Eur. Phys. J. Special Topics **228**, 2065–2080 (2019)

https://doi.org/10.1140/epjst/e2019-900005-8

THE EUROPEAN PHYSICAL JOURNAL SPECIAL TOPICS

Regular Article

Dynamical analysis, sliding mode synchronization of a fractional-order memristor Hopfield neural network with parameter uncertainties and its non-fractional-order FPGA implementation

Karthikeyan Rajagopal[1], Murat Tuna[2], Anitha Karthikeyan[1], İsmail Koyuncu[3], Prakash Duraisamy[1], and Akif Akgul[4,a]

[1] Center for Nonlinear Dynamics, Defence University, Bishoftu, Ethiopia
[2] Department of Electrical, Technical Sciences Vocational School, Kırklareli University, 39000 Kırklareli, Turkey
[3] Department of Electrical and Electronics Engineering, Afyon Kocatepe University, 03200 Afyon, Turkey
[4] Department of Electrical and Electronics Engineering, Sakarya University of Applied Sciences, 54050 Sakarya, Turkey

Received 16 January 2019 / Received in final form 4 March 2019
Published online 14 October 2019

Abstract. Recent developments in the applications of neural networks in various engineering and technology applications have motivated researchers to study the nonlinear behavior of such networks. In this work we investigate a fractional-order Hopfield neural network with memristor synaptic weight. The dynamical properties of the proposed system are examined and the memristor neural network shows hyperchaotic attractors in fractional orders with hidden oscillations. We also propose an adaptive sliding mode control technique to synchronize the proposed fractional-order systems with uncertainties. Numerical simulations are derived to show the effectiveness of the synchronization algorithm. Moreover, the designed chaotic memristor Hopfield neural network system is realized on FPGA using the 4th-order Runge–Kutta (RK4) numerical algorithm. The FPGA-based chaotic memristor HNN is coded in VHDL using the 32-bit IEEE-754-1985 floating point standard. The chaotic memristor neural network designed on FPGA is synthesized and tested using Xilinx ISE. The chip statistics of Xilinx XC6VLX240T-1-FF1156 kit obtained from Place & Route operation for the designed RK4-based system is presented. The operating frequency of newly modeled FPGA-based memristor neural network chaotic signal generator is 231.616 MHz.

1 Introduction

Chua postulated a resistive device with potential hysteresis behavior and derived a series of rigorous mathematical proofs for the fourth fundamental circuit element

[a] e-mail: aakgul@subu.edu.tr

named "Memristor". Its unique property to remember the last functional state in the form of resistance widely opens up new functionalities in both analog and digital electronics [1,2]. The advent of these memristor endows the researches with promising access to mimic the behavior of neuron's synapse. Various memristor techniques are used to study the behavior as an artificial synapse [3–5]. Dynamical studies of neural network (NN) reveal that this technique is very effective [6,7]. The Hopfield Neural Network (HNN) identified a simple way of setting up connections between nodes so that "stable firing patterns" can be made [8].

According to initial research results [9–14], it appears that 3-Dimension (3D) continuous time HNNs do not display chaotic behavior especially in 3D piecewise linear networks of genes [15]. Recent numerical experiments prove that chaos can take place in some simple HNNs [7–9,16–18]. Sun and Shi worked the effect of channel blocks on the spiking regularity in clustered neural networks [19]. Yang and Huang discussed complex dynamics in simple HNNs [8]. Waser et al. identified high nonlinearity in iconic drift-diffusion, at the nanoscale level [3,20].

Fractional-order (FO) treatment of complex dynamic systems provides efficient computational results for information processing and frequent independent phase shifts in oscillating neuronal firing [21]. Kaslik et al. developed an FO-HNN model [22]. Dynamic analysis of such a model was studied and the stability and multistability properties found [23]. Especially Boroomand discussed the stability of FO-NNs [24]. The existence of a nontrivial solution of FO-HNN was analyzed [25]. FO bidirectional associative memory network stabilization was studied [26]. Hu Wang et al. introduced time delay in FO-HNN and global asymptotic stability conditions were obtained [21]. Global projective synchronization of FO-NN was studied [27]. In FO recurrent NNs, multiple Mittag-Leffler stability analyses were carried out [28]. The existence of stability and dissipative study on FO complex-valued NNs with time delay was studied [29,30].

Authors concentrated more on the self-excited attractors on the FO-HNN. However, to the best of our knowledge, hidden attractors influenced the system dynamics significantly. This paper was devoted to presenting the dynamic analysis and synchronization of FO-HNN with line equilibrium and no equilibrium using adaptive sliding mode controller (SMC).

2 Fractional-order memristor neural network (FOMNN)

The memristive neural network (MNN) model is defined [18] as

$$\begin{aligned}
\dot{x} &= -x + 1.6\tanh(x) + 2\tanh(y) + \tanh(z) \\
\dot{y} &= -y + (M(w))\tanh(x) + 1.5\tanh(y) \\
\dot{z} &= -z + 3\tanh(x) - 2\tanh(y) + \tanh(z) + c \\
\dot{w} &= \tanh(x)
\end{aligned} \tag{1}$$

where $M(w) = aw + bw^2$ is the memristor element. System (1) shows chaotic attractor for $a = -0.001$, $b = -0.05$, $c = -0.001$, and initial conditions (IC) [0, 0.01, 0.01, 0]. In this paper, we derive and investigate the fractional-order model of system (1) and derive their dynamic properties.

Caputo method is popular in the literature. It has been shown that the GL method of solving FO systems has smoothness over other methods due to the smoothness in the coefficients [31]. Hence, we chose the GL method to obtain the FOMNN. It can

be defined as

$$ {}_aD_t^q f(t) = \lim_{h\to 0}\left\{ \frac{1}{h^q} \sum_{j=0}^{\left[\frac{t-a}{h}\right]} (-1)^j \binom{q}{j} f(t-jh) \right\} = \lim_{h\to 0}\left\{ \frac{1}{h^\alpha} \Delta_h^q f(t) \right\} \tag{2} $$

where a and t are limits, $\Delta_h^q f(t)$ is the generalized difference, h is the step size and q is the FO.

$$ {}_{(t-L)}D_t^q f(t) = \lim_{h\to 0}\left\{ h^{-q} \sum_{j=0}^{N(t)} \beta_j \left(f(t-jh)\right) \right\} \tag{3} $$

As the memory required for the calculation of binomial coefficients is theoretically infinite, we truncate the number of samples for calculation using

$$ N = \min\left\{ \left[\frac{t}{h}\right], \left[\frac{L}{h}\right] \right\} \tag{4} $$

The binomial coefficients used in (3) are calculated as

$$ \beta_j = \left(1 - \frac{a+q}{j}\right) \beta_{j-1} \tag{5} $$

Let us define the FOMNN oscillator as

$$ \begin{aligned} D^{q_x} x &= -x + 1.6\tanh(x) + 2\tanh(y) + \tanh(z) \\ D^{q_y} y &= -y + (M(w))\tanh(x) + 1.5\tanh(y) \\ D^{q_z} z &= -z + 3\tanh(x) - 2\tanh(y) + \tanh(z) + c \\ D^{q_w} w &= \tanh(x) \end{aligned} \tag{6} $$

To simulate system (8) using the GL method, we use the discretization method discussed [31,32]:

$$ \begin{aligned} x(t_k) &= A(x(t_{k-1}), y(t_{k-1}), z(t_{k-1}), w(t_{k-1}))h^{q_x} - \sum_{j=1}^{N} \beta_j^{q_x} x(t_{k-j}) \\ y(t_k) &= B(x(t_{k-1}), y(t_{k-1}), z(t_{k-1}), w(t_{k-1}))h^{q_y} - \sum_{j=1}^{N} \beta_j^{q_y} y(t_{k-j}) \\ z(t_k) &= C(x(t_{k-1}), y(t_{k-1}), z(t_{k-1}), w(t_{k-1}))h^{q_z} - \sum_{j=1}^{N} \beta_j^{q_z} z(t_{k-j}) \\ w(t_k) &= C(x(t_{k-1}), y(t_{k-1}), z(t_{k-1}), w(t_{k-1}))h^{q_w} - \sum_{j=1}^{N} \beta_j^{q_w} w(t_{k-j}) \end{aligned} \tag{7} $$

where β is the binomial coefficient (5). N is taken as the truncation window size L if the available memory is not fully used as k. Using (7) in (6), the discrete form of

Table 1. Types of equilibrium in FOMNN

Case	Condition	Type of equilibrium	Lyapunov Exponents	Figure
Case 1	$c = 0$	Line equilibrium $(0, 0, 0, w)$	0.0289,0.0107,0,−0.115	1a
Case 2	$c \neq 0 (c = 0.001)$	No equilibrium	0.0287,0.0071,0, −0.111	1b

the FOMNN is given by

$$\begin{aligned}
x(t_k) &= \begin{pmatrix} -x(t_{k-1}) + 1.6\tanh(x(t_{k-1})) \\ +2\tanh(y(t_{k-1})) + \tanh(z(t_{k-1})) \end{pmatrix} h^{q_x} - \sum_{j=1}^{N} \beta_j^{q_x} x(t_{k-j}) \\
y(t_k) &= (-y(t_{k-1}) + (M(w(t_{k-1})))\tanh(x) + 1.5\tanh(y(t_{k-1}))) h^{q_y} - \sum_{j=1}^{N} \beta_j^{q_y} y(t_{k-j}) \\
z(t_k) &= (-z(t_{k-1}) + 3\tanh(x(t_{k-1})) - 2\tanh(y(t_{k-1})) + \tanh(z(t_{k-1})) + c)\, h^{q_z} \\
&\quad - \sum_{j=1}^{N} \beta_j^{q_z} z(t_{k-j}) \\
w(t_k) &= (\tanh(x(t_{k-1})))\, h^{q_w} - \sum_{j=1}^{N} \beta_j^{q_w} w(t_{k-j})
\end{aligned} \tag{8}$$

where $M(w) = aw(t_{k-1}) + bw(t_{k-1})^2$ is the memristor element.

For the parameters values given in (2) and Table 1, the FOMNN oscillator shows different behaviors like the integer-order MNN. For FO, $q = 0.995$, $h = 0.001$ $a = -0.001, b = -0.05, c = -0.001$, and ICs $[0, 0.01, 0.01, 0]$; system (9) shows a chaotic attractor as in Figure 1. As discussed in [18], the system shows two types of equilibrium points as in Table 1. The finite-time Lyapunov exponents (LEs) are obtained using [33,34] for 20 000 s and is presented in Table 1.

We derive the bifurcation of the FOMNN with c and ICs [0,0.01,0.01,0] reinitialized in every loop to the end values of the trajectories and plotting the local maxima as given in Figure 2a. This kind of bifurcation is commonly known as forward continuation. The corresponding LEs of the FOMNN are given in Figure 2b. When investigating a FO system, it is very important to study the importance of the FO's q on the system. To investigate, we consider the commensurate FO of the system with parameter values taken as $a = -0.001, b = -0.05, c = -0.001$ and reinitialize the ICs in every loop. Figure 3 shows the bifurcation of the FOMNN with the commensurate FO q and as seen from the plot, the FOMNN shows chaotic oscillations for $q \geq 0.961$.

3 Synchronization of FOMNN using Fractional-order adaptive sliding mode control

Chaotic systems are used to synchronize active control [35–38], adaptive control [39,40], extended back stepping control [41,42], sliding mode control [43,44], and adaptive sliding mode [45–47]. But synchronization in FO systems is very difficult because of its complexity [48,48,50–52] and hence has applications in developing robust cryptography algorithms [50].

The master system be defined as

$$D^{q_m} x = f(x) + F(x)a \tag{9}$$

and the slave system as

$$D^{q_s} y = g(y) + G(y)b + u \tag{10}$$

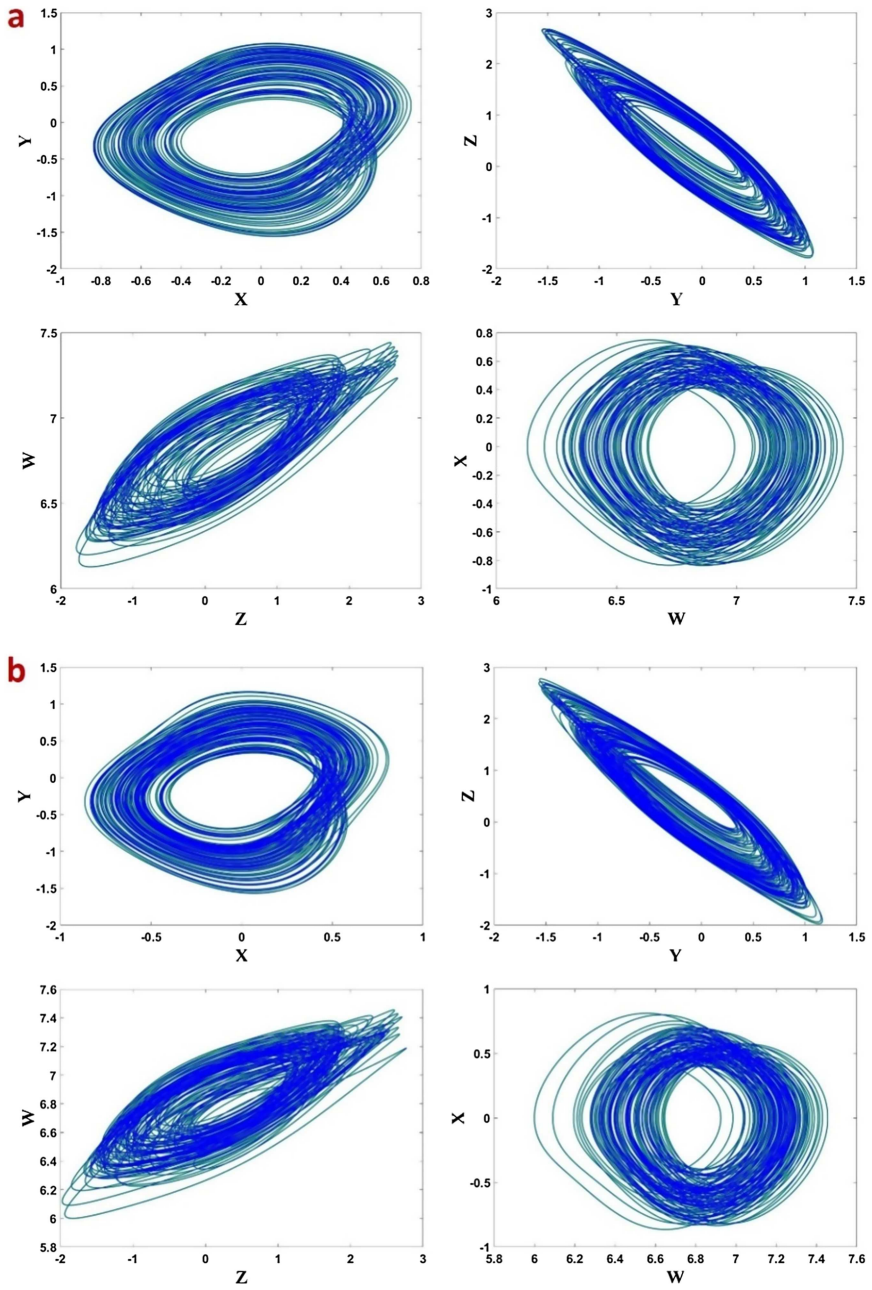

Fig. 1. Phase portraits of FOMNN as given in Table 1.

where $f(x), g(y)$ are $n \times 1$ row vector and $F(x), G(y)$ are $m \times n$ matrix elements, q_m, q_s are the FOs of the master and slave systems respectively, a, b are the unknown parameters of the systems and u is the controller to synchronize the systems.

With IC $y(0)$ with master system of IC $x(0)$, the synchronization errors (11) approach zero:

$$\lim_{t \to \infty} e = y - x \tag{11}$$

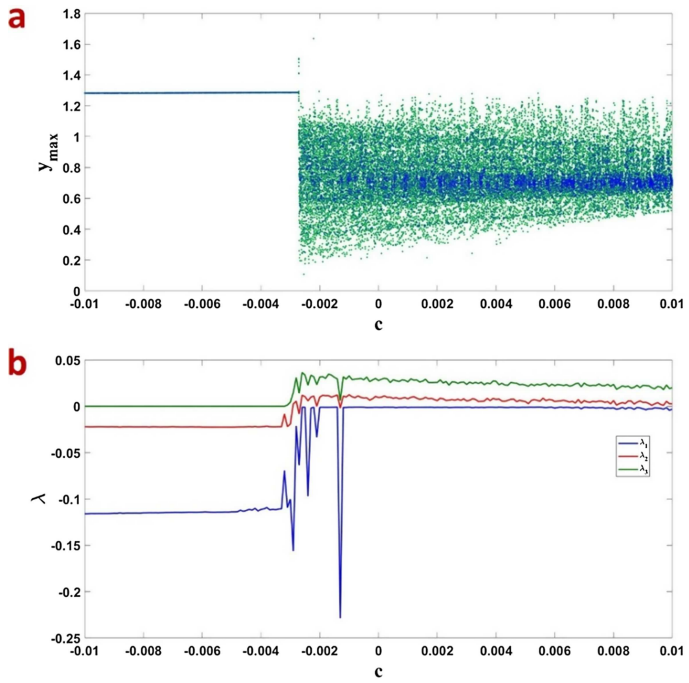

Fig. 2. a: Bifurcation of FOMNN with parameter c and FO kept at $q = 0.995$ for ICs $[0, 0.01, 0.01, 0]$ for first iteration and reinitialized to end values of state trajectories in every iteration. b: Corresponding LEs of FOMNN (fourth LE is out of scale).

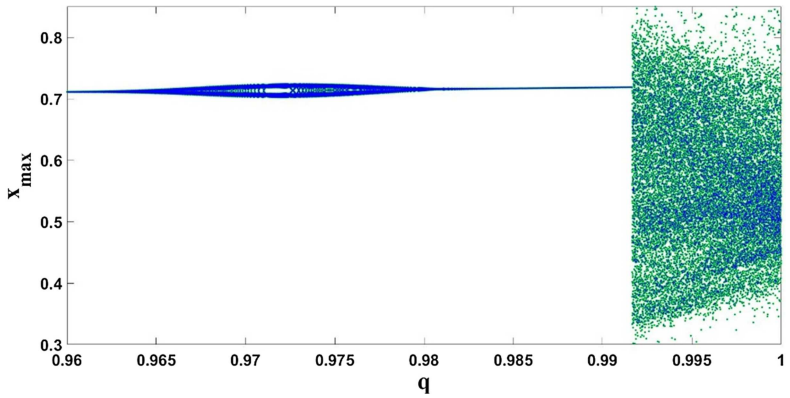

Fig. 3. Bifurcation of FOMNN with commensurate FO q with parameters $a = -0.001, b = -0.05, c = -0.001$ for ICs $[0, 0.01, 0.01, 0]$ for first iteration and reinitialized to end values of state trajectories in every iteration.

The proportional integral sliding surface [37] is defined as

$$s = e + K \int e(\tau) d\tau \tag{12}$$

where K is the proportional constant vector. The fractional first derivative of the sliding surface is obtained as

$$D^q s = D^q e + Ke \tag{13}$$

The FO error dynamics can be derived using (9), (10), (11) as

$$D^q e = g(y) + G(y)b + u - f(x) - F(x)a \tag{14}$$

The master and slave systems are chosen as

$$u = -g(y) - G(y)\hat{b} + f(x) - F(x)\hat{a} - ke - \eta \text{sgn}(s) - \rho s \tag{15}$$

where k, η, ρ are positive gain values, $\hat{a}, \hat{b}$ are parameter estimates of the master and slave systems respectively, and s is the sliding surface. Using (15) in (14), the error dynamics simplifies to

$$D^{q_i} e = G(y)\left[b - \hat{b}\right] - F(x)\left[a - \hat{a}\right] - \eta \text{sgn}(s) - \rho s \tag{16}$$

$$V = \frac{1}{2}s^2 + \frac{1}{2}(b - \hat{b})^2 + \frac{1}{2}(a - \hat{a})^2 \tag{17}$$

$$\dot{V} = s.\dot{s} + (b - \hat{b})(-\dot{\hat{b}}) + (a - \hat{a})(-\dot{\hat{a}}) \tag{18}$$

By definition of fractional calculus

$$\dot{x}(t) = D_t^{1-q} \cdot D_t^q x(t) \tag{19}$$

Using (19) in (18),

$$\dot{V} = s.D_t^{1-q} \cdot D_t^q s - (b - \hat{b})(D_t^{1-q} \cdot D_t^q \hat{b}) - (a - \hat{a})(D_t^{1-q} \cdot D_t^q \hat{a}) \tag{20}$$

Solving (20) to obtain the sign of the Lyapunov first derivate is complex and hence we use the modified FO Lyapunov method [36] as

$$\frac{1}{2}D_t^q x^2(t) \leq x(t)\frac{1}{2}D_t^q x(t), q \in (0, 1) \tag{21}$$

Using (21), (13), (14) in (20),

$$\dot{V} \leq ks\left[G(y)(b - \hat{b}) - F(x)(a - \hat{a}) - \eta \text{sgn}(s) - \rho s\right] - (b - \hat{b})(D^q \hat{b}) - (a - \hat{a})(D^q \hat{a}) \tag{22}$$

In order to make (22) a negative definite, we have to cancel terms $G(y)(b - \hat{b}), F(x)$ $(a - \hat{a})$ against $(b - \hat{b})(D^q \hat{b}), (a - \hat{a})(D^q \hat{a})$and hence we define the parameter estimate laws as

$$\begin{aligned} D^q \hat{b} &= K_a s.G(y) \\ D^q \hat{a} &= K_b s.F(x) \end{aligned} \tag{23}$$

where K_a, K_b are positive constants. Using (23) in (22) the Lyapunov function dynamics is defined as

$$\dot{V} \leq -\eta |s| - \rho s^2 \tag{24}$$

as η and ρ are all positive, $\dot{V}$ is negative definite.

3.1 Synchronization of FOMNN systems

We use the FOMNN systems as master and slave for numerical validation of synchronization. Let us define the FOMNN master system as

$$\begin{aligned} D^{q_{x_m}} x_m &= -x_m + 1.6\tanh(x_m) + 2\tanh(y_m) + \tanh(z_m) \\ D^{q_{y_m}} y_m &= -y_m + (M(w_m))\tanh(x_m) + 1.5\tanh(y_m) \\ D^{q_{z_m}} z_m &= -z_m + 3\tanh(x_m) - 2\tanh(y_m) + \tanh(z_m) + c \\ D^{q_{w_m}} w_m &= \tanh(x_m) \end{aligned} \tag{25}$$

where $M(w_m) = aw_m + bw_m^2$ is the memristor element of the master FOMNN system. The FOMNN slave system with the adaptive SMCs (u_i) is defined as

$$\begin{aligned} D^{q_{x_s}} x_s &= -x_s + 1.6\tanh(x_s) + 2\tanh(y_s) + \tanh(z_s) + u_x \\ D^{q_{y_s}} y_s &= -y_s + (M(w_s))\tanh(x_s) + 1.5\tanh(y_s) + u_y \\ D^{q_{z_s}} z_s &= -z_s + 3\tanh(x_s) - 2\tanh(y_s) + \tanh(z_s) + \hat{c} + u_z \\ D^{q_{w_s}} w_s &= \tanh(x_s) + u_w \end{aligned} \tag{26}$$

where $M(w_m) = \hat{a}w_m + \hat{b}w_m^2$ is the memristor of the slave FOMNN system. The parameters of the slave system are assumed to be unknown with parameter estimates $\hat{a}, \hat{b}, \hat{c}$. Using master system (25) and slave system (26) the error dynamics can be derived as

$$\begin{aligned} D^{q_x} e_x &= \hat{a}(y_s - x_s) + w_s - a(y_m - x_m) - w_m + u_x \\ D^{q_y} e_y &= \hat{c}y_s - x_s z_s - cy_m + x_m z_m + u_y \\ D^{q_z} e_z &= x_s y_s - \hat{b}z_s + y_s - x_m y_m + bz_m - y_m + u_z \\ D^{q_w} e_w &= x_s z_s + \hat{r}w_s - x_m z_m - rw_m + u_w \end{aligned} \tag{27}$$

Let the adaptive SMCs be chosen as

$$\begin{aligned} u_x &= e_x - 1.6f(x) - 2f(y) - f(z) - \gamma_x \mathrm{sgn}(s_x) - \rho_x s_x - k_x e_x \\ u_y &= \begin{pmatrix} e_y - 1.5f(y) + a(w_m\tanh(x_m) - \\ w_s\tanh(x_s)) + b(w_m^2\tanh(x_m) - w_s^2\tanh(x_s)) \end{pmatrix} - \gamma_y \mathrm{sgn}(s_y) - \rho_y s_y - k_y e_y \\ u_z &= e_z - 3f(x) + 2f(y) - f(z) - \gamma_z \mathrm{sgn}(s_z) - \rho_z s_z - k_z e_z \\ u_w &= -f(x) - \gamma_w \mathrm{sgn}(s_w) - \rho_w s_w - k_w e_w \end{aligned} \tag{28}$$

where $\rho_i > 0, \gamma_i > 0$ are the sliding surface gains and k_i is the controller gain for $i = x, y, z, w$. Using (27) and (28) with (23), the parameter update laws can be defined as

$$\begin{aligned} D^{q_y}\hat{a} &= -s_y w_s \tanh(x_s) \\ D^{q_y}\hat{b} &= -s_y w_s^2 \tanh(x_s) \\ D^{q_z}\hat{c} &= -s_z \end{aligned} \tag{29}$$

where $D^{q_y}\hat{a}, D^{q_y}\hat{b}, D^{q_z}\hat{c}$ are the dynamics of the parameter estimates $\hat{a}, \hat{b}, \hat{c}$. For numerical simulations, we take the ICs of the master (25) as $[-0.1, 0.1, 0.2, 0.3]$, slave as $[0.4, 0.4, 0.4, 0.4]$, parameter estimates as $\hat{a}(0) = 0$, $\hat{b}(0) = 0.01$, $\hat{c}(0) = 0.01$, the proportional constants as $K = [1, 1, 1, 1]$. The FOs of the master and slave are taken as $q_x = 0.995, q_y = 0.997, q_z = 0.998, q_w = 0.999$. Figure 4 shows the dynamics of the synchronization errors and Figure 5 shows the estimated parameters of the slave system using the updated parameter law (29).

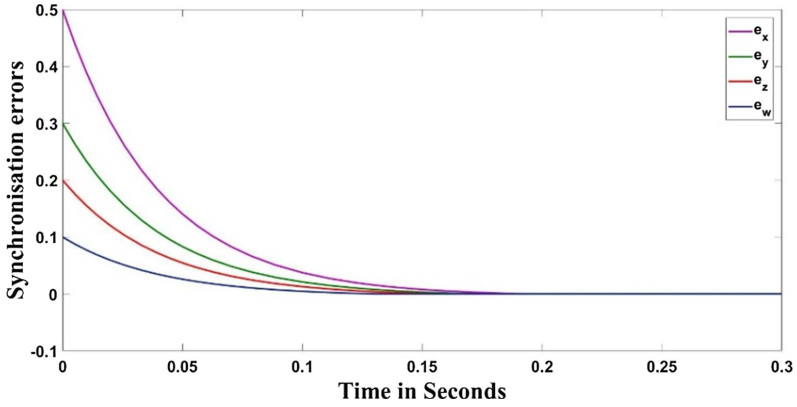

Fig. 4. Dynamics of synchronization errors.

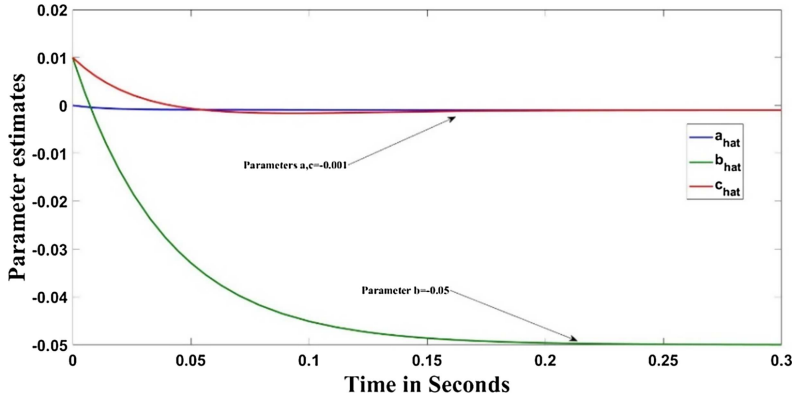

Fig. 5. Estimated parameters of slave system.

4 Implementation of novel chaotic MNN on FPGA

In recent years, chaotic generators are performed with different types of digital platforms such as microprocessor and FPGA. As FPGA chips have parallel signal processing and high speed properties, FPGAs are commonly employed in the applications that require high performance and processing power [31,53–55]. Hence, the study of FPGA-based modeling of chaotic systems has become very important in the literature [56–62]. Rajagopal et al. studied the chaotic chameleon attractors has been realized using VHDL [63]. Alçın et al. designed the PU system in VHDL with artificial neural network (ANN) on FPGA [64]. Tuna et al. realized their novel system with some algorithms on FPGA [65]. Koyuncu et al. realized their newly proposed PW system using Xilinx ISE on FPGA kit [66]. Azzaz et al. performed a three-dimensional system on a Virtex II board using a fixed-point format [67]. Lai et al. successfully realized a multi-butterfly system on a FPGA [68]. Koyuncu et al. implemented Burke–Shaw system on Xilinx Virtex 6 using a fifth-order Runge–Kutta–Butcher (RK5-Butcher) [69]. Rajagopal et al. realized memristor-based chaotic and hyperchaotic systems on FPGA using the Adomian decomposition method [70]. Sadoudi et al. performed the Chen system using FPGA for secure chaotic communications [71]. Akgül et al. realized the design of a system with RK4 on FPGA [72]. In Table 2, the FPGA-based chaotic generators realized, their properties, the algorithms used, and the properties of the FPGA chip used in recent years studies are given.

Table 2. Properties of some FPGA-based chaotic oscillators realized in the literature.

Authors	Chaotic attractor	Algorithm	Numeric standard	FPGA properties
Rajagopal et al. [63]	A novel 3D chaotic Chameleon system	RK5-Butcher	32-bit Floating point	Xilinx XC6VLX240T
Alçın et al. [64]	PU chaotic System	ANN	32-bit Floating point	Xilinx-XC6VLX240T
Tuna et al. [65]	A novel chaotic system	Heun	32-bit Floating point	Xilinx-XC6VLX75T
Koyuncu et al. [66]	Pehlivan-Wei chaotic system	Euler, Heun and RK4	32-bit Floating point	Xilinx-XC6VLX240T
Azzaz et al. [67]	3D Hybrid chaotic system	Euler	32-bit Fixed-point	Xilinx-Virtex II
Lai et al. [68]	Multi-butterfly chaotic systems	RK4	32-bit Floating point	Xilinx-Kintex 7
Koyuncu et al. [69]	Burke-Shaw	RK5-Butcher	32-Bit Floating point	Xilinx-XC6VLX75T
Rajagopal et al. [70]	Memristor-based hyperchaotic sys.	Adomian decomposition method	32-bit Floating point	Xilinx Kintex 7 7k160t
Tlelo-Cuautle et al. [73]	50-scroll chaotic attractor	–	32-bit Fixed-point	Altera Cyclone IV
Sadoudi et al. [71]	Chen chaotic system	RK4	32-bit Floating point	Xilinx Virtex II
Akgül et al. [72]	3-D chaotic system	RK-4	32 bit Floating point	Xilinx Virtex 6
Rajagopal et al. [74]	3D time-delayed Chameleon chaotic sys.	MATLAB Xilinx system generator	–	Xilinx Virtex 7-XC7-VX980tffg
Tuna et al. [75]	Lü-Chen	Heun	32-bit Fixed-point	Xilinx Virtex 6
Alçın et al. [76]	PU chaotic System	ANN	32-bit Floating point	Xilinx Virtex 6
Karakaya et al. [77]	Chua chaotic system	Euler	32-bit Fixed-point	Xilinx Kintex-7

In this study, the fourth-order Runge–Kutta (RK4) numerical algorithm is used to model the chaotic MNN oscillator on FPGA numerically. The RK4 algorithm produces results better than those of classic RK algorithms and the error margin is quite lower in the RK4 algorithm.

$$\begin{aligned} y_{k+1} &= y_k + \frac{1}{6}(k_1 + 2\cdot k_2 + 2\cdot k_3 + k_4) \\ k_1 &= h\cdot f(x_k, y_k) \\ k_2 &= h\cdot f\left(x_k + \frac{h}{2}, y_k + \frac{k_1}{2}\right) \\ k_3 &= h\cdot f\left(x_k + \frac{h}{2}, y_k + \frac{k_2}{2}\right) \\ k_4 &= h\cdot f(x_k + h, y_k + k_3) \end{aligned} \tag{30}$$

The RK4 algorithm is given in equation (30). Δh is the step size of algorithm. In order to calculate y_{k+1} value, the values of k_1, k_2, k_3, and k_4 must be found as the first operation. Here, k_1 is the initial gradient value at the end of Δh, k_2 is the gradient calculated with the value of k_1 at the midpoint of Δh, k_3 is the gradient calculated with the value of k_2 at the midpoint of Δh and k_4 is the gradient calculated with the value of k_3 at the midpoint of Δh. In this way, the value of y_{k+1} is calculated with y_λ and Δh [78].

$$\begin{aligned} \dot{x} &= f(t,x,y,z,w) = -x + 1.6\cdot\tanh(x) + 2\times\tanh(y) + \tanh(z) \\ \dot{y} &= g(t,x,y,z,w) = -y + M(w)\cdot\tanh(x) + 1.5\cdot\tanh(y) \\ \dot{z} &= \rho(t,x,y,z,w) = -z + 3\cdot\tanh(x) - 2\cdot\tanh(y) + \tanh(z) + c \\ \dot{w} &= \sigma(t,x,y,z,w) = \tanh(x) \\ M(w) &= a\cdot w + b\cdot w^2 \end{aligned} \tag{31}$$

$$
\begin{aligned}
x(k+1) &= x(k) + \frac{1}{6} \cdot \Delta h \cdot [\kappa_1(k) + 2 \cdot \kappa_2(k) + 2 \cdot \kappa_3(k) + \kappa_4(k)] \\
y(k+1) &= y(k) + \frac{1}{6} \cdot \Delta h \cdot [\lambda_1(k) + 2 \cdot \lambda_2(k) + 2 \cdot \lambda_3(k) + \lambda_4(k)] \\
z(k+1) &= z(k) + \frac{1}{6} \cdot \Delta h \cdot [\xi_1(k) + 2 \cdot \xi_2(k) + 2 \cdot \xi_3(k) + \xi_4(k)] \\
w(k+1) &= w(k) + \frac{1}{6} \cdot \Delta h \cdot [\psi_1(k) + 2 \cdot \psi_2(k) + 2 \cdot \psi_3(k) + \psi_4(k)]
\end{aligned} \tag{32}
$$

The model of the discretized novel MNN oscillator with respect to functions f, g, δ, and σ in equation (31) is given in equation (32). In equation (33), parameters K_1, K_2, K_3, and K_4 are the coefficients of the MNN oscillator's first equation, λ_1, λ_2, λ_3, and λ_4 are the coefficients of the second equation, ξ_1, ξ_2, ξ_3, and ξ_4 are the coefficients of the third equation and ψ_1, ψ_2, ψ_3, and ψ_4 are the coefficients of the fourth equation. The coefficients in equation (32) are calculated. Then by substituting them in the RK4 algorithm, the next values of the MNN oscillator $x_{(k+1)}$, $y_{(k+1)}$, $z_{(k+1)}$, and $w_{(k+1)}$ are calculated with the step size Δh. The outputs of the every iteration ($x_{(k+1)}$, $y_{(k+1)}$, $z_{(k+1)}$, and $w_{(k+1)}$) are used as the ICs in the next iteration. In the modeling of the new chaotic MNN, Δh value is taken as 0.005. In Figure 6, the top-level schema of the MNN unit realized on FPGA with VHDL are given:

$$
\begin{aligned}
\kappa_1 &= f(x(k), y(k), z(k), w(k)) \\
\lambda_1 &= g(x(k), y(k), z(k), w(k)) \\
\xi_1 &= \delta(x(k), y(k), z(k), w(k)) \\
\psi_1 &= \sigma(x(k), y(k), z(k), w(k)) \\
\kappa_2 &= f(x(k) + \frac{1}{2}\Delta h \cdot \kappa_1, y(k) + \frac{1}{2}\Delta h \cdot \lambda_1, z(k) + \frac{1}{2}\Delta h \cdot \xi_1, w(k) + \frac{1}{2}\Delta h \cdot \psi_1) \\
\lambda_2 &= g(x(k) + \frac{1}{2}\Delta h \cdot \kappa_1, y(k) + \frac{1}{2}\Delta h \cdot \lambda_1, z(k) + \frac{1}{2}\Delta h \cdot \xi_1, w(k) + \frac{1}{2}\Delta h \cdot \psi_1) \\
\xi_2 &= \delta(x(k) + \frac{1}{2}\Delta h \cdot \kappa_1, y(k) + \frac{1}{2}\Delta h \cdot \lambda_1, z(k) + \frac{1}{2}\Delta h \cdot \xi_1, w(k) + \frac{1}{2}\Delta h \cdot \psi_1) \\
\psi_2 &= \sigma(x(k) + \frac{1}{2}\Delta h \cdot \kappa_1, y(k) + \frac{1}{2}\Delta h \cdot \lambda_1, z(k) + \frac{1}{2}\Delta h \cdot \xi_1, w(k) + \frac{1}{2}\Delta h \cdot \psi_1) \\
\kappa_3 &= f(x(k) + \frac{1}{2}\Delta h \cdot \kappa_2, y(k) + \frac{1}{2}\Delta h \cdot \lambda_2, z(k) + \frac{1}{2}\Delta h \cdot \xi_2, w(k) + \frac{1}{2}\Delta h \cdot \psi_2) \\
\lambda_3 &= g(x(k) + \frac{1}{2}\Delta h \cdot \kappa_2, y(k) + \frac{1}{2}\Delta h \cdot \lambda_2, z(k) + \frac{1}{2}\Delta h \cdot \xi_2, w(k) + \frac{1}{2}\Delta h \cdot \psi_2) \\
\xi_3 &= \delta(x(k) + \frac{1}{2}\Delta h \cdot \kappa_2, y(k) + \frac{1}{2}\Delta h \cdot \lambda_2, z(k) + \frac{1}{2}\Delta h \cdot \xi_2, w(k) + \frac{1}{2}\Delta h \cdot \psi_2) \\
\psi_3 &= \sigma(x(k) + \frac{1}{2}\Delta h \cdot \kappa_2, y(k) + \frac{1}{2}\Delta h \cdot \lambda_2, z(k) + \frac{1}{2}\Delta h \cdot \xi_2, w(k) + \frac{1}{2}\Delta h \cdot \psi_2) \\
\kappa_4 &= f(x(k) + \Delta h \cdot \kappa_3, y(k) + \Delta h \cdot \lambda_3, z(k) + \Delta h \cdot \xi_3, w(k) + \Delta h \cdot \psi_3) \\
\lambda_4 &= g(x(k) + \Delta h \cdot \kappa_3, y(k) + \Delta h \cdot \lambda_3, z(k) + \Delta h \cdot \xi_3, w(k) + \Delta h \cdot \psi_3) \\
\xi_4 &= \delta(x(k) + \Delta h \cdot \kappa_3, y(k) + \Delta h \cdot \lambda_3, z(k) + \Delta h \cdot \xi_3, w(k) + \Delta h \cdot \psi_3) \\
\psi_4 &= \sigma(x(k) + \Delta h \cdot \kappa_3, y(k) + \Delta h \cdot \lambda_3, z(k) + \Delta h \cdot \xi_3, w(k) + \Delta h \cdot \psi_3)
\end{aligned} \tag{33}
$$

In Figure 7, the second-level structure of the MNN unit is given. The chaotic signal generator given in Figure 7 is composed of three blocks, namely Multiplexer (MUX) unit, Novel_Chaotic_MNN_Oscillator unit, and Filter unit. The MUX block is used to satisfy the initial values of the system while the Novel_Chaotic_MNN_Oscillator block generates the chaotic signals. The filter is designed to block undesirable signals

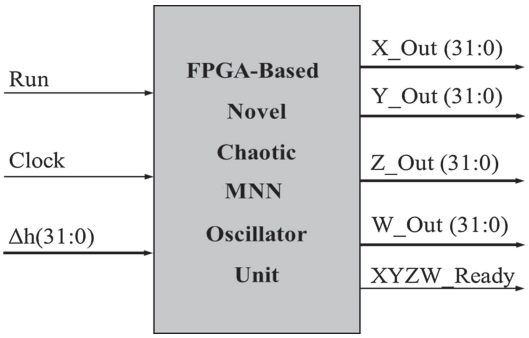

Fig. 6. Top-level schema of novel MNN system on FPGA.

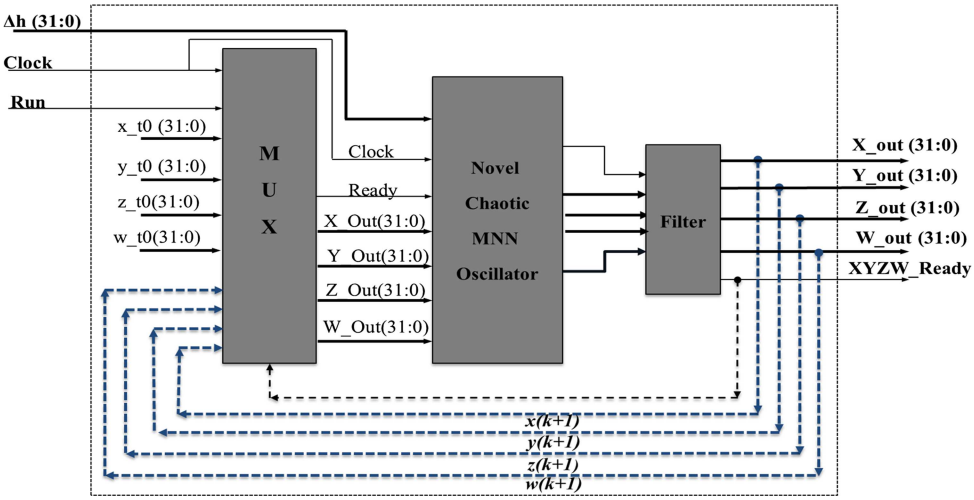

Fig. 7. Second-level block schema of MNN system on FPGA.

when the chaotic oscillator does not produce any results. When the MNN oscillator unit receives the Run signal, the ICs become the predefined values in the oscillator. As the designed unit generates the first results, XYZW_Ready becomes "1" and the outputs of the unit become the ICs of the unit in the next iteration.

In the designed system, there are four output signals in IEEE-754-1985 standard as X_out, Y_out, Z_out, and W_out, and there is a control signal XYZW_Ready. The outputs of the Filter block are both the outputs of the MNN oscillator system and the ICs, $x_{(k+1)}$, $y_{(k+1)}$, $z_{(k+1)}$, and $w_{(k+1)}$, that are sent to the MUX block for calculations of the next iteration.

In Figure 8, the third-level schema of the MNN unit that is designed using RK4 numerical algorithm is given. There are six parts in the MNN unit: MUX, k_1, k_2, k_3, k_4, and ys. The $k\varsigma$ calculates the values of K_ς, ξ_ς, λ_ς, and ψ_ς given in the model of the system for $\varsigma = 1, \ldots, 4$. The values of $x_{(k+1)}$, $y_{(k+1)}$, $z_{(k+1)}$, and $w_{(k+1)}$ given in equation (31) are calculated in ys.

All designed, multiplication, subtraction and other modules are created with Xilinx IP Core Generator. The MNN unit is run as a pipeline and the chaotic oscillator generates the first result after 182 clock pulses. After this, new results are generated every 182 clock pulses. The realized chaos-based generator is tested on Xilinx XC6VLX240T FPGA chip. In Figure 9, the simulation results of the realized new MNN oscillator are given.

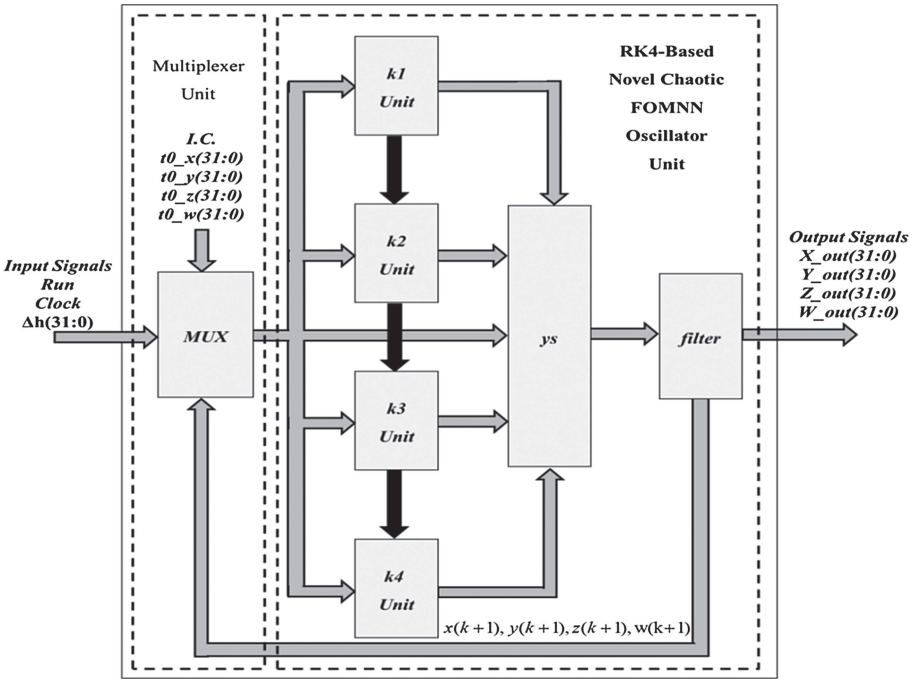

Fig. 8. Third-level schema of MNN system.

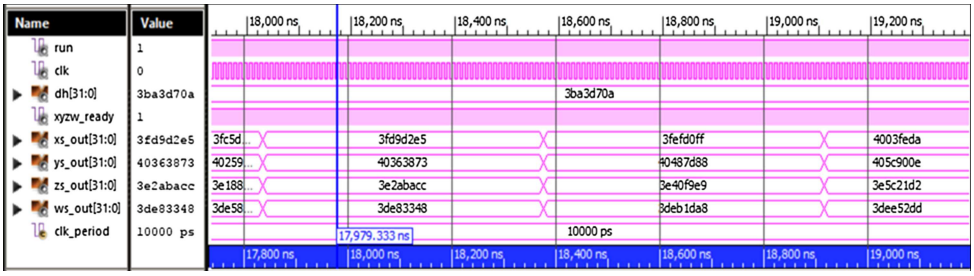

Fig. 9. Xilinx ISE Design Tool Simulator results for MNN system on FPGA.

4.1 FPGA-based modeling results of the novel chaotic MNN oscillator

In this study, discrete time design of the novel MNN oscillator is realized. In order to realize the newly designed MNN system with the RK4 algorithm, chip statistics are obtained. The operation frequency of the MNN oscillator designed on FPGA with VHDL is 231.616 MHz. In Table 3, the chip statistics are given. The minimum operating period of the MNN oscillator is 4.318 ns.

Finally, the output signals of the new MNN oscillator are saved on a file during the testing stage. Then, these signals are converted to a real number to obtain the time series of the chaotic oscillator. To plot the phase portraits of MNN, the first-generated 4×1500 dataset is used. The phase portraits of the discrete time MNN unit are given in Figure 10.

Table 3. XC6VLX240T-1-FF1156 chip statistics for chaotic MNN oscillator designed with RK4 algorithm.

Logic	Used	Available	Utilization (%)
Slice registers	109734	301440	36
LUT-FF pairs	87190	140082	62
Slice LUTs	117538	150720	77
IOBs	163	600	27
DSP48E	12	288	4
BUFG/BUFGCTRLs	1	32	3

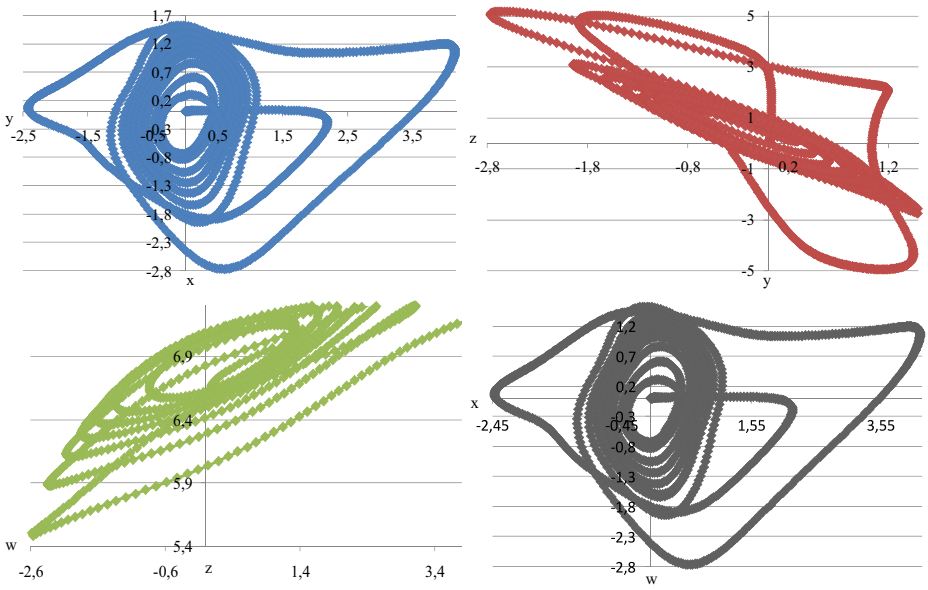

Fig. 10. Phase portraits of new MNN unit on FPGA.

5 Conclusion

This paper investigated the FO-HNN with memristor synaptic weight. The proposed system has no equilibrium points and hence shows hidden oscillations. Bifurcation of the system with parameters and FOs is performed and the corresponding LEs are also presented to show the existence of chaos. An adaptive sliding mode controller is designed to synchronize the identical FOMNN systems with uncertainties. The newly designed MNN oscillator unit is operated with RK4 numerical on FPGA. The maximum operating frequency of the new MNN oscillator unit is 231.616 MHz for Xilinx XC6VLX240T-1-FF1156 FPGA. The successful results obtained from the new chaotic MNN oscillator introduced to the literature show that the memristor-based structure can be used in engineering applications like cryptology. In future studies, applications like data masking, random number generators, etc. can be performed with the FPGA-based discrete time new chaotic MNN oscillator.

References

1. L. Chua, IEEE Trans. Circuit Theory **18**, 507 (1971)
2. L.O. Chua, Sung Mo Kang, Proc. IEEE **64**, 209 (1976)

3. D.B. Strukov, G.S. Snider, D.R. Stewart, R.S. Williams, Nature **453**, 80 (2008)
4. S. Shin, K. Kim, S.-M. Kang, IEEE Trans. Nanotechnol. **10**, 266 (2011)
5. Y.V. Pershin, M. Di Ventra, Neural Networks **23**, 881 (2010)
6. J.L. Hindmarsh, R.M. Rose, Nature **296**, 162 (1982)
7. H. Qin, J. Ma, W. Jin, C. Wang, Sci. China Technol. Sci. **57**, 936 (2014)
8. X.-S. Yang, Y. Huang, Chaos Interdiscip. J. Nonlinear Sci. **16**, 033114 (2006)
9. X.-S. Yang, Q. Li, Int. J. Bifurc. Chaos **16**, 157 (2006)
10. J.E. Lewis, L. Glass, Int. J. Bifurc. Chaos **01**, 477 (1991)
11. J.E. Lewis, L. Glass, Neural Comput. **4**, 621 (1992)
12. R. Edwards, Phys. D Nonlinear Phenom. **146**, 165 (2000)
13. R. Edwards, L. Glass, Chaos Interdiscip. J. Nonlinear Sci. **10**, 691 (2000)
14. T. Mestl, C. Lemay, L. Glass, Phys. D Nonlinear Phenom. **98**, 33 (1996)
15. T. Mestl, E. Plahte, S.W. Omholt, Dyn. Stab. Syst. **10**, 179 (1995)
16. A. Das, P. Das, A.B. Roy, Int. J. Bifurc. Chaos **12**, 2271 (2002)
17. X.-S. Yang, Q. Yuan, Neurocomputing **69**, 232 (2005)
18. V.T. Pham, S. Jafari, S. Vaidyanathan, C. Volos, X. Wang, Sci. China Technol. Sci. **59**, 358 (2016)
19. X. Sun, X. Shi, Sci. China Technol. Sci. **57**, 879 (2014)
20. D. Wang, H.i Zhao, J. Yu, in *2009 International Conferences on Communications Circuits and Systems* (IEEE, 2009), pp. 958–960
21. H. Wang, Y. Yu, G. Wen, S. Zhang, J. Yu, Neurocomputing **154**, 15 (2015).
22. E. Kaslik, S. Sivasundaram, in *2011 International Joint Conference on Neural Networks* (IEEE, 2011), pp. 611–618
23. E. Kaslik, S. Sivasundaram, Neural Networks **32**, 245 (2012)
24. A. Boroomand, M.B. Menhaj, *Fractional-Order Hopfield Neural Networks* (Springer, Berlin, Heidelberg, 2009), pp. 883–890
25. C. Song, J. Cao, Neurocomputing **142**, 494 (2014)
26. A. Wu, Z. Zeng, X. Song, Neurocomputing **177**, 489 (2016)
27. H. Wu, L. Wang, Y. Wang, P. Niu, B. Fang, Adv. Differ. Equations **2016**, 132 (2016)
28. P. Liu, Z. Zeng, J. Wang, IEEE Trans. Syst. Man, Cybern. Syst. **47**, 2279 (2017)
29. R. Rakkiyappan, J. Cao, G. Velmurugan, IEEE Trans. Neural Networks Learn. Syst. **26**, 84 (2015)
30. G. Velmurugan, R. Rakkiyappan, V. Vembarasan, J. Cao, A. Alsaedi, Neural Networks **86**, 42 (2017)
31. M.F. Tolba, A.M. AbdelAty, N.S. Soliman, L.A. Said, A.H. Madian, A.T. Azar, A.G. Radwan, AEU – Int. J. Electron. Commun. **78**, 162 (2017)
32. I. Petráš, *Fractional-Order Nonlinear Systems?: Modeling, Analysis and Simulation* (Higher Education Press, 2011)
33. A. Wolf, J.B. Swift, H.L. Swinney, J.A. Vastano, Phys. D Nonlinear Phenom. **16**, 285 (1985)
34. M.-F. Danca, Nonlinear Dyn. **81**, 227 (2015)
35. J.-J.E. Slotine, W. Li, *Applied Nonlinear Control* (Prentice Hall, 1991)
36. K. Rajagopal, S. Vaidyanathan, A. Karthikeyan, P. Duraisamy, Electr. Eng. **99**, 721 (2017)
37. B.A. Idowu, U.E. Vincent, A.N. Njah, Chaos Solitons Fractals **39**, 2322 (2009)
38. S. Vaidyanathan, K. Rajagopal, Int. J. Signal Syst. Control Eng. Appl. **4**, 55 (2011)
39. V. Sundarapan, R. Karthikeya, Int. J. Soft Comput. **6**, 111 (2011)
40. V. Sundarapan, R. Karthikeya, J. Eng. Appl. Sci. **7**, 45 (2012)
41. S.S. Majidabad, H.T. Shandiz, J. Control Syst. Eng. **1**, 1 (2013)
42. S. Vaidyanathan, Arch. Control Sci. **27**, 409 (2017)
43. O.S. Onma, O.I. Olusola, A.N. Njah, J. Nonlinear Dyn. **2014**, 1 (2014)
44. B. Wang, Y. Li, D.L. Zhu, Int. J. Control Autom. **8**, 425 (2015)
45. C. Yin, S. Dadras, S. Zhong, Y. Chen, Appl. Math. Model. **37**, 2469 (2013)
46. H. Liu, J. Yang, Entropy **17**, 4202 (2015)
47. S. Wang, Y. Yu, M. Diao, Phys. A Stat. Mech. Appl. **389**, 4981 (2010)

48. K. Rajagopal, L. Guessas, A. Karthikeyan, A. Srinivasan, G. Adam, Complexity **2017**, 1 (2017)
49. K. Rajagopal, A. Karthikeyan, A.K. Srinivasan, Nonlinear Dyn. **87**, 2281 (2017)
50. P. Muthukumar, P. Balasubramaniam, K. Ratnavelu, Int. J. Dyn. Control **5**, 115 (2017)
51. X. Song, S. Song, I.T. Balsera, L. Liu, L. Zhang, J. Control Sci. Eng. **2017**, 1 (2017)
52. Y. Toopchi, J. Wang, Entropy **16**, 6539 (2014)
53. I. Koyuncu, Adv. Electr. Comput. Eng. **18**, 79 (2018)
54. I. Koyuncu, M. Tuna, M. Alçın, in *International Eurasian Conference on Science Engineering Technology (EurasianSciEnTech 2018), November 22–23, 2018* (Ankara, Turkey, 2018), pp. 2532–2541
55. A. Senouci, H. Bouhedjeur, K. Tourche, A. Boukabou, AEU – Int. J. Electron. Commun. **82**, 211 (2017)
56. İ. Koyuncu, A. Turan Özcerit, Comput. Electr. Eng. **58**, 203 (2017)
57. İ. Koyuncu, İ. Şahin, C. Gloster, N.K. Sarıtekin, J. Circuits, Syst. Comput. **26**, 1750015 (2017)
58. Ü. Çavuşoğlu, A. Akgül, S. Kaçar, İ. Pehlivan, A. Zengin, Secur. Commun. Networks **9**, 1285 (2016)
59. M. Tuna, C.B. Fidan, İ. Koyuncu, *The Chaos-Based Dual Entropy Core TRNG On FPGA: VHDL CODES of Chaotic Systems* (LAMBERT Academic Publication (LAP), 2019)
60. K. Rajagopal, A. Karthikeyan, P. Duraisamy, Complexity **2017**, 1 (2017)
61. I. Koyuncu, Int. J. Intell. Syst. Appl. Eng. **4**, 33 (2016)
62. İ. Koyuncu, H. İ. Şeker, Sak. Univ. J. Sci. **23**, 859 (2019)
63. K. Rajagopal, A. Akgul, S. Jafari, A. Karthikeyan, I. Koyuncu, Chaos Solitons Fractals **103**, 476 (2017)
64. M. Alçın, İ. Pehlivan, İ. Koyuncu, Opt. – Int. J. Light Electron Opt. **127**, 5500 (2016)
65. M. Tuna, C.B. Fidan, Opt. – Int. J. Light Electron Opt. **127**, 11786 (2016)
66. I. Koyuncu, A.T. Ozcerit, I. Pehlivan, Nonlinear Dyn. **77**, 49 (2014)
67. M.S. Azzaz, C. Tanougast, S. Sadoudi, R. Fellah, A. Dandache, Commun. Nonlinear Sci. Numer. Simul. **18**, 1792 (2013)
68. Q. Lai, X.-W. Zhao, K. Rajagopal, G. Xu, A. Akgul, E. Guleryuz, Pramana **90**, 6 (2018)
69. I. Koyuncu, A.T. Ozcerit, I. Pehlivan, Optoelectron. Adv. Mater. Rapıd Commun. **7**, 635 (2013)
70. K. Rajagopal, A. Karthikeyan, A. Srinivasan, Nonlinear Dyn. **91**, 1491 (2018)
71. S. Sadoudi, M.S. Azzaz, M. Djeddou, M. Benssalah, Int. J. Nonlinear Sci. **7**, 1749 (2009)
72. A. Akgul, H. Calgan, I. Koyuncu, I. Pehlivan, A. Istanbullu, Nonlinear Dyn. **84**, 481 (2015)
73. E. Tlelo-Cuautle, A.D. Pano-Azucena, J.J. Rangel-Magdaleno, V. H. Carbajal-Gomez, G. Rodriguez-Gomez, Nonlinear Dyn. **85**, 2143 (2016)
74. K. Rajagopal, S. Jafari, G. Laarem, Pramana **89**, 92 (2017)
75. M. Tuna, M. Alçın, İ. Koyuncu, C.B. Fidan, İ. Pehlivan, Microprocess. Microsyst. **66**, 72 (2019)
76. M. Alcin, I. Koyuncu, M. Tuna, M. Varan, I. Pehlivan, Int. J. Circuit Theory Appl. **47**, 365 (2019)
77. B. Karakaya, A. Gülten, M. Frasca, Chaos Solitons Fractals **119**, 143 (2019)
78. J.C. Butcher, *Numerical Methods for Ordinary Differential Equations* (J. Wiley, 2008)