Special issue of the International Conference on Computational and Experimental Science and Engineering (ICCESEN 2014)

The Modeling and Hardware Implementation of Semiconductor Circuit Elements by Using ANN and FPGA

R. Tuntas*
Yuzuncu Yil University,
Ercis Technical Vocational School of Higher Education, Electronic and Communication Technologies, Van, Turkey

This study, the modeling and hardware implementation of semiconductor circuit elements very frequently used in electronic circuits are carried out by using artificial neural networks and field programmable gate array chip. Initially the artificial neural network models obtained has been written in very high speed integrated circuit hardware description language (VHDL). Then, these configurations have been simulated and tested under ModelSim Xilinx software. Finally, the best configuration has been implemented under the Xilinx Spartan-3E FPGA (XC3S500E) chip of Xilinx. The modeling of electronic circuit elements is very important both in respect of engineering, and in respect of practical mathematics. The main aim is to shorten the simulation time and to examine the real physical system applications easily by using the model elements instead of using the ones used in real applications. The effectiveness of the implemented artificial neural network models on field programmable gate array was found successful.

DOI: 10.12693/APhysPolA.128.B-78
PACS: 07.05.Mh, 85.30.De, 85.40.Bh

1. Introduction

The field programmable gate array (FPGA)-based approach employs the advantages of both hardware and software models. FPGAs have many consistent advantages such as reliable, flexible, fast response rapid prototyping, adaptation, reduced cost, simplicity of design and programmable architecture [1–3]. FPGA model can be easily reprogrammed and is completely customizable. VHDL is one of the most wide techniques for programming hardware [4]. The usage of VHDL provides a number of advantages, such as speed, high capacity and repeat designs [5, 6]. Metal oxide semiconductor field effect transistor (MOSFET) is commonly used in analogy and digital circuits. MOSFET is a transistor used for amplifying or switching electronic signals. It is the fundamental building block of digital, analogy, and memory circuits. Artificial neural networks (ANNs) have been commonly used in many fields for solving the difficult and complex problems. The most widely used ANN is the multi-layer perceptron (MLP) [7, 8]. The parallel structure of the neural network is very important feature for real-time implementations. FPGA-based application is proper for parallel realizations and is a good solution for the complex problem. In literature, there are various artificial neural network applications on FPGA [9, 10].

In this study, intelligent predictors for MOSFET are proposed and applied into FPGA Xilinx Spartan-3E FPGA (XC3S500E). A MLP has been used for modelling MOSFET. The intelligent ANN-MOSFET model obtained has been written in VHDL. Configurations obtained have been simulated and tested under ModelSim Xilinx software. The best configuration has been implemented under the Xilinx Spartan-3E FPGA (XC3S500E) chip of Xilinx.

The rest of the paper is organized as follows. Section 2 provides a brief overview on the theoretical considerations. Section 3 presents methodology and implementation. Results and discussion are demonstrated in Sect. 4. Finally, Sect. 5 also reports the conclusion.

2. Theoretical considerations

FPGAs are integrated circuits which provide flexibility and reconfiguration advantages for supporting the design and production of digital systems. It can be programmable by the user. An FPGA is a semiconductor electronic device that contains internal components such as gates and multiplexers. These devices utilize the hardware description language (HDL) and VHDL programming language.

The artificial neuron is the most fundamental component of the neural network. The neuron computation methods consist in calculating the pondered sum function and updating the state of the neuron with applying the activation function. Neuron computation methods are carried out in three stages. These stages consist of artificial neuron implementation, data representation and sigmoid activation function approximation. The various steps for the practice of the ANN on FPGA are presented in the following subsection.

The processing of the multiplication and addition of parallel inputs and weights is done in the first stage. The second stage is the nonlinear sigmoid function for the output signal. For the first stage, several types of adders can be used to achieve the weighted sum of states: combinatorial, in series, dynamic, carry look ahead, Manch-

*e-mail: rtuntas@hotmail.com

ester, Wallace tree, etc. [11]. Likewise, several ways can be used to realize the multiplication, the mostly used are: in serial, serial/parallel and completely parallel. The multiplier–accumulator (MAC) is carried out by a multiplier related to an adder looped on it in order to acquire an accumulation.

There is a need of signed real numbers for the computation of neural networks. The fixed-point representation and the floating point representation are used to define the position of the decimal point. The floating point is more dynamic than the fixed point. It allows the encoding of a greater number of real values for the same number of bits [12]. The fixed-point representation is preferred in various applications. Fixed-point arithmetic is used for encoding the parameters and performing the computations [13]. In this study, we have chosen a fixed-point representation of 18 bits (9 bits for integer part and 9 bits for real part). This facilitates the MAC realization which is the main element of the neural network [14].

The direct application of the sigmoid activation function on FPGA is very difficult. It is because the nonlinear sigmoid function has an infinite exponential series. To approximate the sigmoid function with the use of FPGA design, there are three practical approaches consisting of direct approximation, lookup table approximation and piece-wise linear approximation. In this study, we have chosen the direct approximation approach. Direct approximation approach consumes less memory space and has better precision.

3. Methodology and implementation

In this section, a software and hardware design method is described to implement the ANN-MOSFET model on a FPGA chip. This initially an intelligent (ANN–MOSFET) model developed is written in VHDL. In the next step, these configurations are simulated and tested under ModelSim Xilinx software. Then the best configuration is implemented under the Xilinx Spartan-3E FPGA (XC3S500E) chip of Xilinx. Flow diagram of the proposed methodology for developing the ANN–MOSFET model is shown in Fig. 1.

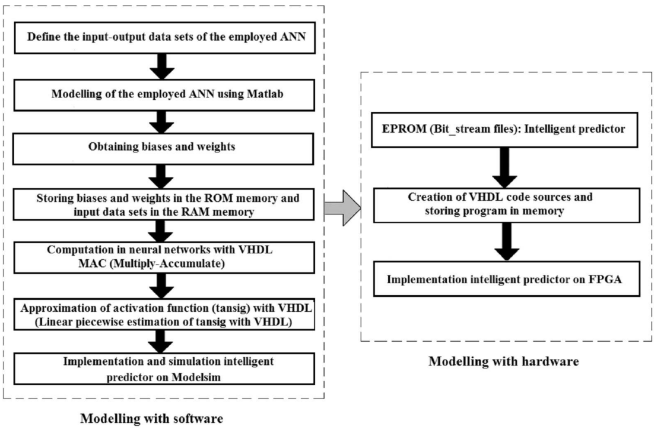

Fig. 1. The flow diagram of the proposed methodology for developing ANN-MOSFET model.

3.1. ANN-based modelling and simulation for MOSFET

A feed-forward ANN architecture is used for identification of the MOSFET. The input–output data sets for ANN model training are obtained based on the characteristic values of the simulink model of the MOSFET circuit. The input data to ANN model for MOSFET are the gate-source voltage (V_{gs}), drain-source voltage (V_{ds}), threshold voltage (V_{th}) and gain constant (K), while the output data are the drain-source current (I_{ds}) of MOSFET. The ANN model of MOSFET is formed by using the software Matlab Ver. 7.9.0.529 (R2009b). For this, the number of hidden layers and the number of neurons within these layers are optimized through the training process. In this study, for the training process, the mean square error (MSE) performance criterion is used. A proper computing program is applied for the prediction of the MOSFET based on the Levenberg–Marquardt (LM) algorithm. To obtain the best performance, the structure of the ANN and training parameters are obtained as shown in Table after several different experiments.

TABLE

ANN architecture and training parameters.

The number of layers	3	activation function on the layers hidden	tangent sigmoid
Neuron on the layers input	4	activation functions on the layers output	linear
Neuron on the layers hidden	5	training parameters	Levenberg–Marquart
Neuron on the layers output	1	sum-squared error	0.000001

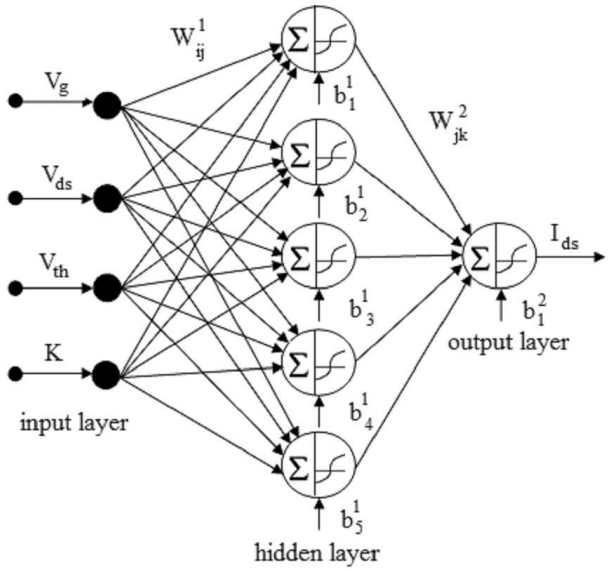

Fig. 2. The neuron architecture of ANN–MOSFET.

A database of 1500 patterns is divided into two parts to constitute the intelligent predictor of the MOSFET.

A dataset of 1200 patterns are used for the learning phase, and other dataset of 300 patterns are used for the testing phase, for the implementation of the intelligent predictor on the FPGA. After the biases and weight values of the ANN were obtained, these values were stored in ROM. In the same manner, previously obtained input data from mathematical model are stored in RAM. The neuron architecture, which is the essential element for application of ANN–MOSFET based on VHDL simulation is shown in Fig. 2.

3.2. FPGA-based implementation for MOSFET

The design of an integrated circuit can be made at four abstraction levels. Short description of these four abstraction levels can be summarized as follows.

First, at behaviour level, suitable VHDL models are improved. Second, at the register transfer level, circuit models are formed. Then, at physical abstraction level, synthesized VHDL models of the predictor structure are performed and tested with simulation. After the latest circuit behaviour is validated, the FPGA is synthesized and configured in a private apparatus. The application of online training algorithms, the obtaining of the activation function and the coefficient values of connection weight are the most important for hardware implementation of neural network (NN). For constructing of the NN, the elementary neuron used in this study is created as follows.

Neuron inputs, previously written to the block RAM, are read from the RAM memory. Likewise, the weights and biases values, previously written to the block ROM, are read from the ROM memory. For the basic neuron computation, firstly, the multiplication result of the neuron input and weight are obtained. In the next step, a bias value is added to them (adder and multiplier). And then, results of these processes are subjected to a unit that approximates an activation function tansig. Finally, activation function generates the neuron output by processing the results of these processes. The basic neuron computation scheme in neural networks based on VHDL is shown in Fig. 3.

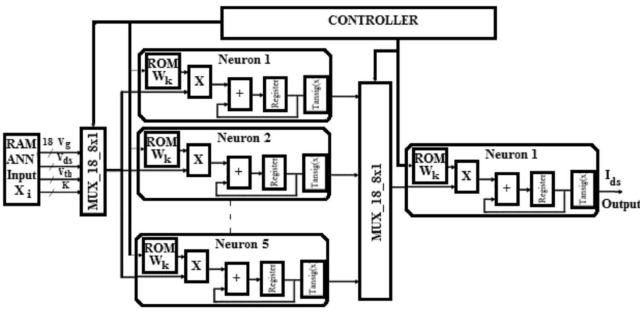

Fig. 3. The basic neuron computation scheme in neural networks based on VHDL.

4. Results and discussion

This section consists of two subsections. The first one concerns with the VHDL simulation results on ModelSim while the second section deals with the FPGA-implementation of the developed ANN–MOSFET model. The block diagram of intelligent model in this study is shown in Fig. 1. The intelligent model structure is composed of two stages, intelligent model with software and intelligent model with hardware.

In software stage, firstly, the ANN architecture of MOSFET is optimized according to the number of used hidden layers and the number of the neurons within these layers. Then the obtained weights and bias are recorded for the configuration of the ANN model of MOSFET on the FPGA. In this study, the software version 10.2c of Xilinx is used for obtaining the VHDL source code. For this, proper computing programs were created by using the VHDL. After the best architectures are created in VHDL, these configurations is simulated and tested under ModelSim Xilinx software.

Fig. 4. The hardware used for implementation.

Then in hardware stage, the best configuration is implemented into FPGA hardware board using a Xilinx Spartan-3E FPGA (XC3S500E) from Xilinx. For this, firstly, the proper computing program files written in VHDL for intelligent ANN model are embedded into FPGA hardware board to make them operative. Then the experimental realization of the electronic hardware circuit is implemented and tested. A picture of the experimental system of the electronics hardware circuit is shown in Fig. 4. Furthermore, for different parameter values, the comparisons between the ANN–MOSFET model implemented on FPGA with experimental output signals are shown in Fig. 5.

As shown in the results obtained, there is a good agreement between experimental data (I_{ds}) and ANN–MOSFET model data (I_{ds}) based on FPGA. The statistical value of the correlation coefficient r is obtained between 97% and 99%, which is very satisfactory. These results show the effectiveness of the implemented ANN–MOSFET model on FPGA. In comparison with related study the following results were obtained.

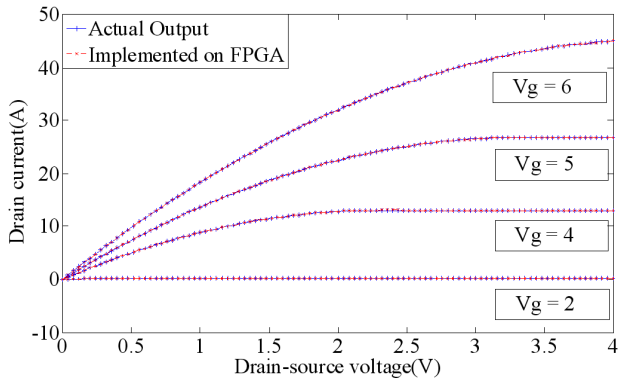

Fig. 5. The comparison between actual output and implemented on FPGA for $V_g = 2, 4, 5, 6$, $V_{th} = 1.7$, gain = 5.

The first comparison item is implementation area. The proposed FPGA-based intelligent hardware implementation [7, 10–13] is very suitable method for the semiconductor circuit elements.

The second comparison item is performance comparisons of the related study. The proposed method gives better results than the other traditional methods in terms of lower cost, rapid processing, and high efficiency when comparing the software and hardware applications [2, 5, 14]. This is because the FPGA and ANN have very fast parallel computation capabilities.

The third comparison item is the reconfigurations features of electronic devices. The execution time and hardware space used is excessively better on FPGA [7, 9, 10, 13] when comparing the classical electronic chips. FPGA chips have endless reconfigurations advantages according to ASIC's, CPLD's or DSP's.

5. Conclusion

In this paper, intelligent predictors for MOSFET are proposed and applied into FPGA Xilinx Spartan-3E FPGA (XC3S500E). The special-purpose hardware with FPGA programming can be used for very wide applications. FPGAs have the speed, security, and parallel processing and reconstruction ability. The designs made with FPGA based on ANN have enormous advantages according to VLSI in terms of time and cost. The implementation of artificial neural networks with FPGA provides flexibility in the programmed systems. The faster it converges, the more accurate results and the acceptable generalization for the system modelling are demonstrated by comparing experimental data with test results. The results indicate that the FPGA architectures are a good technique for MLP hardware applications.

References

[1] T.A. York, *Microproc. Microsyst.* **17**, 371 (1993).

[2] D.K. Iakovidis, D.E. Maroulis, D.G. Bariamis, *Microproc. Microsyst.* **31**, 160 (2007).

[3] J. Li, D. An, L. Lang, D. Yang, *Proc. Eng.* **29**, 2633 (2012).

[4] K. Kompton, S. Hauck, *ACM Comput. Surv.* **34**, 171 (2002).

[5] J. Lu, L. Jing, *Proc. Eng.* **16**, 858 (2011).

[6] G. Krampl, M. Rona, *Microelectron. J.* **33**, 855 (2002).

[7] P. Ferreira, P. Ribeiro, A. Antunes, F.M. Dias, *Neurocomputing* **71**, 71 (2006).

[8] S.K. Mandal, S. Sural, A. Patra, *IEEE Trans. Comput. Aid. Des. Integr. Circ. Syst.* **27**, 188 (2008).

[9] D. Shen, L. Jin, X. Ma, *Lect. Notes Comput. Sci.* **3173**, 988 (2004).

[10] O. Polat, T. Yıldırım, *Digit. Sign. Proc.* **20**, 881 (2010).

[11] H. Mekki, A. Mellit, H. Salhi, B. Khaled, *AIP Conf. Proc.* **1019**, 211 (2008).

[12] S. Saadi, A. Guessoum, M. Bettayeb, *Microproc. Microsyst.* **37**, 52 (2013).

[13] S. Li, M. Moussa, S. Areibi, *Canad. J. Electric. Comput. Eng.* **31**, 31 (2006).

[14] H. Mekki, A. Mellit, S.A. Kalogirou, A. Messai, G. Furlan, *Prog. Photovolt. Res. Appl.* **18**, 115 (2010).