# Implementation of Optimized Approximate Sigmoid Function on FPGA Circuit to use in ANN for Control and Monitoring

**Djalal Eddine.Khodja*, Stéphane Simard****
**Rachid Beguenan*****

** Faculty of technology , BP 161 Ichbilia Msila*
*Algeria (Tel: +213 355 519 24; e-mail: djalal_ed@yahoo.fr).*
*** Applied Sciences Department, UQAC, Canada,*
*(Tel:+154185455012 ; e-mail: stexyz@ymail.com)*
**** Depmt of Electrical, R.Military College Kingstone Canada,*
*(Tel:+1613544-8107e-mail; rachid.beguenane@gmail.com)}*

**Abstract:** In this paper, an approximation of sigmoid function in polynomial form has been proposed, then this function is optimized in order to implement that on FPGA using the Xilinx library. This implementation aim is to contribute in the hardware integration solutions in the areas such as monitoring, diagnosis, maintenance and control of power system as well as industrial processes. Since the Simulink library provided by Xilinx, has all the blocks that are necessary for the design of Artificial Neural Networks except a few functions such as sigmoid function. Tests results for control and diagnosis with FPGA Device are satisfactory.

*Keywords:* ANN, FPGA, Xilinx, Sigmoid Function, power system.

## 1. INTRODUCTION

Monitoring, control and maintenance of any element of wide area power system become an important issue for ensuring the continuity of electric supply and avoiding the black out. It is important to early detect the defects that can occur in these elements by monitoring their operations and then developing methods for applying adaptive control and preventive maintenance (Roy-Neogi et al., 1995; Renovell et al., 2001; Areibi, 2007). The fast implementation of the developed methods is necessary needs for the industry and the complex (smart) power systems (S. Simard et al., 2009). These methods may be investigated using several techniques that have different characteristics to solve the encountering problems (Altera Inc, 2005; Xilinx Corporation Inc, 2006). The most commonly used techniques are the artificial intelligence-based techniques such as Artificial Neural Networks (ANN) (Dick, 2002; Gadea, 2007) that they are easier to implement on electronic circuit board such as: Digital Signal Processing (DSP) chips, Application Specific Integrated Circuits (ASICs) or Field programmable gate array (FPGAs) (Mailoux et al., 2007; Armato, 2011; Ricci, 2003). The objective of this work is the implementation of artificial neural network on a FPGA. This implementation aim to contribute in hardware integration solutions using FPGA applied to different areas such as monitoring, diagnosis, maintenance and control of power systems. In this study, is beginning by adapting ANN to allow optimal implementation. This implementation must ensure efficiency, timeless and a minimum possible space on the FPGA. Then, we schedule the ANN on the System Generator in order to use that in control and Diagnosis. The System Generator to generate VHDL code. This code is verified and implemented on a FPGA Spartan-like by the ISE software from Xilinx Foundation.

## 2. APPROXIMATE OF SIGMOIDE FUNCTION

The principle of the library provided by Xilinx is almost the same as of the classical Simulink library. It contains blocks representing different functions that are ready to be connected with each other to form algorithms (Mailoux, 2007). These blocks do not only serve to simulation, but can also generate VHDL or Verilog code.

In addition, the library provided by Xilinx to Simulink, has all blocks that are necessary for the design of ANN, however some activation functions are not available and an example of them is the sigmoid function.

For this reason, in the paper, the authors suggest a method which is based on the Vandermonde's algorithm to implement the sigmoid function (Armato, 2011). The sigmoid function is given by the following:

$$f(x)=\frac{1}{1+e^{-cx}} \tag{1}$$

This function can be approximated by second order polynomial equation:

$$f(x)=c+bx+ax^2 \tag{2}$$

The only unknowns in this equation are a,b and c which are determined using Vendermond's algorithm. Let's have three points from the sigmoid function $\{(x_{i,}, y_i)$, i=1, 2 and $3\}$ which satisfy the following:

$$\begin{cases} y_1 = \dfrac{1}{1+e^{-x1}} = c + bx_1 + ax_1^{\,2} \\ y_2 = \dfrac{1}{1+e^{-x2}} = c + bx_2 + ax_2^{\,2} \\ y_3 = \dfrac{1}{1+e^{-x3}} = c + bx_3 + ax_3^{\,2} \end{cases} \quad (3)$$

Equation 3 can be rewritten under matrix form as follows:

$$Y = X * A$$

Where:

$$X = \begin{bmatrix} x_1^{\,2} & x_1 & 1 \\ x_2^{\,2} & x_2 & 1 \\ x_3^{\,2} & x_3 & 1 \end{bmatrix}, \quad Y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} \text{ and } \quad A = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \quad (4)$$

As results, the coefficients are calculated by the following equation:

$$X^{-1}Y = A \quad (5)$$

With

$$f(x) = \begin{cases} -0.0332x^3 + 0.2549x + 0.5 \quad for \; -4 \geq x < 0 \\ +0.0332x^3 + 0.2549x + 0.5 \quad for \quad 0 \leq x < 4 \end{cases} \quad (6)$$

*2.1. Modeling of sigmoid function by Xilinx*

In order to implement this obtained function, we must decide on an input accuracy and precision for reasonable output sigmoid function (Tommiska et al., 2003, Tisan et al., 2009). As agreed, for the approximation of this equation, it needs the bounds of the domain of x as input to adjust the number of bits and the fractional precision. The output current varies between 0 and 1. It also needs to know the accuracy required output.

It should be noted that mathematically, the area of the sigmoid is not bounded. In these models, it uses floating point arithmetic in double precision, so we do not perceive any limit (Tisan et al., 2009). But in fixed point arithmetic, especially in hardware, we must define fairly tight bounds on the inputs and outputs, and the infinite must be bounded (Zwinngelsten, 1995; B.Dubuisson, 1992). Certainly in practice, the variable x does not vary between - infinity and + infinity in the neural network. Anyway, It should be consider the fact that the A / D input and outputs are all-at-most 16 bits of precision in total. It will be try to judge from the model, where its hardware implementation is presented in figure1.

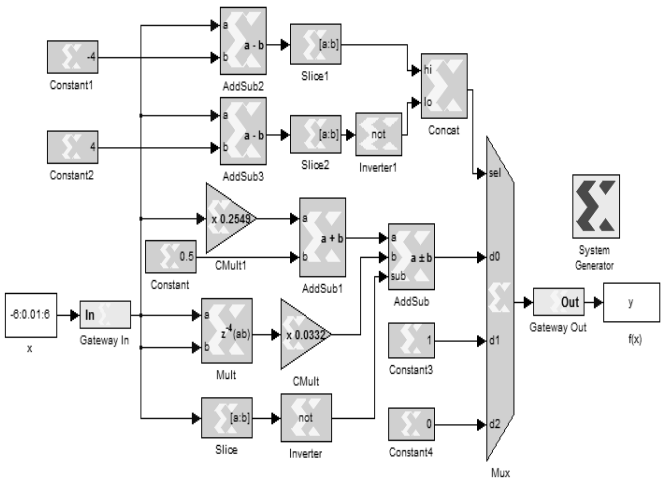



Fig. 1. Modeling of Sigmoid approximation function.

The comparative and errors representations introduced by approximated sigmoid function beside hardware implementation are shown in figure 2. In these tests, the numbers of bits allocated are : 32 bits with 16 bits binary point (32,16) , then (16,8) and (8,4) (Tisan et al., 2009).

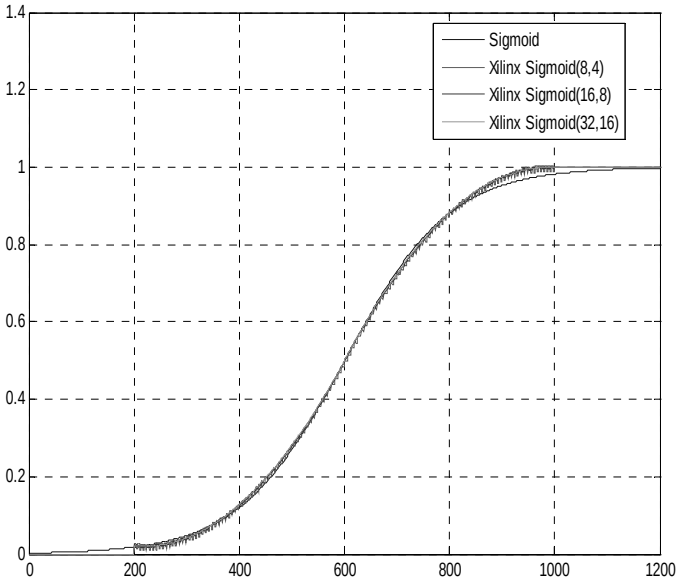



Fig. 2. Comparative between the sigmoid function and Xilinx approximated Sigmoid.

The analysis of results presented in figure 3, it can be notice that the maximum error introduced by representation (8,4) is only 0.485%. it is smaller than 0.6335% obtained by (16,8) in (Tisan, 2009).

However, the total equivalent logic gates used for hardware implementation, for this approximated sigmoid function, which has been obtained with Xilinx Integrated Software Environment (ISE) is presented in table 1.

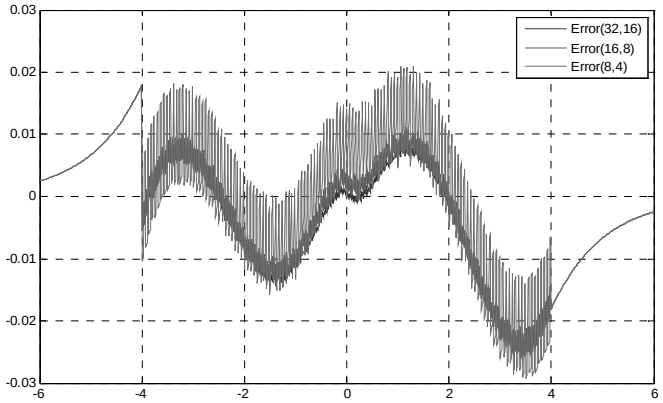



Fig. 3. Error between the sigmoid function and Xilinx approximated functions.

**Table 1. Results of synthesis of obtained sigmoid function with different fixed points.**

| Device utilization summary: Selected Device : 4vsx35ff668-10 | | | | |
|---|---|---|---|---|
| Logic utilization | Different fixed points | | | *Availabl e* |
| | (8,4) | (16,8) | (32,16) | |
| Slices | 11 | 19 | 40 | 15360 |
| Slice Flip Flops | 12 | 28 | 54 | 30720 |
| 4 input LUTs | 02 | 08 | 16 | 30720 |
| bonded IOBs | 20 | 52 | 90 | 448 |
| GCLKs | 2 | 2 | 2 | 32 |

*2.2. Optimization of the approximate sigmoid function*

In order to implement the obtained polynomial function, it is should be to decide on an input accuracy and precision for reasonable output sigmoid function (Khodja et al., 2005; Tolovka et al., 2001).

One can without difficulty increase the accuracy of the approximation of the sigmoid, but this has a cost. The more accurate is the approximation, the more bulky and slower is the implementation. This must be taken into account and use only the minimum necessary precision in the control circuits. One can use as much blackboxes as we can take care of managing the CLK and CLK_EN signals however according to (Moubayed et al., 2007; Atencia et al., 2007; Polat et al., 2010), simulations this way are slower than working in Simulink. If it's more convenient, the code of the approximation of the sigmoid could easily be described with the Xilinx Blockset in Simulink, rather than VHDL, since the form of the equation is not complicated. The only difficulty is finding the right parameters for the approximation (that is to say the coefficients of the equation), but there is a way to do that, then it becomes quite easy to adjust them according to any need (taking into account the cost of the implementation).

Knowing that the output of each neuron is between 0 and 1, we considered several factors, including synaptic weights used in our example. According to a rough estimate, we can conclude that the necessary precision before the decimal point in the sigmoid functions does not exceed 9-10 bits (depending on the application) in the input layer and 6 bits in the other 2 layers. Regarding the precision after the decimal point, given the fact that the effect of a neural network is a global phenomenon, the sigmoid functions need not to be very accurate. It is enough for the approximated function to follow fairly well the sigmoid curve, and a precision of 8-10 bits after the decimal point seems more than enough to give exactly the same output as our final example. Thus, we have implemented an approximation of the sigmoid in System Generator and have performed several experiments. However, all the parameters mentioned above have to be optimized based on the application. We notice that the number of bits needed before the decimal point is greater in the input layer of neural network than in the other 2 layers. This has a lot of influence on the circuit size. One can adjust the number of bits in the properties editor of the *Gateway In* block in the sigmoid. In the output, precision is fixed to 10 bits after the decimal point. The comparative graphs between sigmoid function and Xilinx optimized function (27 bits with 10 bits binary point) as shown in Figure 4. In fact, the maximum error is 0398%, See Figure 5.

The results of synthesis of optimized sigmoid function are shown in table.2. It can reveal that the VHDL code of sigmoid function occupies only 25 of 30720 Slice Flip Flops of the used FPGA.

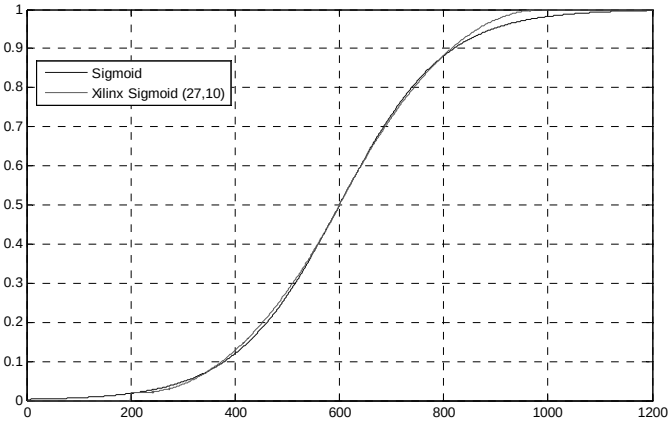



Fig. 4. Comparative between the sigmoid function and Xilinx optimized Sigmoid function (27,10).

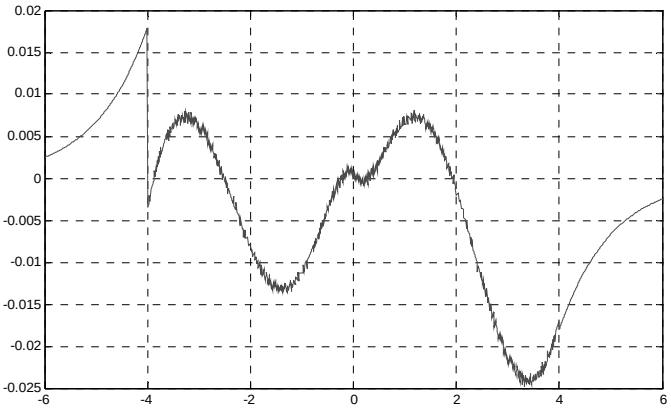


Fig. 5. Error between the sigmoid function and Xilinx optimized function (27,10).

**Table 2. Results of synthesis of optimized sigmoid function (27, 11).**

| Device utilization summary: Selected Device : 4vsx35ff668-10 | | |
|---|---|---|
| Slices | 25 out of 15360 | 0% |
| Slice Flip Flops | 38 out of 30720 | 0% |
| 4 input LUTs | 10 out of 30720 | 0% |
| bonded IOBs | 75 out of 448 | 16% |
| GCLKs | 2 out of 32 | 6% |

## 3. APPLICATION OF ANN UNDER XILINX FOR DTC USING THE OPTIMIZED SIGMOID FUNCTUIN

In order to perform the co-simulation of ANN for DTC of induction machine using FPGA soft processor, the obtained ANN must run on the FPGA board which requires converting ANN's Matlab code to VHDL. This task is subdivided to three parts, namely:

a. Building ANN block in Simulink/ Xilinx;
b. Generation the ANN Xilinx block;
c. Network testing by Simulation.

### *3.1 Building ANN block using Simulink/Xilinx*

The application of the technique of neural network in machine control is simple and allowed the resolution of several problems related to control these systems. In this work with DTC, it is easy to use this technique, where the conventional DTC is replaced by controller based on neural networks as is illustrated in Figure.6.

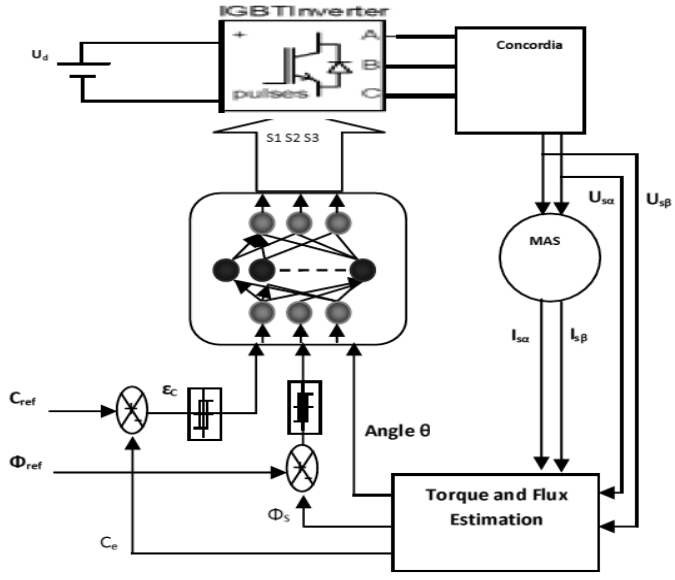



Fig. 6. Scheme of DTC_based ANN technique.

The proposed neural network is a multilayer network (3-6-3) with the architecture. Each neuron is connected to all neurons of the next layer by connections whose weights are randomly chosen real numbers. We notice that $w_{xy}$ is the weight of the connection between neurons x and y. The following steps are necessary to obtain this ANN:

. ANN topology.
. ANN Learning stage
. ANN validation

The proposed artificial network consists of three layers, namely: the input layer consists of three neurons, whose function is to transmit the input values that correspond to the input variables to the next layer called hidden layer. The hidden layer is characterized by six neurons with sigmoid shaped activation function. The output layer is composed of three neurons whose output is either 0 or 1.

The second stage in designing the ANN is the learning process which requires a data base defining the ANN input-output mapping. This data base is mostly given under matrix form as to clarify the inputs and the desired outputs according to the switching table of DTC, see table.3.

**Table. 3 switching table of the conventional DTC.**

| N | | $N=1$ | $N=2$ | $N=3$ | $N=4$ | $N=5$ | $N=6$ |
|---|---|---|---|---|---|---|---|
| cflx | ccpl | | | | | | |
| 1 | 1 | $V_2$ | $V_3$ | $V_4$ | $V_5$ | $V_6$ | $V_1$ |
| 1 | 0 | $V_6$ | $V_1$ | $V_2$ | $V_3$ | $V_4$ | $V_5$ |
| 0 | 1 | $V_3$ | $V_4$ | $V_5$ | $V_6$ | $V_1$ | $V_2$ |
| 0 | 0 | $V_5$ | $V_6$ | $V_1$ | $V_2$ | $V_3$ | $V_4$ |

In this application, the input matrix consists of three inputs (lines) corresponding to :
• The first variable is the position of the flux in the reference frame related to the stator.
• The second input variable is used the state variable error flux.
• The third input variable, the state variable error of the couple is used.

This input values are done with this matrix:

a= [1 1 1; 1 1 0; 1 0 1; 1 0 0; 2 1 1; 2 1 0; 2 0 1; 2 0 0; 3 1 1; 3 1 0; 3 0 1; 3 0 0; 4 1 1; 4 1 0; 4 0 1; 4 0 0; 5 1 1; 5 1 0; 5 0 1; 5 0 0; 6 1 1; 6 1 0; 6 0 1; 6 0 0];

The output is represented by the pulses of the inverter switches that represent the values zero or one, there matrix called desired output is done by.

d= [1 1 0; 0 1 0; 1 0 1; 0 0 1; 0 1 0; 0 1 1; 1 0 0; 1 0 1; 0 1 1; 0 0 1; 1 1 0; 1 0 0; 0 0 1; 1 0 1; 0 1 0; 1 1 0; 1 0 1; 1 0 0; 0 1 1; 0 1 0; 1 0 0; 1 1 0; 0 0 1; 0 1 1];

Using Graphical User Interface (GUI) of the Neural Network Toolboox in Matlab, it's easy from a given input – output data to train the proposed ANN. The Levenberg – Marquardt backpropagation algorithm is used to train the proposed network topology. To measure the network performance, the Mean Squared Error between the target and the output of the fitting network is used. The optimal values of the Neural Network weights have been obtained after 56 iterations with an error of 3.58058e-012. The error is very small however the obtained Network must be checked using validation test.

Modeling the resulting of ANN Block Co-simulation block is given by Xilinx as shown figure.7:

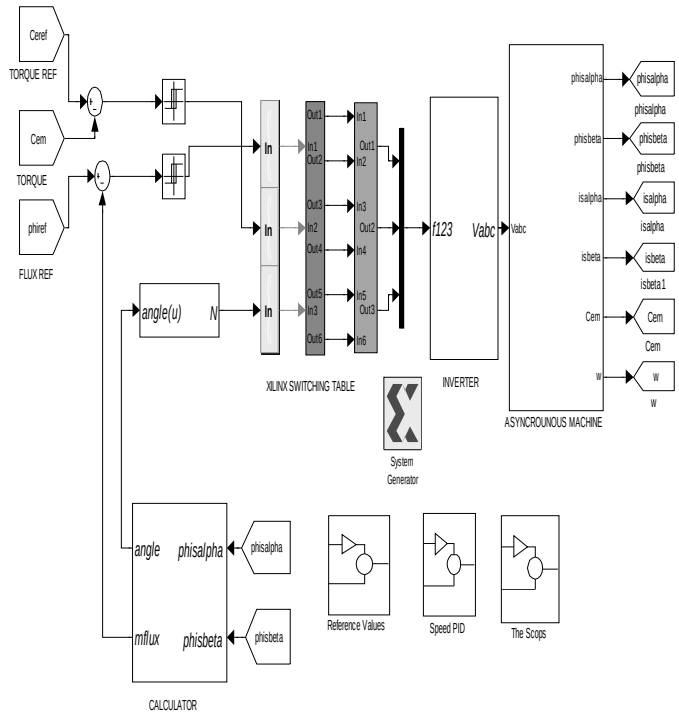



Fig. 7. Modeling of ANN of DTC under Xilinx.

The results of synthesis of obtained ANN Xilinx Bloc is shown in table 4.

**Table 4. Results of synthesis of obtained ANN Block under Xilinx.**

| Device utilization summary: Selected Device : 4vsx35ff668-10 | | |
|---|---|---|
| Number of Slices | 6613 out of 15360 | 43% |
| Number of Slice Flip Flops | 6633 out of 30720 | 21% |
| Number of 4 input LUTs | 11726 out of 30720 | 36% |
| Number of bonded IOBs | 115 out of 448 | 25% |
| Number of GCLKs | 1 out of 32 | 3% |

From the results obtained on the table.3 reveals that the VHDL code occupies an area of 21% on FPGA, means that the FPGA type Virtex4 XC4VSX35-1011668 supported this VHDL code.

*3.2 Simulation Results of DTC by ANN on FPGA*

The test result shows the behavior of the structure of the DTC with ANN-Based Xilinx applied to the induction machine. One can see clearly that there is a decrease on the electromagnetic torque ripple and also note that the electromagnetic torque accurately follows its reference and it is its response time (1.5 sec).

Otherwise, One can notice that the implemented ANN has generated the right binarry code according to the switching table in the conventional DTC applied to the induction motor. It is easy to filter or eliminate these harmonics during themotor operating by increasing the fixed point.

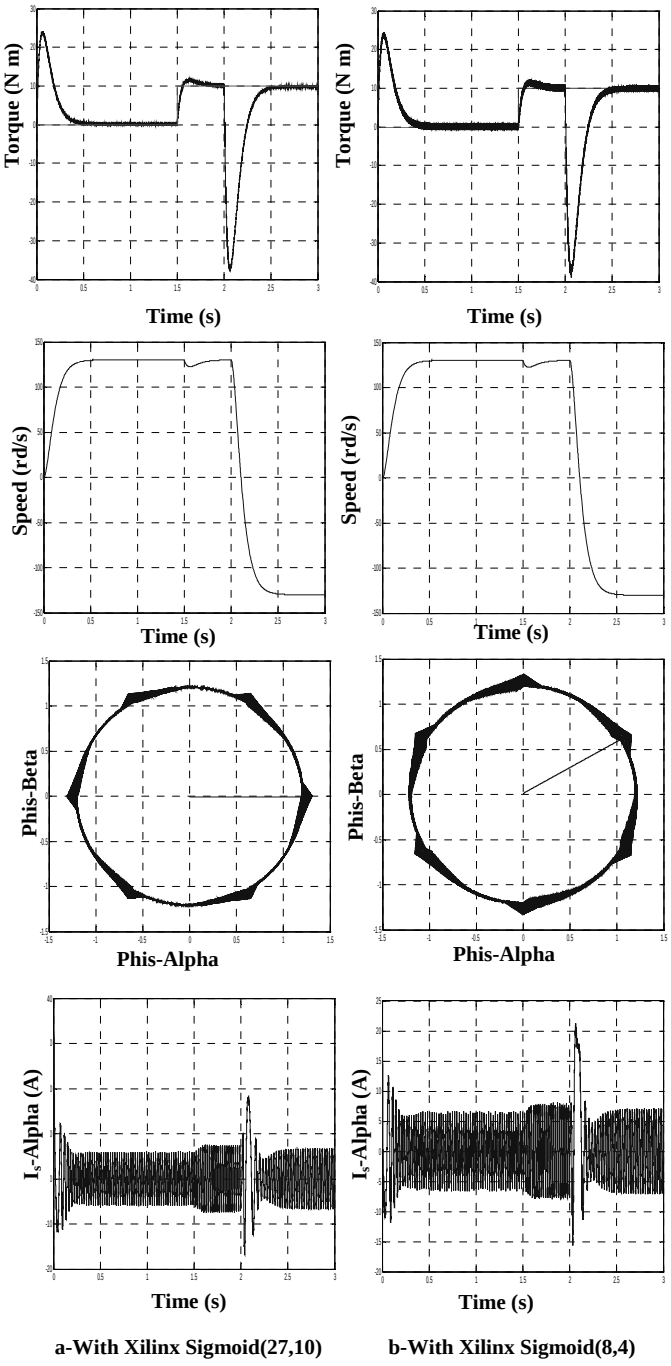



Fig. 8. Simulation results with DTC by ANN block under Xilinx (use of optimized Sigmoid function (27,11) and (8,4)).

## 4. APPLICATION OF ANN-XILINX FOR DIAGNOSIS OF IM FAULTS WITH OPTIMIZED SIGMOID FUNCTION

The proposed neural network is to identify the induction motor faults. this artificial network consists of three layers, namely: the input layer consists of three neurons, whose function is to transmit the input values that correspond to the input variables ($V_{eff1}$, $V_{eff2}$ and $V_{eff3}$) to the next layer called hidden layer. $V_{effi}$ stands for the effective value of the *i-th* voltage (*i=1,2* or *3*). The output layer is composed of three neurons whose output is either 0 or 1. The RMS voltages

$V_{eff1}$, $V_{eff2}$ and $V_{eff3}$ are calculated by using the RMS block which is available in Simulink library.

*4.1 ANN Learning stage*

The second stage in designing the ANN is the learning process which requires a data base defining the ANN input-output mapping. This data base is mostly given under matrix form as to clarify the inputs and the desired outputs. In our application, the matrix consists of three inputs (lines) corresponding to the three RMS voltages ($V_{eff1}$, $V_{eff2}$, $V_{eff3}$) and three digits (outputs) giving a code which corresponds to the appropriate fault.

An optimal learning stage requires that the database must be very rich and cover different types of faults. For this purpose, the following tasks are completed:

- The machine is simulated in normal (healthy) state;
- The machine is simulated in abnormal regime (in the presence of faults: single phase, two-phase,..etc.);
- The RMS values have been taken in each case including the healthy state

This can be summarized by a classification of different states as shown in table 5.

**Table 5. Classification of different faults of IM.**

| Category | faillure Type | Symbol | Code | | |
|---|---|---|---|---|---|
| 1 | Healthy State | ES | 0 | 0 | 0 |
| 2 | Single-phase cut 1 | CM1 | 1 | 0 | 0 |
| 3 | Single-phase cut 2 | CM2 | 0 | 1 | 0 |
| 4 | Single-phase cut 3 | CM3 | 0 | 0 | 1 |
| 5 | Unbalance single-phase1 | CB1 | 1 | 1 | 0 |
| 7 | Unbalance single-phase2 | CB2 | 0 | 1 | 1 |
| 6 | Unbalance single-phase3 | CB3 | 1 | 0 | 1 |

Using Graphical User Interface (GUI) of the Neural Network Toolboox in Matlab, it's easy from a given input – output data to train the proposed ANN. The Levenberg – Marquardt backpropagation algorithm is used to train the proposed network topology. To measure the network performance, the Mean Squared Error between the target and the output of the fitting network is used. The optimal values of the Neural Network weights have been obtained after 56 iterations with an error of 7.69704e-014. The error is very small however the obtained Network must be checked using validation test.

In contrast to the method used in Matlab, which consists in dividing the learning data base to two parts (Khodja et al., 2005; Hafaifa et al., 2013), 70% is used for training and the remaining for the validation test, in the paper we have used another data base to test the generalization ability of the trained neural network. The same faults have been introduced but the times at which these faults occur have been changed according to table.5 (). Therefore, the ANN has been simulated and the different faults have been introduced while the induction motor is working. The results given by the trained network and the behavior of the motor are depicted in figure 6. The obtained simulation results show clearly its ability to differentiate faults according to the input-target data.

**Table 6. Times of faults application.**

| Time of setting | Fault types |
|---|---|
| à t =2s | Single-phase cut 1 |
| à t = 4s | Healthy State |
| à t = 5s | Single-phase cut 2 |
| à t = 7s | Healthy State |
| à t = 8s | Single-phase cut 3 |
| à t = 10s | Healthy State |
| à t =11s | Unbalance single-phase2 |

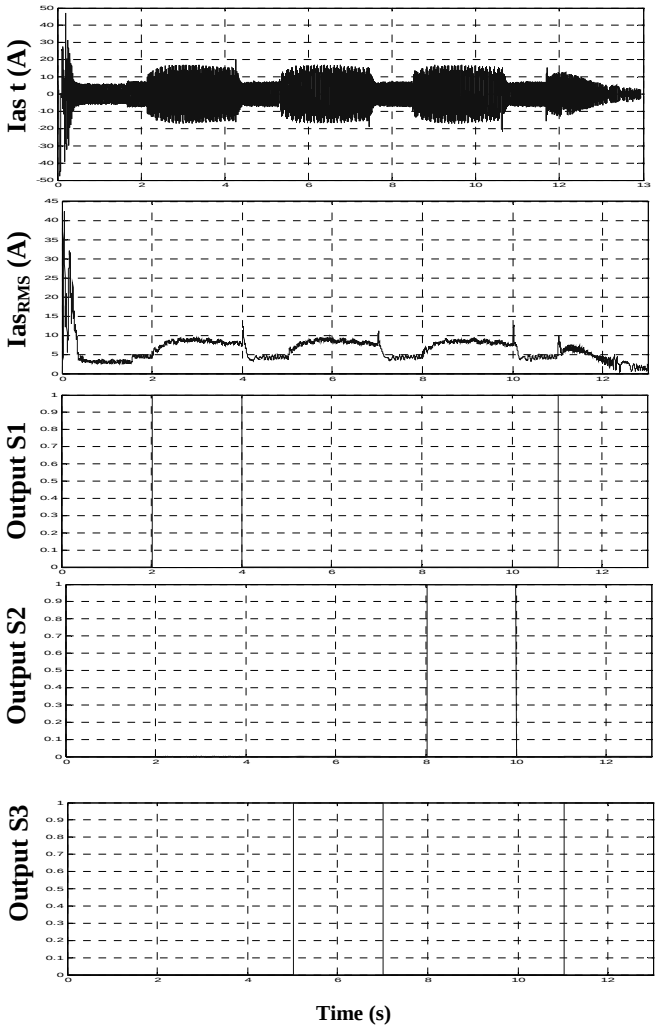



Fig. 9. The stator current with different faults and the ANN outputs.

*4.2. Building ANN block using Simulink /Xilinx*

In order to perform the simulation of ANN diagnosis of induction machine faults using FPGA soft processor, the obtained ANN must run on the FPGA board which requires converting ANN's Matlab code to VHDL.

The design of ANN using Simulink / Xilinx is shown in Figure 10.

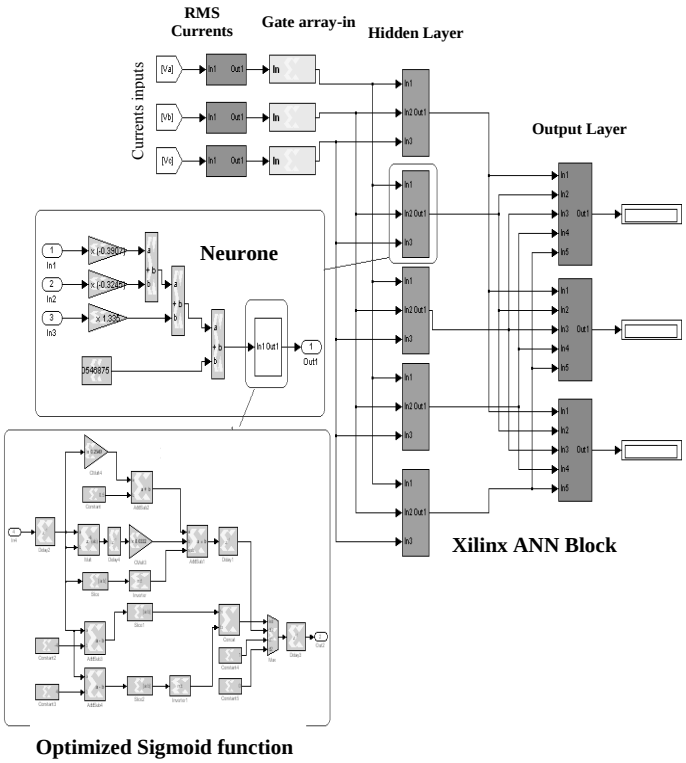
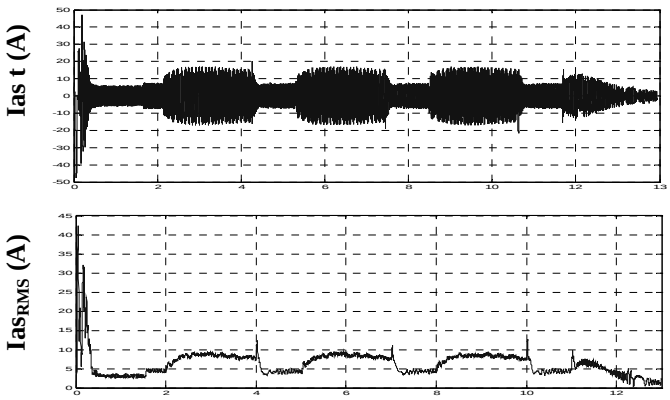



Fig. 10. ANN modeling in Simulink / Xilinx ANN block with Virtex4 Device).

*4.3 ANN testing by co-simulation*

Once the ANN code is dowloaded on the FPGA board, simultion of the whole system will be ready carry on. To check the performance of the prposed ANN as well as the opimized approximated activation sigmoid function, the same scenario used to validate the ANN, has been coducted for this test. The results shown in figure 11 show respectively the instatanous stator current, the stator current and the three ANN binnary outputs that identify the fault type. One can notice that the implemented ANN has generated the right binarry code according to fault applied to the motor. It is easy to filter or eliminate the ANN ouput during the motor starting otherwise it does not matter as the signal do not last long as to be confused with consequential one (See tables 5 and 6).



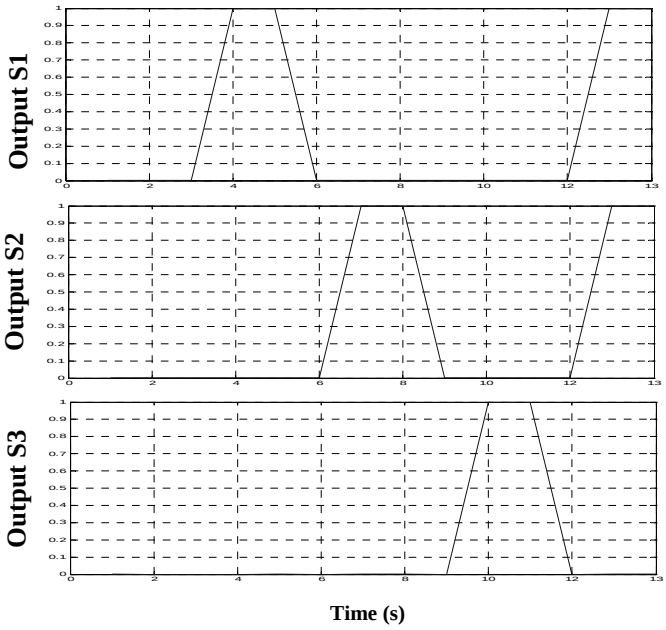




Fig. 11. Test of ANN made by Xilinx Simulink.

Finally, the figure 12 shows the association of the induction motor Simulink model with the artificial neural network system Xilinx model. The System Generator is used to convert the Xilinx model (ANN), generate the equivalent VHDL code and download it into the FPGA processor board (Virtex4 XC4VSX35-1011668). The system will look like that shown in figure 12. The connection of the FPGA to Matlab / Simulink in the PC is made through a Ethernet cable.

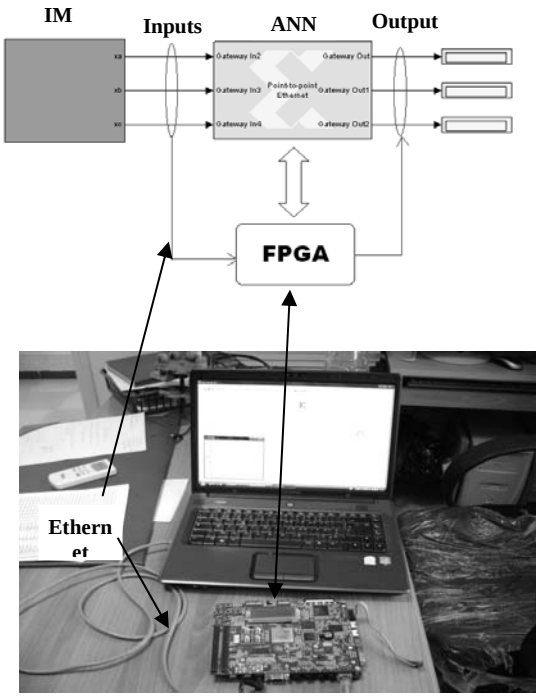



Fig. 12. Principle of implementation of ANN block on FPGA using Co-simulation (Co-simulation of ANN block with Virtex4 Device).

## 5.CONCLUSION

In this work, an ANN has been developed and implemented for purpose of diagnosis of induction machine faults and its control with DTC technique. In fact, the most important parameters in the implementation procedure is the modeling of the activation sigmoid function which is very commonly used in ANNs. In this work a method has been applied to approximate the sigmoid function by 2nd order polynomial equation resulting in less computation time and memory requirements.

The comparative representations and errors introduced by approximated sigmoid function beside hardware implementation with several fixed points ((32,16); (16,8) and (8,4)) were shown. Then, an optimized approximated sigmoid function has been proposed in order to minimize the number of logic gates and improving the computation time.

The proposed ANNs have five neurons hidden layer, three binary outputs and three inputs– targets data for motor diagnosis and control are available; Levenberg-Marquardt backpropagation algorithm of Neural Network Toolbox has been used to train the ANN. The results have shown through the validation test that the obtained synaptic weights of the ANN are the optimal values by obtaining a very small mean squared error.

The system generator has been used to generate the VHDL code corresponding to the ANN model and downloaded into the FPGA Vertex 4 board. Co-Simulation of the induction motor modeled using Simulink blocks and the ANN running in the FPGA has been successfully done. The obtained ANN outputs show the effectiveness of the proposed topology and the optimization of activation sigmoid function.

The use of high-level design tool such as system generator is very beneficial for the verification and design of any complex diagnosis or control algorithm.

## REFERENCES


A. Areibi, G. Grewal, D. Banerji and P. Du. *Hierarchical FPGA Placement. CAN*. J. ELECT. COMPUT. ENG., VOL. 32, NO. 1, Winter 2007.

A. Arias, Luis Romeral, Emiliano Aldabas, Marcel Jayne, « Stator flux optimised Direct Torque Control system for indukction motors », Electric Power Systems Research, Elsevier, 73 (2005) 257–265

A. Armato, L. Fanucci, E.P. Scilingo, D. De Rossi. *Ow-error digital hardware implementation of artificial neuron activation functions and their derivative*. Microprocessors and Microsystems, Elsevier 35 (2011) 557–567.

A. Hafaifa, A.Z. Djeddi, A, Daoudi, *Fault detection and isolation in industrial control valve based on artificial neural networks diagnosis*. CEAI, Vol.15, No.3 pp. 61-69, 2013.

A.Tisan, S.Oniga, D.Mic, A.Buchman. *Digital Implementation of the Sigmoid Function for FPGA Circuits*. ACTA TECHNICA NAPOCENSIS, Electronics and Telecommunications, Volume 50, Number 2, 2009, PP: 15-20.

Altera, Inc., http://www.altera.com. 2005.

B. Dubuisson. *Fault Detection and Diagnostic of Process*. Technics of ingeneer. R7597,1992.

B.Chetate, DJ.Khodja. *Online Faults Diagnostic of Assembly of Asynchrone Machine-Inverter Using Artificial Neuron Networks*. Journal Electrotekhnika, Moscou 12/2003, pp:16-20.

C. Dick. *The Platform FPGA: Enabling the Software Radio*. Software Defined Radio Tech. Conf. and Product Expo. (SDR), 2002.

D. Kumsuwan, S. Premrudeepreechacharn, H. A. Toliyat, « Modified direct torque control method for induction motor drives based on amplitude and angle control of stator flux», Electric Power Systems Research, Elsevier, 78 (2008) 1712–1718.

DJ. Khodja, B.Chetate. *ANN system for identification and localisation of faillures of anatomawed electric asynchronous drive*. 2nd International Symposium on Electrical, Electronic and Computer engineering en Exhibition, March, 11-13, NEU-CEE 2004, NICOSIA, TRNC pp : 156-161.

DJ. Khodja, B.Chetate. *Development of Neural Network module for fault identification in Asynchronous machine using various types of reference signals*. 2nd International Conference PHYSICS and CONTROL, August,24-26, Physcon 2005, St Ptersburg, Russia. pp : 537-542.

E. Zio, Giulio Gola. *A neuro-fuzzy technique for fault diagnosis and its application to rotating machinery*, Reliability Engineering and System Safety, Elsevier, 94 (2009) 78–88.

F. Ricci, Hoang Le-Huy, « *Modeling and simulation of FPGA-based variable-speeddrives using Simulink* », Mathematics and Computers in Simulation, Elsevier, 63 (2003) 183–195.

G.Zwinngelsten. *Diagnosis of failures: theory and practice for industrial systems*. Ed. Hermès Paris. 1995.

J.G.Mailoux, S.Simard, R.Beguenane. *Implementation of Division and Square Root using XSG for FPGA-Based vector control Drives*. IJEPE 2007, ISSN: 1990-7958. PP: 524-529.

K. Renovell, P. Faure, J.M. Portal, J. Figueras and Y. Zorian. IS-FPGA: *A New Symmetric FPGA Architecture With Implicit SCAN*. IEEE ITC International Test Conference, Paper 33.1, 0-7803-7169-0/2001.

M. Atencia, Hafida Boumeridja, Gonzalo Joya, Francisco Garci, ´a-Lagos, Francisco Sandoval. *FPGA implementation of a systems identification module based upon Hopfield networks*. Neurocomputing, Elsevier. 70 (2007) 2828–2835.

M.S. Carmeli, M. Mauri, « Direct torque control as variable structure control: Existence conditions verification and analyskis», Electric Power Systems Research, Elsevier, 81 (2011) 1188–1196

M.T. Tommiska. *Efficient digital implementation of the sigmoid function for rprogrammble logic*. IEE 2003, Computer and Digital Techniques, PP:403-411.

N. Moubayed. *Detection and Localisation of Faults in the Static Inverters. 6TH International Conference on*

*Electromechanical and Power Systems*, Chişinău, Rep. Moldova, October, 2007.

Ö. Polat, Tülay Yildirim. *FPGA implementation of a General Regression Neural Network: An embedded pattern classification system*. Digital Signal Processing, Elsevier. 20 (2010) 881–886.

R.Gadea, J.Cerdá, F.Ballester, A.Mocholí. *Artificial Neural Network Implementation on a single FPGA of a Pipelined On-Line Backpropagation*. Proceeding ISSS2000 Conference, Madrid, Spain. 2000 IEEE 1080-1082/00 PP :225-230.

S. Simard, R. Beguenane, J-G, *Mailloux, Performance Evaluation of Rotor Flux-Oriented Control onFPGA for Advanced AC Drives*, pp:113, 117, Journal of Robotics and Mechatronics Vol.21 No.1, 2009.

S. Vaez-Zadeh, E. Jalali, « Combined vector control and direct torque control method for high performance induction motor drives », Energy Conversion and Management, Elsevier 48 (2007) 3095–3101

V.A.Tolovka. *Neural Networks : Training, Organisation and Useful.* Ed, Establshement journal Redaction 'Radiotekhnika' Moscou, 2001.

Xilinx Corporation, Inc., www.xilinx.com 2006.

## APPENDIX

$P_n$: Nominal power 1.1 kW
$V_n$: Voltage of a stator phase 220 V
$I_n$: Current Stator phase 3.9 A
p: Number of pairs of poles 1
$f_s$: Stator frequency of tension 50 Hz
$R_S$: Resistance of a stator phase 7.58 Ω
$R_R$: Resistance of the rotor cage 6.3 Ω
$L_R$: Rotor inductance 0.1612H
$L_S$: Stator cyclic inductance 0.5976H
$J_m$: Moment of inertia 0.0054 Nms2
$N_S$: Number of coils per stator phase: 160
$M_{SR}$: Mutual inductance stator nets 26.5mH
FPGA Processor board (Virtex4 XC4VSX35-1011668)
IM: Induction Machine