HARDWARE IMPLEMENTATION OF DYNAMICAL NEURAL NETWORKS SUITABLE FOR ONLINE TRAINING

V. V. Khilkevichv,
Moscow Power Engineering Institute

Abstract — **The report is devoted to the hardware implementation of dynamical neural network using Field Programmable Gate Arrays. Simulation results are presented.**

Index items — **Dynamical neural networks, neural network training, Field Programmable Gate Arrays.**

I. Introduction

Dynamical neural networks are universal tools which can be used in such areas as system modeling, construction of control systems, signal generation and so on. One of the simplest continuous time dynamical neural networks can be represented by the following general form [1]:

$$\dot{y}_i = \frac{1}{\tau_i}\left(-y_i + \sum_{j=1}^{N} w_{ji}\sigma(y_j + \theta_j) + I_i\right), \quad (1)$$
$$i = 1, \ldots, N,$$

where N — number of neurons, y_i — state of i-th neuron, τ_i — time constant of i-th neuron, $s(x) = 1/(1 + e^{-x})$ — logistic activation function, w_{ij} — weight of the connection between j-th and i-th neurons, θ_i — bias of i-th neuron, I_i — external input of i-th neuron.

It was shown [2] that such neural networks are universal approximators of smooth dynamics, which makes them promising in many applications. Besides, system of the form (1) comprised of more than 2 neurons can exhibit chaotic behavior, presumably being the smallest continuous time dynamical neural network in which chaotic dynamics has been observed [1].

Successful training of dynamical networks can be achieved using genetic (or evolutionary) algorithms [3]. This approach implies that neural networks with different genotypes (representing their weights, biases and time constants) constitute a population which is subject to certain genetic operators such as mutation, crossover and selection. At each iteration for each individual in population a value of fitness has to be calculated. This means that at each iteration the output of each neural network in population $y_i(t), i = 1, \ldots, N,\ t \in [t_1, t_2]$ must be determined and compared with the desired one, which in turn demands the implementation of neural network.

There are several approaches to implementation of dynamical neural networks. System of differential equations (1) can be relatively easily integrated numerically, so the evolution and subsequent functioning of evolved network can be entirely modeled on the computer. However this approach needs considerable processing power both at the stage of the training, and in the actual application of evolved network. If population consists of large number of networks and/or their order is high, time required for training can be unacceptable. Computer modeling of evolved network which often has to work in real time environment is also a hard task.

So (in offline training situation) networks are trained on the computer and after that implemented in the form of analog or hybrid circuits [4].

II. Dynamical neural network implementation

Equation (1) can easily be transformed to the form

$$y_i = \frac{1}{p\tau_i + 1}\left(\sum_{j=1}^{N} w_{ji}\sigma(y_j + \theta_j) + I_i\right), \quad (2)$$
$$i = 1, \ldots, N,$$
$$p = \frac{d}{dt},$$

so the neuron can be build of three basic blocks: a) summing unit with weighted inputs; b) a leaky integrator with an adjustable time constant; c) an amplifier with a logistic transfer characteristic.

Block diagram of 2-neuron neural network, built according to equation (2), is shown at fig. 1.

While this system can be implemented using, for example, operational amplifiers, such circuit may

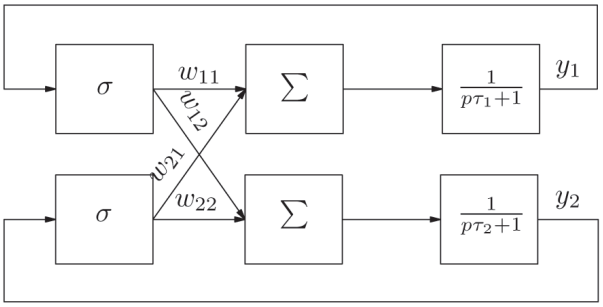

Fig. 1. Dynamical neural network

exhibit different behavior compared to its modeled prototype (especially in the case of large network), due to the fact that mathematical model (1) doesn't directly describe physical processes in the electric circuit [5]. Moreover, modeling evolution on the digital computer can still demands significant amount of time.

Both of these problems can be solved in online training situation, when individuals are represented not by their mathematical models, but by actual hardware circuits, allowing electronic control of adjustable parameters.

One of possible solutions lies in using of Field Programmable Gate Array (FPGA) chips for digital hardware modeling of systems (1). Training may be realized according to the following diagram (fig. 2):

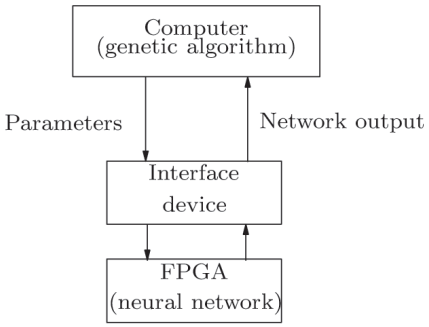

Fig. 2. Neural network training

At each iteration computer calculates genomes of individuals, transfers them sequentially to the chip's memory, receives corresponding network outputs (through specialized interface device) and selects best specimens. This architecture can be very efficient since computer performs relatively simple calculations and FPGA neural network provides samples of its output at the rate of the clock oscillator.

For testing purposes we designed a computer model of 2-neuron FPGA-based neural network using Altera's Quartus II software.

Due to the difficulties of on-chip exponent calculation we used piecewise-linear approximation:

$$\sigma(x) = \begin{cases} 1, & x \geq 2; \\ \dfrac{x+2}{4}, & -2 \leq x \leq 2; \\ 0, & x \leq -2. \end{cases}$$

Analog filters were modeled by 1-st order IIR lowpass digital filters.

State variables were represented by 16-bit signed integers, while IIR filter and summing unit had internal 21-bit registers.

III. Simulation results

Performance of the synthesized circuit were tested using following parameters: $w_{11} = 4.5, w_{12} = -1$, $w_{21} = 1, w_{22} = 4.5, \theta_1 = -2.75, \theta_2 = -1.75, \tau_1 = 1, \tau_2 = 1, I_1 \equiv 0, I_2 \equiv 0$. According to [1] this parameter set provides limit cycle in the phase-space of neural network.

Trajectories of system (1), system (1) with piecewise-linear approximation of nonlinear function $\sigma(x)$, and FPGA model are shown at fig. 3.

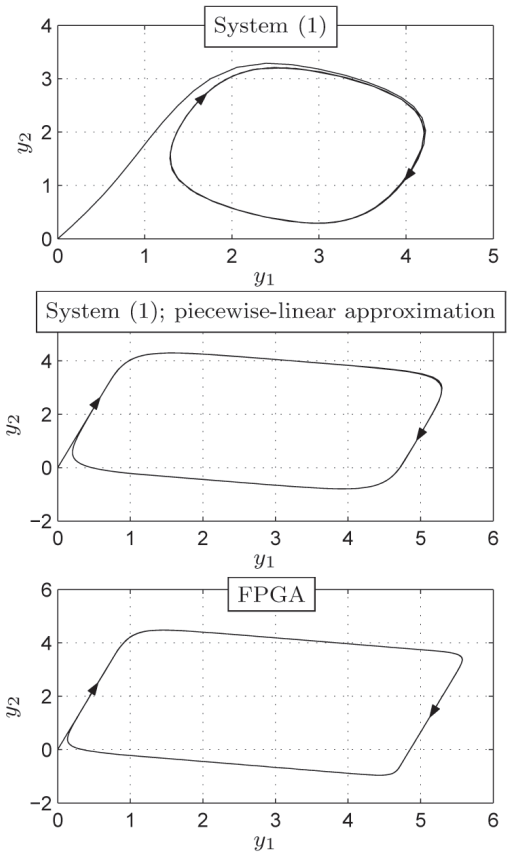

Fig. 3. Phase-space trajectories

According to software simulation, maximum clock frequency of the circuit is approximately

44 Mhz for FLEX10K target devices, and minimum sample calculation time is 45.5 ns (taking into account that filters provide output values at the half rate of clock frequency).

IV. Hardware requirements

Number of hardware resources required for circuit implementation were estimated. Results are summarized in the following table:

Neurons	Logic cells per block				Memory bits
	a)	b)	c)	Overall	
2	596			3046	192
3	892	887	40	5495	336
4	952			7516	512
5	1205			10660	720

V. Conclusions

FPGA-based digital models of continuous time dynamical neural network are able to produce dynamics qualitatively resembling that of their continuous time prototypes. Precision of representation depends on number of data bits and quality of nonlinear function approximation.

Dynamical neural networks with up to 4 neurons and 16-bit data can be implemented using FLEX10K chips (with capacity up to 9,984 logic elements).

REFERENCES

[1] R. Beer. On the dynamics of small continuoustime recurrent neural networks, Adaptive Behavior, vol. 3, no. 4, pp. 469-509, 1995.

[2] K. Funahashi and Y. Nakamura, Approximation of dynamical systems by continuous time recurrent neural networks, Neural Networks, vol. 6, pp. 801-806, 1993.

[3] R. Beer and J. Gallagher, Evolving dynamical neural networks for adaptive behavior," Adaptive Behavior, vol. 1, no. 1, pp. 91-122, 1992.

[4] J. Gallagher and J. Fiore, Continuous time recurrent neural networks: A paradigm for evolvable analog controller circuits," in The Proceedings of the National Aerospace and Electronics Conference. NAECON 2000, pp. 299-304, IEEE Press, 2000.

[5] A. Stoica, C. Lazaro, D. Keymeulen, and K. Hayworth, Evolution of CMOS circuits in simulations and directly in hardware on a programmable chip, in GECCO-99: Proceedings of the Genetic and Evolutionary Computation Conference, vol. 2, pp. 1198-1203, 13-17 July 1999.