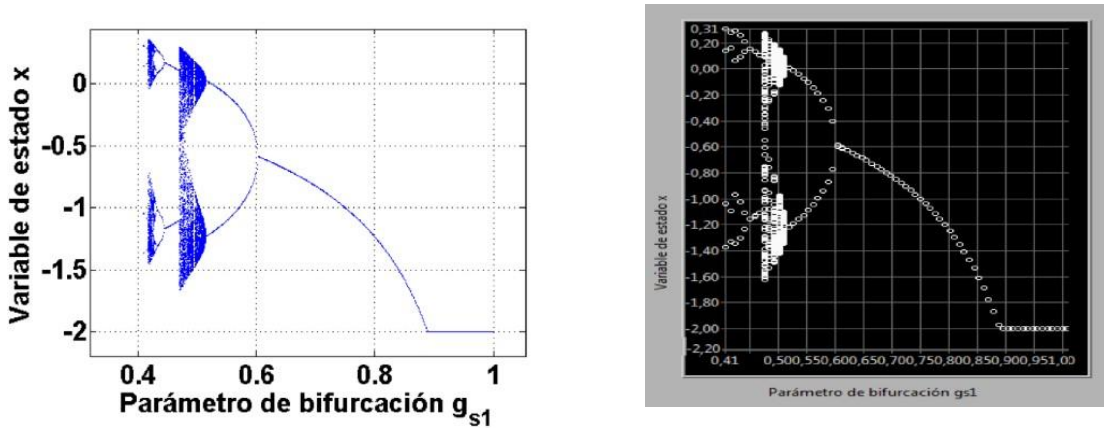
as observed in previous cases, were attributed again to the effects of fixed-point arithmetic on system resolution.

Figure 8. Controlled Flip Map *x(0)=0.4, r=2, x^*=0 and g_s=[0,2]*. (a) Matlab Simulation, (b) FPGA Response
Figura 8. Mapa Flip controlado con *x(0)=0.4, r=2, x^*=0 and g_s=[0,2]*. (a) Simulación en Matlab, (b) Respuesta en FPGA



(a) (b)

Figure 9. Controlled Coupled Flip Map *x(0)=0.4, y(0)=0.3, ε=0.275, r=2, x^*=y^*=-2 and gs=[0,1]*. (a) Matlab Simulation, (b) FPGA Response
Figura 9. Mapa Flip acoplado controlado con *x(0)=0.4, y(0)=0.3, ε=0.275, r=2, x^*=y^*=-2 and gs=[0,1]*. (a) Simulación en Matlab, (b) Respuesta en FPGA



(a) (b)

Compared to previous studies that implemented chaos control or chaotic signal generation on FPGA hardware, the present work provides a simplified and fully fixed-point implementation of the FPIC strategy, enabling real-time bifurcation analysis and stabilization. For example, Trujillo et al. applied FPIC along with ZAD and TDAS control strategies in a Boost converter, but their implementation was limited to simulations and did not explore FPGA-based hardware realization or bifurcation tracking [10]. Similarly, Guillén-Fernández et al. focused on chaotic oscillator synchronization for secure communications using

FPGA, but without direct control of bifurcations or parameter sweeps [22]. Mohamed et al. proposed fractional-order chaotic systems in FPGA, yet their implementation involved floating-point operations and lacked real-time adaptability [20]. More recently, Damaj et al. developed high-speed Lorenz system cores on FPGA, prioritizing throughput and precision [28], while Babajans et al. explored hybrid synchronization of analog–digital chaotic oscillators using fixed-point arithmetic [29]. However, neither addressed stabilization through live parameter control. In contrast, our approach combines analytical fixed-point design with dynamic parameter sweeps, demonstrating both control effectiveness and low-resource FPGA deployment suitability.

4. CONCLUSIONS

This work demonstrates the successful hardware implementation of FPIC-controlled chaotic systems using LabVIEW and an FPGA. Three well-known one-dimensional chaotic maps—the Logistic, Fold, and Flip Maps—were modeled, analyzed, and stabilized using the Fixed-Point Inducing Control technique. Both standalone and linearly coupled configurations were evaluated, providing insight into the scalability and flexibility of the proposed control approach. All systems were implemented on a Virtex II FPGA using fixed-point arithmetic with 16-bit precision. The combination of LabVIEW-FPGA's graphical programming interface and the efficient use of FPGA resources enabled rapid prototyping and real-time testing. The FPGA-based implementations were validated through comparison with Matlab simulations, showing a high degree of agreement, with minor discrepancies attributable to the limitations of fixed-point representation.

The FPIC strategy proved to be effective in stabilizing chaotic dynamics and driving the system to its analytically determined fixed points. Moreover, the results revealed how control strength, defined by the slave parameter g_s, influences system behavior and convergence properties. Stability regions were successfully identified for all systems, including critical bifurcation points and ranges of g_s leading to convergence or divergence. The developed prototyping platform offers a valuable tool for researchers exploring chaos control in real-time embedded systems. The simplicity of the FPIC control law and its suitability for fixed-point arithmetic make it an excellent candidate for low-power, high-speed applications such as secure communications, cryptographic hardware, and nonlinear sensor networks. However, the proposed approach has certain limitations. The use of fixed-point arithmetic imposes constraints on numerical precision, which may affect stability in highly sensitive systems. Additionally, the scalability of the implementation to more complex or higher-dimensional chaotic systems may be limited by the availability of FPGA resources and increased computational demands.

Future efforts will explore the implementation of higher-dimensional chaotic systems by coupling multiple distinct chaotic maps. Nonlinear coupling schemes and heterogeneous topologies will also be investigated, extending the current approach toward more complex and realistic models. Additionally, FPGA implementations of chaotic systems based on trigonometric functions or piecewise-smooth dynamics, such as Chua's circuit and the Tent Map, are planned. Furthermore, we plan to integrate adaptive control strategies with the FPIC framework to enable real-time tuning of controller parameters, enhancing robustness under varying system dynamics. Finally, the integration of adaptive control strategies and intelligent tuning of FPIC parameters using machine learning or optimization algorithms represents a promising direction for increasing the robustness and autonomy of chaos-based controllers in real-world applications. Particularly, the exploration of machine learning techniques to optimize the g_m and g_s parameters could lead to more efficient control laws and broaden the applicability of FPIC to nonlinear, high-order, and coupled chaotic systems.

REFERENCES

[1] L. Xiao, W. Jianzhen, and L. Hongqin, "Nonlinear System Identification based on Orthonormal Wavelet," *IJIREEICE*, vol. 4, no. 10, pp. 82–85, Oct. 2016, doi: 10.17148/IJIREEICE.2016.41017.

[2] F. Yang, X. An, and L. xiong, "A new discrete chaotic map application in image encryption algorithm," *Phys Scr*, vol. 97, no. 3, p. 035202, Mar. 2022, doi: 10.1088/1402-4896/ac4fd0.

[3] P. Fang, L. Dai, Y. Hou, M. Du, and W. Luyou, "The Study of Identification Method for Dynamic Behavior of High-Dimensional Nonlinear System," *Shock and Vibration*, vol. 2019, no. 1, Jan. 2019, doi: 10.1155/2019/3497410.

[4] N. Nguyen, L. Pham-Nguyen, M. B. Nguyen, and G. Kaddoum, "A Low Power Circuit Design for Chaos-Key Based Data Encryption," *IEEE Access*, vol. 8, pp. 104432–104444, 2020, doi: 10.1109/ACCESS.2020.2998395.

[5] A. Iqbal, "Chaos control of brushless direct current motor using sliding mode control with a low cost hardware-in-loop validation," *Science Talks*, vol. 14, p. 100453, Jun. 2025, doi: 10.1016/j.sctalk.2025.100453.

[6] J. Chen *et al.*, "Intelligent robust control for nonlinear complex hydro-turbine regulation system based on a novel state space equation and dynamic feedback linearization," *Energy*, vol. 302, p. 131798, Sep. 2024, doi: 10.1016/j.energy.2024.131798.

[7] D. Das, I. Taralova, and J. J. Loiseau, "Time-delay Feedback Control of Fractional Chaotic Rössler Oscillator," *IFAC-PapersOnLine*, vol. 58, no. 5, pp. 90–95, 2024, doi: 10.1016/j.ifacol.2024.07.069.

[8] Y. Zhang *et al.*, "Chaotic band-gap modulation mechanism for nonlinear vibration isolation systems based on time-delay feedback control," *J Phys D Appl Phys*, vol. 58, no. 1, p. 015311, Jan. 2025, doi: 10.1088/1361-6463/ad8008.

[9] N. T. García, Y. A. G. Gomez, and V. H. Cespedes, "Robust control technique in power converter with linear induction motor," *International Journal of Power Electronics and Drive Systems (IJPEDS)*, vol. 13, no. 1, p. 340, Mar. 2022, doi: 10.11591/ijpeds.v13.i1.pp340-347.

[10] S. C. Trujillo, J. E. Candelo-Becerra, and F. E. Hoyos, "Analysis and Control of Chaos in the Boost Converter with ZAD, FPIC, and TDAS," *Sustainability*, vol. 14, no. 20, p. 13170, Oct. 2022, doi: 10.3390/su142013170.

[11] F. E. Hoyos, J. E. Candelo-Becerra, and C. I. Hoyos Velasco, "Application of Zero Average Dynamics and Fixed Point Induction Control Techniques to Control the Speed of a DC Motor with a Buck Converter," *Applied Sciences*, vol. 10, no. 5, p. 1807, Mar. 2020, doi: 10.3390/app10051807.

[12] F. E. Hoyos Velasco, J. E. Candelo-Becerra, and A. Rincón Santamaría, "Dynamic Analysis of a Permanent Magnet DC Motor Using a Buck Converter Controlled by ZAD-FPIC," *Energies (Basel)*, vol. 11, no. 12, p. 3388, Dec. 2018, doi: 10.3390/en11123388.

[13] F. E. Hoyos, J. E. Candelo, and J. A. Taborda, "Selection and Validation of Mathematical Models of Power Converters using Rapid Modeling and Control Prototyping Methods," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 8, no. 3, p. 1551, Jun. 2018, doi: 10.11591/ijece.v8i3.pp1551-1568.

[14] F. E. Hoyos Velasco, J. E. Candelo, and J. I. Silva Ortega, "Performance evaluation of a DC-AC inverter controlled with ZAD-FPIC," *INGE CUC*, vol. 14, no. 1, pp. 9–18, Jan. 2018, doi: 10.17981/ingecuc.14.1.2018.01.

[15] C. Mohamed, K. Messaoudi, and L. Lamri, "Multi-level and real-time implementations using FPGA devices of PWM techniques used for the control of static converters," *J Supercomput*, vol. 81, no. 4, p. 525, Feb. 2025, doi: 10.1007/s11227-024-06905-0.

[16] A. Ravera, A. Oliveri, M. Lodi, and M. Storace, "FPGA Implementation of Nonlinear Model Predictive Control for a Boost Converter with a Partially Saturating Inductor," *Electronics (Basel)*, vol. 14, no. 5, p. 941, Feb. 2025, doi: 10.3390/electronics14050941.

[17] A. A. Nada and M. A. Bayoumi, "Development of embedded fuzzy control using reconfigurable FPGA technology," *Automatika*, vol. 65, no. 2, pp. 609–626, Apr. 2024, doi: 10.1080/00051144.2024.2313904.

[18] Q. Wang *et al.*, "Theoretical Design and FPGA-Based Implementation of Higher-Dimensional Digital Chaotic Systems," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 63, no. 3, pp. 401–412, Mar. 2016, doi: 10.1109/TCSI.2016.2515398.

[19] M. Y. Hamada, "Investigating the dynamics of generalized discrete logistic map," *Math Methods Appl Sci*, vol. 48, no. 4, pp. 5325–5336, Mar. 2025, doi: 10.1002/mma.10606.

[20] S. M. Mohamed, W. S. Sayed, L. A. Said, and A. G. Radwan, "Reconfigurable FPGA Realization of Fractional-Order Chaotic Systems," *IEEE Access*, vol. 9, pp. 89376–89389, 2021, doi: 10.1109/ACCESS.2021.3090336.

[21] S. Liu, Y. Wei, J. Liu, S. Chen, and G. Zhang, "Multi-Scroll Chaotic System Model and Its Cryptographic Application," *International Journal of Bifurcation and Chaos*, vol. 30, no. 13, p. 2050186, Oct. 2020, doi: 10.1142/S0218127420501862.

[22] O. Guillén-Fernández, A. Meléndez-Cano, E. Tlelo-Cuautle, J. C. Núñez-Pérez, and J. de J. Rangel-Magdaleno, "On the synchronization techniques of chaotic oscillators and their FPGA-based implementation for secure image transmission," *PLoS One*, vol. 14, no. 2, p. e0209618, Feb. 2019, doi: 10.1371/journal.pone.0209618.

[23] F. Capligins, A. Litvinenko, D. Kolosovs, M. Terauds, M. Zeltins, and D. Pikulins, "FPGA-Based Antipodal Chaotic Shift Keying Communication System," *Electronics (Basel)*, vol. 11, no. 12, p. 1870, Jun. 2022, doi: 10.3390/electronics11121870.

[24] C. Mayo-Wilson, "Structural Chaos," *Philos Sci*, vol. 82, no. 5, pp. 1236–1247, Dec. 2015, doi: 10.1086/684086.

[25] H. Castro and J. A. Taborda, "Rapid prototyping of chaotic generators using LabView-FPGA," in *2012 IEEE 4th Colombian Workshop on Circuits and Systems (CWCAS)*, IEEE, Nov. 2012, pp. 1–6. doi: 10.1109/CWCAS.2012.6404075.

[26] National Instruments, "LabVIEW User Manual." Accessed: Jul. 06, 2025. [Online]. Available: https://www.ni.com/docs/en-US/bundle/labview/page/user-manual-welcome.html

[27] MathWorks, "MATLAB." Accessed: Jul. 06, 2025. [Online]. Available: https://la.mathworks.com/help/matlab/index.html?s_tid=hc_panel

[28] I. Damaj, A. Zaher, and W. Lawand, "Optimizing FPGA implementation of high-precision chaotic systems for improved performance," *PLoS One*, vol. 19, no. 4, p. e0299021, Apr. 2024, doi: 10.1371/journal.pone.0299021.

[29] R. Babajans, D. Cirjulina, and D. Kolosovs, "Field-Programmable Gate Array-Based Chaos Oscillator Implementation for Analog–Discrete and Discrete–Analog Chaotic Synchronization Applications," *Entropy*, vol. 27, no. 4, p. 334, Mar. 2025, doi: 10.3390/e27040334.