NEURAL ASIC CONTROLLER FOR PWM POWER SYSTEMS

A. Dinu, M. N. Cirstea, M. McCormick
De Montfort University, Leicester UK
dinu@dmu.ac.uk,marci@dmu.ac.uk,

Antonio Ometto, Nicola Rotondalle
University of L'Aquila Italy
ometto@ing.univaq.it

Abstract

This paper describes an ASIC approach of a predictive current controller for VSI-PWM inverters used in power systems. The new structure is based on FPGA implementation of neural networks providing high performance and compact hardware structure. The operation principles, architecture, design and implementation strategy are presented alongside with simulation results.

I. Introduction

The current control strategy plays the most important role in current-controlled PWM inverter-fed drive systems in which fast current response is required. A high current derivative switching mode must be chosen to produce the desired high speed current response. The authors have developed and adapted the switching strategy initially proposed by Nabae, Ogasawara and Akagi in [1]. This paper presents the novel control scheme based on neural networks which able to optimally solve the quick response problem for transient operation. When used in conjunction with complex control strategies for drive systems, it allows the elimination of the speed and/or position transducer depending on the motor type.

The use of the neural networks ensures a high speed of operation and a compact implementation of the novel switching strategy. The networks were trained but constructed on simple logic and not trained thereby avoiding the draw-backs of the classical back-propagation training algorithm. The analogue models were converted into digital structures containing elementary logic gates.

II. The Structure of the Control System

The control system was designed as a high speed digital ASIC. It comprises four main blocks illustrated in Fig. 1. The load voltage observer and the calculation block ('Calc.') are classical digital structures whereas the PWM generator and the inductance estimator were implemented as digital neural networks to obtain high speed of operation.

III. The State Space Observer

The principle of the state space observer relies on the equation relating R, L and the load voltage, which in terms of space vectors can be written as:

$$\underline{u}(t) = R\underline{i}(t) + L\frac{d\underline{i}(t)}{dt} + \underline{e}(t) \qquad (1)$$

If the sampling process is taken into account then the equation above becomes:

$$\underline{u}(k) = R\underline{i}(k) + \frac{L}{T_S}\left[\underline{i}(k) - \underline{i}(k-1)\right] + \underline{e}(k) \qquad (2)$$

The non-inductive load voltage is given by the expression:

$$\underline{V}_l(k) = R\underline{i}(k) + \underline{e}(k) \qquad (3)$$

Therefore the equation for the state space observer can be written as:

$$\underline{V}_l(k) = \underline{u}(k) - \frac{L}{T_S}\left[\underline{i}(k) - \underline{i}(k-1)\right] \qquad (4)$$

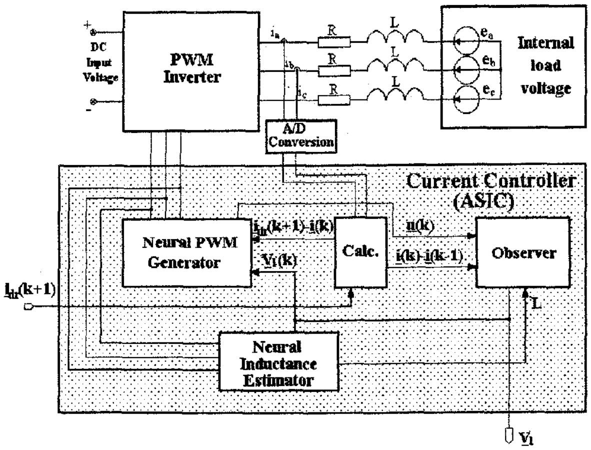

Fig. 1 - The Novel Current Control Scheme

0-7803-4980-6/98/$10.00 © 1998 IEEE

IV. The Neural PWM Generator

The neural PWM generator produces the control voltages for the transistors in the inverter so that the inverter output voltage is adequate for maintaining the required currents across the load. The voltage for the next sample period will be calculated so that the current at the end of the period will equal the desired one:

$$\underline{u}_{dr}(k+1) = R\underline{i}(k) + \frac{L}{T_S}\left[\underline{i}_{dr}(k+1) - \underline{i}(k)\right] + \underline{e}(k) \quad (5)$$

The PWM inverter contains 6 transistors connected into 3 pairs. In each pair one transistor is switched on and the other off. The state of each pair can be described by one bit. This can be either 1 or 0 depending on which transistor is switched on. The state of the inverter can be described by 3 bits. There are only seven different output voltage space vectors attached to the eight possible states of the inverter. This is due to an identical result (short circuit) corresponding to states (1,1,1,) and (0,0,0). The zero output voltage (0,0,0) will not be discussed in this paper. The space vector of the calculated output voltage is not generally identical to any of these seven vectors but the most appropriate one has to be chosen, that is the one which generates the closest resulting current to the desired one $\underline{i}_{dr}(k+1)$ at the end of the sampling period. Equation (5) may be rewritten as:

$$R\left[\underline{i}_{dr}(k+1) - \underline{i}(k)\right] = \frac{RT_S}{L}\left[\underline{u}_{dr}(k+1) - \underline{V}_I(k)\right] \quad (6)$$

Equation (6) demonstrates that for each sampling period the vertex of the current space vector $\underline{i}(k)$ will undergo a shift along the direction determined by the vector $[\underline{u}(k+1)-\underline{V}_I(k)]$. Therefore the strategy proposed for finding the optimum output voltage implies the minimization of the angle between the vector $R[\underline{i}_{dr}(k+1)-\underline{i}(k)]=R\Delta\underline{i}_{dr}$ and the vector $[\underline{u}_{INV}(k+1)-\underline{V}_I(k)]$ (where u_{INV} is one of the possible output voltages of the inverter). In other words their scalar product is maximized.

The algorithm presented above is complex and time consuming as it implies a series of multiplications and divisions to calculate the maximum of six different angles by algebraic and trigonometric means. In order to avoid this problem a four-layer feed-forward artificial neural network (FFANN) was used. It was "constructed" on the base of Voronoi diagrams as described in [2] rather than trained by back propagation algorithm. Due to its parallel processing capability the propagation delay is only as long as the time needed for the signal to pass through the layers of logic gates implementing the FFANN (up to a few hundred nanoseconds).

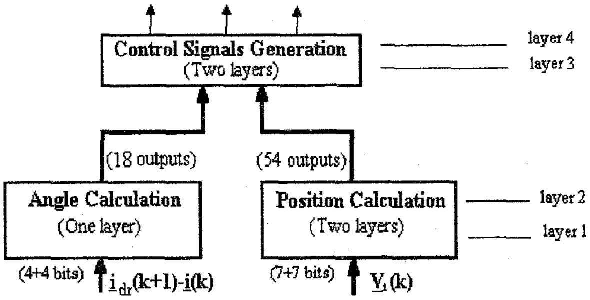

Fig. 2 - The architecture of the neural network

As illustrated in Fig. 2 the novel architecture is defined by two tiers containing three interconnected subnetworks. It cannot be entirely assimilated to a classical feed-forward network because the 'Angle Calculation Subnetwork' contains only one layer breaking thereby the symmetry of the neural structure. The first subnetwork determines the position of the non-inductive voltage space vector $\underline{V}_I(k)$ in the complex plane, the second determines the direction along which the current space vector has to move in the next sampling period following the information provided by the direction space vector $\Delta\underline{i}=\underline{i}_{dr}\ (k+1)-\underline{i}(k)$, while the third subnetwork generates the three bits which describe one of the six possible output voltages of the PWM inverter.

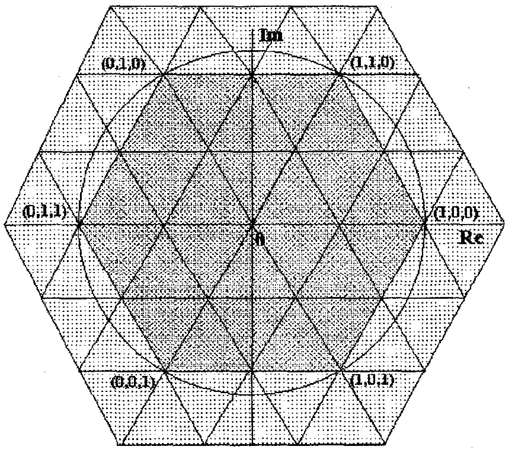

Fig. 3 - The partition of the interest area into Voronoi cells

The operation accuracy of the PWM controller has been limited to an acceptable value in order to keep the hardware implementation complexity to a low level. As a result, the adopted implementation solution is described as follows: the angle calculating subnetwork will determine the angle designating the desired movement direction with a precision of $\pm 5^0$ while the position calculating subnetwork divides the complex plane into 54 triangular regions (Voronoi cells) as shown in Fig. 3 and calculates in which of these regions the vertex of $\underline{V}_1(k)$ is. The decision as to what control signal set to be provided to the transistors in the PWM inverter will be taken as if the vertex of $\underline{V}_1(k)$ is located in the center of the corresponding cell. MATLAB simulations proved that the small errors introduced by the imperfect PWM controller do not significantly affect the overall operation of the system.

V. The Inductance Estimator

If the inductance L in the equivalent circuit is exactly known then it is always possible to predict the direction of the space vector $\Delta\underline{i}=\underline{i}(k+1)-\underline{i}(k)$ as for prediction purposes equation (4) can be rewritten as:

$$\underline{V}_1(k) = \underline{u}(k+1) - \frac{L}{T_s}\Delta\underline{i} \qquad (7)$$

As a result, the direction of $\Delta\underline{i}$ is the same with the direction of $\underline{V}_1(k)$-$\underline{u}(k+1)$. In case L is not correctly known, the $\underline{V}_1(k)$ is affected by the error too, so that there will be a difference between the predicted direction of $\Delta\underline{i}$ and the real one. Due to reasons of simplicity, the inductance estimator only operates when the PWM inverter voltage corresponds to vertex (0,1,1) of the output voltage hexagon (see Fig. 4), that is when $\underline{u}$=-U_{dc} (where U_{dc} is the DC voltage at the inverter input). In this case the estimator calculates the theoretical direction vector (*td*) and compares it to the real direction of $\Delta\underline{i}$. If the corresponding angles α and β are not equal it means that the assumed value of L has to be corrected. The correction process is incremental: every time α does not equal β a small value ΔL is added or subtracted from the current L value until the right value is found.

Whether ΔL is added or subtracted depends both on the relationship between angles α and β and on the last value of the inverter output voltage $\underline{u}(k-1)$. A simplified version of the estimator may avoid taking into account the possibility when the imaginary part is zero. In this case the correction will only be performed when the imaginary part of vector $\underline{u}(k-1)$ is not null. This simplified version is the one used in this paper as it has the advantage of a simpler hardware implementation. There are two possible cases to consider, depending on the sign of the imaginary part of $\underline{u}(k-1)$:

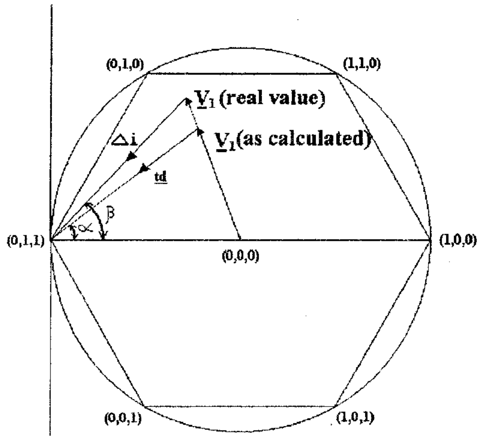
(0,1,0)
(1,1,0)
$\underline{V}_1$ (real value)
Δi
$\underline{V}_1$(as calculated)
td
β
α
(0,1,1)
(1,0,0)
(0,0,0)
(0,0,1)
(1,0,1)

Fig. 4 - Graphical Representation of the induction estimator operating principle

1) The imaginary part of $\underline{u}(k-1)$ is positive. If β is greater than α (situation illustrated in Fig. 4) then the imaginary part of $\underline{V}_1$ was underestimated. But $\underline{V}_1$ was calculated according to equation (7) and the imaginary part of $\Delta\underline{i}$ was positive due to the sign of the imaginary part of $\underline{u}(k-1)$. Therefore the value considered for L/T_s is too large and has to be decreased. As a consequence ΔL will be subtracted from the previous value of L. Conversely, if β is smaller than α the same ΔL will be added to L.

2) The imaginary part of $\underline{u}(k-1)$ is negative. The situation is reversed. All the signs are changed and ΔL will be added to L for β greater than α and subtracted for β less than α.

VI. Implementation Principles

The hardware structure corresponding to a neural network is always intricate which makes it impossible to be designed using the classical schematic capture approach. Artificial Neural Networks have already been implemented using field programmable devices, taking advantage of their potential for rapid prototyping [3], and density enhancement [4]. The design approach adopted here minimises the model-to-hardware conversion algorithm while still achieving compact hardware implementation.

Thus a set of three C++ computer programs has been developed in order to automatically generate a VHDL file describing the digital implementation of any neural network. The implementation is based on elementary logic gates only. The input data file for the first program contains the mathematical matrix description of the network to be converted. It generates a netlist description of the digital architecture which is then optimized by the second program to eliminate all the redundant structures. The third program converts the netlist description into a VHDL file.

The conversion from the matrix description to the netlist takes place in several stages. In the first stage the analogue bipolar inputs of the neural network are digitized. The inputs of the neurons in the first layer are split into 'n' other inputs which can only receive two different signals: '+1' and '-1'. The values of the new weights are calculated according to equation (8).

$$w_{ik} = \frac{2^{k-1}}{2^{n}-1} \cdot w_{i} \qquad (8)$$

The bipolar neurons are transformed into their unipolar counterparts in the second stage with '1' and '0' only as input and output signals. All the negative input weights are transformed into positive weights by using inverter gates for the corresponding input signals (see Fig. 5). Finally the input weights are arranged in descending order. All the possible input signal combinations which can trigger the neuron output are systematically analyzed and a logic gate structure which behave accordingly is generated.

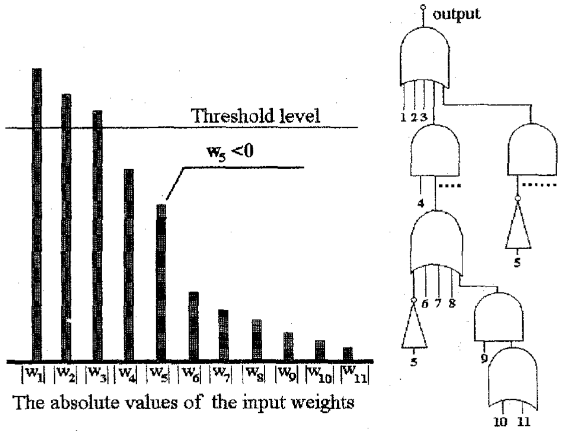
output
Threshold level
$w_5 < 0$
1 2 3
4
5
6 7 8
9
10 11
$|w_1|$ $|w_2|$ $|w_3|$ $|w_4|$ $|w_5|$ $|w_6|$ $|w_7|$ $|w_8|$ $|w_9|$ $|w_{10}|$ $|w_{11}|$
The absolute values of the input weights

Fig. 5 - The Neural Network-Logic Gate Structure Conversion Principle

Some input weights may be larger than the threshold level in which case each corresponding input signal can separately trigger the neuron output (see Fig. 5). Other input signals can only do so if they occur simultaneously and the sum of the corresponding weights surpasses the threshold value.

Fig. 6 illustrates the data flow for the conversion process of the PWM generator neural network. ANGLES, REGIONS and CTRL are the programs which generate the matrix description of the neural subnetworks corresponding to the user requirements (number of Voronoi cells, precision in angle calculation, number of bits for the digitized input values). The others are the three universal conversion programs described above.

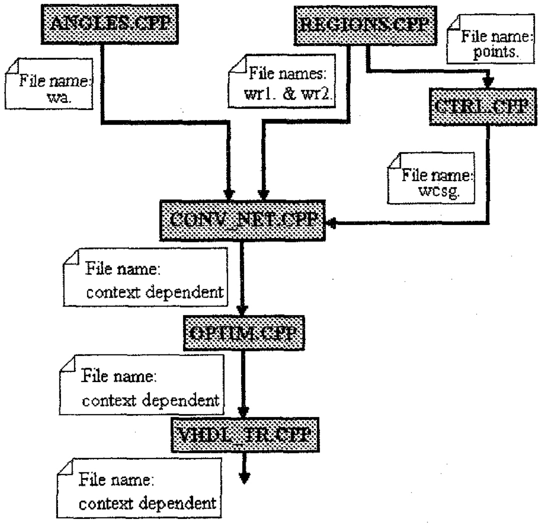

Fig. 6 - The Data Flow Diagram for the Conversion Process of the Neural PWM Generator

VII. Simulation Results and Conclusions

Digital simulations in VHDL proved the correct operation of the new digital controller. Analogue MATLAB simulations were performed as well to check the performance of the new ASIC in controlling a PWM inverter feeding an 11.1 kW induction motor. The current controller proved able to calculate the correct value of the inductance in the equivalent circuit alongside with good results in controlling the currents across the motor windings. The stepwise inductance estimation incrementing towards the final correct value is illustrated in Fig. 7. This is reflected in the decreasing ripples of the stator current vector represented as separate real and imaginary parts in Fig. 8, thereby providing better operation conditions for the drive system. The figure also illustrates torque and speed variation in time during transient operation.

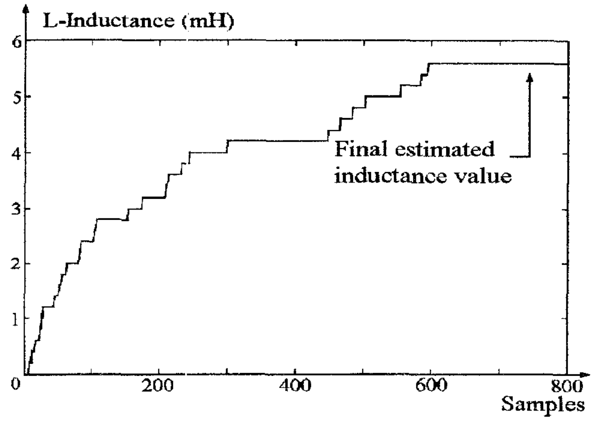

Fig. 7 - Estimated induction value

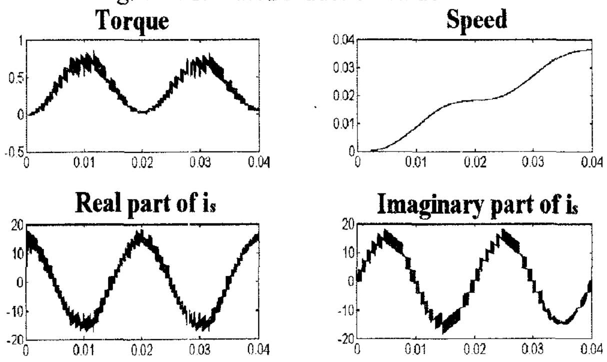

Fig. 8 - Induction motor behavior (transient operation)

The adopted PWM current control strategy is more complex but more flexible and accurate than classical strategies using hysteresis current controllers. It ensures a high speed current response in transient operation. Microprocessor software control with a high mathematical content has been replaced by a fast ASIC implementation easily allowing the generation of optimal PWM signals with switching frequencies higher than 10 kHz. This solution is a simple, reliable, compact and cost effective alternative to other implementations. The presented current control scheme is independent of the load constants and is opened to further developments as the ASIC calculates the $\underline{V}_1$ space vector containing information which can be input into additional circuits implementing complex control strategies for a large variety of power systems.

References

[1] A. Nabae, S. Ogasawara, H. Akagi, "A new control scheme for current controlled PWM inverters", IEEE Trans. Ind. Appl. vol IA-22, no. 4, pp. 697-701, July/August 1986.

[2] N.K. Bose and A.K. Garga "Neural Network Design Using Voronoi Diagrams" IEEE Transactions on Neural Networks vol. 4, no. 5, September 1993 pp. 778-787.

[3] Stephen L. Bade and Brad L. Hutchings, "FPGA-Based Stochastic Neural Networks Implementation", IEEE Workshop on FPGAs for Custom Computing Machines, Napa, 10-13 April 1994.

[4] Joseph G. Eldredge and Brad L. Hutchings, "Density Enhancement of a Neural Network Using FPGAs and Run-Time Reconfiguration", IEEE Workshop on FPGAs for Custom Computing Machines, Napa, 10-13 April, 1994.

[5] G. W. Irwin, K. Warwick, K. J. Hunt "Neural Network Applications in Control" Institution of Electrical Engineers, London 1995.

[6] Vas P.; Li J. Stronach "Artificial Neural Network-Based Control of Electromechanical Systems". in Proceedings of 4th European Conference on Control IEE. Coventry. 1994

[7] Heinz Willi van der Broeck, Hans Cristoph Skudelny, Georg Viktor Stanke "Analysis and Realisation of a Pulsewidth Modulator Based on Voltage Space Vectors" IEEE.