Implementation of an enhanced neural learning algorithm for rotor resistance estimation for solar fed IM drive

Chitra A.1*, Razia Sultana W.2, S. Sreejith3, and J. Vanishree4

1,2,3,4 Associate Professor, School of Electrical Engineering, VIT University, Vellore - 632014, India.

Orcid Id: 1*0000-0001-5820-1305, 20000-0003-4046-7759 30000-0002-2008-6388, 40000-0002-0065-6929

Abstract
In induction motors with Indirect field oriented control (IFOC) scheme, the rotor resistance (Rr) variation results in loss of dynamic performance. Hence the estimation of Rr is essential for high performance vector controlled drives. Neural based estimators are now receiving active contemplation as it has number of advantages over conventional techniques. The learning algorithm of the neural network determines its estimation speed, accuracy of estimation, and viability for digital implementation. In this work an enhanced learning algorithm is proposed with superior performance involving less computational complexity. Also the proposed neural based estimator has been implemented in TMS320F2812 digital signal processor (DSP) and Spartan 3E field programmable gate array (FPGA) and the results are presented. Multilevel inverter (MLI) has been used as it is an attractive alternative over conventional two level inverters in medium and high power applications. The enhanced performance of the MLI fed green drive with the proposed Rr estimator is presented. The impact of estimation time and estimation accuracy of the two implementation schemes on the MLI fed drive has been explored. The drive performance with DSP and FPGA based estimator has been analysed for a wide range of variations in Rr.

Keywords: Neural Networks, Back Propagation, Rotor resistance Estimators, Vector Control, DSP, FPGA.

INTRODUCTION
The IFOC scheme is highly sensitive to parameter variations [1, 2]. Among all the parameters Rr is of prime importance, as it is employed in the slip calculation [3, 4]. Unfortunately the Rr is subjected to vary widely with the operational factors of the machine such as temperature, frequency and saturation level of the machine. If the controller is not updated with the changing Rr, then it will lead to poor dynamic performance. Many schemes have been developed for Rr estimation such as model reference adaptive control, signal injection method, extended kalman filter and recursive least square method [5-10]. To provide a continuous on-line estimation neural network learning based model reference adaptive system (NL-MRAS) is more attractive [10-12].

The learning algorithm used for MRAS adaptation has an effect on issues such as learning speed, stability, weight convergence, accuracy of estimation, and speed of tracking and ease of implementation. These issues remain as areas of research and comparison of training algorithm is therefore essential to arrive at the appropriate learning method [13]. Initially, the NL-MRAS based estimator is trained with the conventional back propagation (BP) and BP with variable learning rate (BP with VLR) algorithms. An enhanced neural learning algorithm namely constraint based BP proposed in this paper.

Digital implementation of the neural learning algorithm is possible either with a general purpose DSP or FPGA. DSP is preferred, as it is general purpose hardware at low cost and also involves simple implementation. The proposed neural based rotor resistance estimator is first implemented in a low cost DSP. As the estimation speed of this estimator is not faster enough to improve the dynamic performance of the drive, the implementation is carried out in FPGA. The implementation schemes have been compared in terms of estimation time and estimation accuracy.

Induction motor drives with standard two level inverters have been investigated for numerous applications. The IFOC scheme has become standard industrial solution for better dynamic performance of these machines [14]. MLIs have gained momentum for high power motor drives and automotive drive systems [15-19]. The advantages of using the MLI over the two level counter parts are the higher voltage operation ability, near sinusoidal outputs, smaller or even no output filter and reduced switching losses, as they can be operated with lesser switching frequency. Common MLI topologies are Neural point clamped (NPC), Cascade H-Bridge (CHB), and flying capacitor type [20]. Among the three topologies, the CHBMLI is more appropriate for renewable energy systems (RES), as it requires more number of isolated DC sources. There are many modulation techniques available such as sine pulse width modulation (SPWM), space
vector pulse width modulation (SVPWM) [21] and the widely
used is multicarrier SPWM scheme [22, 23].

In this work the high performance intelligent induction motor
drive with IFOC scheme has been developed in Matlab /
Simulink. The neural learning algorithm is coded as an m-file
with sample time of 0.1ms. The conventional VSI fed drive
and the seven level CHBMLI fed drive performance
comparison is presented. Also the enhanced performance of
the drive scheme with estimator is analyzed.

STRUCTURE OF NL-MRAS BASED ROTOR
RESISTANCE ESTIMATOR

This section presents an NL-MRAS based rotor resistance
estimator for the induction motor drive. A simple two layered
feed forward neural network trained by BP technique is used
for the real time adaptive estimation. Two models of the state
variable estimation are used, one provides the actual induction
motor output and the other gives the neural model output. The
total error between the desired and actual state variables is
then back propagated.

The rotor flux of the induction motor is estimated with a
classical voltage model. The flux estimated with this voltage
model will be correct irrespective of the variations in Rr since
voltage model equations are independent of Rr and this
provides the reference state variable. The other state model is
the adaptive model based on current model equations which is
dependent on Rr and this provides the estimated state variable.
The error between the reference flux and estimated flux is
used by the adaptive mechanism to vary Rr such that the
error tends to zero. Equation based on stator voltages and
currents is called as voltage model equation and is given in
(1). The structure of NL-MRAS based Rr estimator is shown in
Fig. 1a.

Equations based on stator currents and rotor speed is called as
current model equations. The discrete current model equations
are given as in (2) and (3)

\[
\lambda_{dr}(k) = W_{1} \lambda_{dr}(k-1) - W_{2} \lambda_{dr}(k-1) + W_{3} i_{ds}(k-1)
\]

\[
\lambda_{qr}(k) = W_{1} \lambda_{qr}(k-1) + W_{2} \lambda_{dr}(k-1) + W_{3} i_{qs}(k-1)
\]

The neural model represented by equation (2) and (3) is
shown in Fig. 1b, where W1, W2, W3 represent the weights of
the two layer neural network used to estimate rotor resistance
given in (4). The neural network (NN) is a simple two layered
network with linear activation functions.

\[
W_{1} = \frac{1}{T_{r}} \int_{0}^{T_{r}} v_{ds} R_{s} + \sigma L_{s} i_{ds} dt
\]

\[
W_{2} = \sigma_{r} T_{r}
\]

\[
W_{3} = \frac{T_{L} L_{m}}{T_{r}}
\]

Here T is the sampling period, Tr is the rotor time constant,
or is electrical rotor angular velocity, \(\lambda_{dr}\) and \(\lambda_{qr}\) are d-axis
and q-axis rotor fluxes, Ids and Iqs are d-axis and q-axis stator
currents and \(\sigma\) is called leakage coefficient. Among the three
weights W2 is already known and W1 and W3 need to be
updated. The weights between neurons, W1and W3 are
updated, so as to minimize the energy function E. The error
and the energy function are given in (5) and (6).

\[
e_{d}(k) = \lambda_{dr}(k) - \lambda_{dr}(k)_{ref} - \lambda_{dr}(k)_{est}
\]

\[
e_{q}(k) = \lambda_{qr}(k) - \lambda_{qr}(k)_{ref} - \lambda_{qr}(k)_{est}
\]
The change in weight updates have been derived for all the neural algorithms and the algorithms are coded as m-File in Matlab. The voltage model equations are implemented in simulink model file. The neural model estimator is updated at a sampling frequency of 10 kHz.

The update equations for both the methods are as given in literature [24]. Due to the limitations of BP with momentum algorithm, BP with Variable Learning Rate (VLR) is introduced. The weight W2 is fixed and the weights W1 and W3 are updated in such a way that the energy function E is minimized. To highlight the performance of the proposed algorithm the neural based estimator has been trained with various learning strategies and the results are compared in terms of their estimation time, accuracy of estimation and complexity. The learning strategies used for comparison are

(i) BP with momentum
(ii) BP with variable learning rate
(iii) Proposed Constrained based BP

PROPOSED LEARNING ALGORITHM AND PERFORMANCE COMPARISON

This proposed algorithm is a simple modification to the BP with momentum method. In the conventional method both the weights W1 and W3 are updated. Here the error is used to update the weight W3 and the weight W1 is found from the value of W3. From the equations (1) to (6) the update laws for the proposed constraint based back propagation algorithm are derived and presented. The error and the energy function are given in (5) and (6). The change in weight is given by equations (7) and (8).

\[
\Delta W_3(k) = -\alpha \frac{\partial E}{\partial W_3} 
\]

(7)

\[
\Delta W_3(k) = -\alpha \frac{\partial E}{\partial \lambda_{est}^r} \times \frac{\lambda_{est}^r}{\partial W_3}
\]

(8)

\[
\frac{\partial E}{\partial \lambda_{est}^r(k)} = -\varepsilon_r(k)
\]

(9)

\[
\varepsilon_r(k) = \left[\varepsilon_d(k), \varepsilon_q(k)\right]^T
\]

(10)

\[
\varepsilon_r(k) = \lambda_{est}^r(k) - \lambda_r^est(k)
\]

(11)

Differentiating partially equations (2) & (3) with respect to \( \partial W_3 \)

\[
\frac{\partial \lambda_{est}^r}{\partial W_3} = \left[i_d \hspace{1mm} (k-1) \ i_q \hspace{1mm} (k-1)\right]^T
\]

(12)

Differentiating partially equation (6), with respect to \( \frac{\partial \lambda_{est}^r}{\partial \lambda_r^est} \r 

Substituting the above results in equation (8), we get,

\[
\Delta W_3(k) = \alpha \left[\varepsilon_d(k) \ i_d \hspace{1mm} (k-1) + \varepsilon_q(k) \ i_q \hspace{1mm} (k-1)\right]^T
\]

(13)

The weight adjustments require choice of learning rate (\( \alpha \)). If \( \alpha \) is large, it will lead to oscillations in the output. To overcome this difficulty a momentum (\( \eta \)) added to the equation (13), which takes the past (\( k-1 \)) weight changes on the \( (k)^{th} \) weight. This ensures accelerated convergence of the algorithm. Thus the current weight adjustment \( W_3(k) \) is described as

\[
W_3(k) = W_3(k-1) + \Delta W_3(k) + \eta \Delta W_3(k-1)
\]

(14)

From the equation (4) \( W_1 \) can be found as below

\[
W_1(k) = 1 - \left(\frac{W_3(k) }{L_m}\right)
\]

(15)

The rotor resistance can be calculated either from \( W_1 \) or \( W_3 \) from the equations given in (16) and

\[
R_r = \left[L_r W_3 \over L_m T\right] = \frac{L_r}{T} (1-W_1)
\]

(16)

The proposed algorithm is used for \( R_r \) estimation and compared with conventional algorithms.

A. Performance comparison of the conventional and proposed learning algorithms

The induction motor has been modelled using T-model equations in Matlab simulink to incorporate the variations in rotor resistance as in the practical case [25]. The simulation results for the rotor resistance estimation using proposed NL-MRAS based \( R_r \) estimator is studied for the following cases.

i. With 40% step change in Rotor Resistance
ii. With 40% trapezoidal change in Rotor Resistance

All the variants of BP such as BP with momentum factor and BP with VLR have been used for training the NL-MRAS based rotor resistance estimator. The results are shown in Table I. From the Table it can be observed that the proposed algorithm performs better than the other two algorithms.
Table I  Results of NL-MRAS based Rr estimator with conventional and proposed learning algorithms

<table>
<thead>
<tr>
<th>Learning Algorithm</th>
<th>Nature of change in Rr</th>
<th>Actual Rr (Ω)</th>
<th>Estimated Rr (Ω)</th>
<th>Error %</th>
<th>Time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Back propagation</td>
<td>Step</td>
<td>0.049</td>
<td>0.048</td>
<td>0.1</td>
<td>0.03</td>
</tr>
<tr>
<td>with Momentum</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BP with VLR</td>
<td>Step</td>
<td>0.049</td>
<td>0.048</td>
<td>0.1</td>
<td>0.03</td>
</tr>
<tr>
<td>Proposed Constraint</td>
<td>Step</td>
<td>0.049</td>
<td>0.048</td>
<td>0.1</td>
<td>0.03</td>
</tr>
<tr>
<td>Back propagation</td>
<td>Trapesoidal</td>
<td>0.065</td>
<td>0.060</td>
<td>0.1</td>
<td>0.03</td>
</tr>
<tr>
<td>with Momentum</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Among all these methods the proposed constraint based BP is concluded to be the most suitable learning algorithm for neural based rotor resistance estimator. Though the BP with VLR provides comparable performance with the proposed method, the computation complexity involved in VLR increases the time taken for estimation. Further in the proposed algorithm the number of equations gets reduced from 5 to 3 which simplify the process. The proposed learning algorithm exhibits good tracking profile with less estimation time and error. Also it has the added advantage that the computational simplicity allows easy implementation in digital processors.

DIGITAL IMPLEMENTATION OF THE PROPOSED NL-MRAS BASED RR ESTIMATOR

Digital implementation is desired because of higher accuracy, high repeatability, low noise sensitivity, better testability, higher flexibility, and compatibility with other types of pre-processors. Digital implementation is possible either with a general purpose Digital Signal Processor (DSP) or with a Field Programmable Gate Array (FPGA). DSP is preferred, as it is general purpose hardware at low cost and also involves simple implementation. To start with the digital implementation of the online neural based rotor resistance estimator is carried out with a low cost Digital Signal Processor (DSP) F2812 using code composer studio (CCS). The proposed algorithm is implemented, as it involves less complexity and better accuracy compared with the conventional algorithms.

A. Digital implementation using DSP

The proposed learning algorithm for the neural estimator namely constraint based back propagation has been coded in code composer studio. A 32 bit DSP from Texas instruments TMS320F2812 is used for the implementation of neural based rotor resistance estimator and the result is shown in Fig. 2.

![Fig. 2. DSP result for proposed neural based Rr estimator](image)

The estimation time can be calculated from the above result as follows

Time taken for one iteration = Number of cycles*(1/f)

Where,

The processor frequency ‘f’ = 150 MHz.

Time taken for one iteration = 157149*(1/150*106)

= 1.179ms

Total time taken for estimation = Time taken for one iteration * No. of iterations

= 1.179ms * 1183

Time for estimation in DSP = 1.4sec

To improve the speed of estimation a dedicated hardware on chip is realized using FPGA.

B. Digital implementation using FPGA

The Field Programmable Gate Array (FPGA) programmable logic combines the best of both DSP and ASIC technology. FPGAs are programmable and changeable (like DSPs); the designers can make changes quickly, without additional cost and time of ASIC design. On the other hand the FPGA implemented algorithms allow parallel processing and hence are faster than a general purpose DSP. Also the re-configurability of Field Programmable Gate Array (FPGA) has improved the flexibility in the digital system design [26]. Greater density and high speed FPGAs have enhanced the ability to realize special purpose processors for high-end applications.

In digital implementation of on-line rotor resistance estimator an important parameter is the precision of the various blocks. Precision or word length decides the output resolution. Higher resolution means higher accuracy but leads to larger resource requirement, which results in higher cost. Efficient hardware implementation can be achieved by identifying the optimal resolution suitable for the application. To identify the optimal
bit precision the analysis is carried out for various bit precisions using MATLAB and the results are tabulated in Fig.3.

**Figure 3:** Results for bit precision with various bit precisions

The results for various bit precisions are shown in Table II. The bit precision for the variables is chosen to compromise between accuracy and cost. From the results, it can be observed that 16 bit precision achieves good accuracy, and increasing the number of bits further does not appreciably enhance the performance. Hence it is concluded that 16 bit precision is most optimal for FPGA implementation.

**Table II**  Results for various bit precisions

<table>
<thead>
<tr>
<th>Bit Precision</th>
<th>Actual R (Ohm)</th>
<th>Estimated R (Ohm)</th>
<th>Error (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>8.519</td>
<td>8.115</td>
<td>4.7</td>
</tr>
<tr>
<td>12</td>
<td>8.519</td>
<td>8.344</td>
<td>2.05</td>
</tr>
<tr>
<td>16</td>
<td>8.519</td>
<td>8.472</td>
<td>0.511</td>
</tr>
<tr>
<td>32</td>
<td>8.519</td>
<td>8.475</td>
<td>0.516</td>
</tr>
</tbody>
</table>

The inputs and weights are represented as bits. An 8-bit whole number, 16-bit fractional number along with a sign bit represents each input and output for the network. The input data is a signed 25-bit number (one sign bit + 8 whole bits + 16 fractional bits). The weights are also represented with the same bit precision. The multiplication is a 24 bit * 24 bit multiplication whose result is stored as 48 bits and a sign bit. The output X is obtained with 49 bits (one sign bit + 16 whole bit number + 32 bit fraction). Here the output is truncated to 25 bits (one sign bit + 8 whole bits + 16 fractional bits). The activation function f(x) is linear function so the result of forward pass is the truncated value of X. Appropriate bit precision and FPGA architecture is chosen. The simplified structure of NN considered for FPGA implementation is shown in Fig. 4. X1, X2, and X3 are the inputs to the FPGA block and they are given by equation (17).

$$X_1 = \begin{bmatrix} \lambda_{est}(k-1) \\ \lambda_{d}(k-1) \end{bmatrix}^T$$

$$X_2 = \begin{bmatrix} \lambda_{q}(k-1) \\ \lambda_{est}(k-1) \end{bmatrix}^T$$

$$X_3 = I_{d}(k-1) I_{q}(k-1)$$

(17)

The inputs and weights are represented as bits. The input and output data consists of an 8-bit whole number, 16-bit fractional number along with a sign bit. Three sets of input and output is shown in Table III.

**Table III:** Three sets of input and output data for FPGA implementation

<table>
<thead>
<tr>
<th>Value of R</th>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th>Output W3</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.085</td>
<td>[0.9795 0.9795]</td>
<td>[-0.9795 0.9795]</td>
<td>[2.72 2.72]</td>
<td>5.712e4</td>
</tr>
<tr>
<td>8.519</td>
<td>[0.9798 0.9798]</td>
<td>[-0.9798 0.9798]</td>
<td>[2.71 2.71]</td>
<td>7.994e4</td>
</tr>
<tr>
<td>10.95</td>
<td>[0.9814 0.9814]</td>
<td>[-0.9814 0.9814]</td>
<td>[2.7 2.7]</td>
<td>0.000303</td>
</tr>
</tbody>
</table>

The FPGA implementation of on-line rotor resistance estimator is divided into various sub blocks and these blocks are implemented individually first and then they are integrated to form the entire estimator as shown in Fig. 5a. The Rotor Resistance Estimator consists of a forward pass and a weight updating block. The Forward block calculates the value of neural model motor rotor flux and the weight updating block calculates the value of weight W3(k) corresponding to resistance estimation.
The results obtained for $R_r = 6.085$ ohms is shown below in Fig. 5b. There are various steps involved in FPGA design flow. Among them the implementation is an important aspect. Implementation means conversion of the logical design file format created during design entry into a physical file format for a specific Xilinx architecture. The implementation in FPGA is specified in term of the number of slices (or cells). A slice consists of two look up tables (LUT’s) and two registers. For the implementation of on-line rotor resistance estimator the number of slices occupied gives the resource requirement. The time taken to process the input data and obtain the result gives the timing requirement. The entire network is simulated, the resource requirements and timing requirement is obtained from the HDL synthesis report. The timing report and the device utilization summary for the selected device are presented as in Table IV and V.

### Table IV  Timing report

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum period</td>
<td>20.03ns</td>
</tr>
<tr>
<td>Maximum Frequency</td>
<td>49.925MHz</td>
</tr>
<tr>
<td>Minimum input arrival time before clock</td>
<td>12.476ns</td>
</tr>
<tr>
<td>Maximum output required time after clock</td>
<td>7.51ns</td>
</tr>
</tbody>
</table>

### Table V  Device utilization summary

<table>
<thead>
<tr>
<th>Component</th>
<th>Value</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Slices</td>
<td>655 out of 4800</td>
<td>13%</td>
</tr>
<tr>
<td>Number of Slice Flip-Flops</td>
<td>469 out of 9600</td>
<td>4%</td>
</tr>
<tr>
<td>Number of 4 input LUTs</td>
<td>1202 out of 9600</td>
<td>12%</td>
</tr>
<tr>
<td>Number of bonded IOBs</td>
<td>51 out of 170</td>
<td>30%</td>
</tr>
<tr>
<td>Number of GCLKs</td>
<td>1 out of 4</td>
<td>25%</td>
</tr>
</tbody>
</table>

The estimation time can be calculated from the above result as follows

- Total time taken for estimation = Time taken for one iteration $\times$ No. of iterations
- Time taken for estimation in FPGA = $8.9$ ms

The proposed neural based rotor resistance estimator is first implemented in a low cost Digital Signal Processor. As the estimation speed of this estimator is not faster enough to improve the dynamic performance of the drive, the implementation is carried out in FPGA. The results of the neural based estimator under the two implementation schemes are consolidated in terms of their speed and accuracy in the Table VI.

### Table VI  Performance Comparison of FPGA And DSP Based Implementations

<table>
<thead>
<tr>
<th>Type of Processor</th>
<th>Estimation error (%)</th>
<th>Estimation time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSP</td>
<td>0.38</td>
<td>1.4</td>
</tr>
<tr>
<td>FPGA</td>
<td>0.39</td>
<td>0.0089</td>
</tr>
</tbody>
</table>

Though the estimation error is almost same with both implementation methods the time taken for estimation is large in the DSP implementation due its sequential processing nature. The faster tracking in the FPGA based implementation is because of its parallel computing units. Hence the FPGA implementation is found to be suitable for the proposed neural estimator.

### ENHANCED PERFORMANCE OF THE VECTOR CONTROLLED DRIVE WITH PROPOSED NL-MRAS BASED RR ESTIMATOR

The IFOC drives have been used for numerous industrial applications. Instead of observing the machine flux, the correct field orientation control is obtained by a feed forward slip control in an IFOC based drive. With the use of a shaft encoder the indirect rotor flux oriented control (IRFOC) can be accomplished with low cost and high performance. The block diagram of the entire drive scheme is as shown in Fig. 6a. High performance can be achieved by using MLI [27, 28]. Also the conventional two level inverter fed drive and 7 level CHBMLI fed drive has been compared and it is consolidated in Fig. 6b [29-31]. It is observed from the results that the MLI fed drive outperforms the conventional inverter fed drive. The
performance is analyzed in terms of efficiency, torque ripples, voltage and current THD. Thus, for further analysis with the proposed NL-MRAS based $R_r$ estimator, the MLI fed drive has been considered.

![Basic block diagram of green drive](image1)

**Figure 6:**
(a) Basic block diagram of green drive
(b) Block diagram of IFOC scheme with NL-MRAS $R_r$ estimator
(c) Performance comparison of two level and 7 level CHBMLI fed induction motor drive

The results for the drive scheme with the proposed DSP and FPGA based online NL-MRAS $R_r$ estimator is explored in Fig. 7. For both the cases the drive has been operating with a reference speed of 100rad/s, rotor flux reference of 0.9 wb, and with a dynamic load torque of 6Nm to 5.5Nm at 2 sec when the rotor resistance of the machine is changed from 6.085ohm to 10.95 ohm at 2.5 seconds and is shown in Fig. 8a&8b.

![Speed, torque and flux response of IFOC drive with DSP based $R_r$ estimator](image2)

**Figure 7.**
(a) Speed, torque and flux response of IFOC drive with DSP based $R_r$ estimator
(b) Speed, torque and flux response of IFOC drive with FPGA based $R_r$ estimator
(c) ids and iqs of IFOC drive with DSP based $R_r$ estimator
(d) ids and iqs of IFOC drive with FPGA based $R_r$ estimator

In previous section it has been given that the time for estimation is 1.4sec in DSP implementation and 0.0089sec for the FPGA based implementation. In the case of the drive with DSP implemented NL-MRAS $R_r$ estimator the responses are not satisfactory due to the delayed estimation and are shown in Fig. 7a. It shows that the actual motor torque deviates from the command torque for 1.4sec. Also the flux response displayed in Fig. 7a portrays that rotor flux rises by 32.2% from its nominal value for 1.4sec. Though theoretically the flux linkages can increase up to 20% but practically it is not possible due to magnetic saturation effects. Thus with DSP based $R_r$ estimator the instantaneous torque and flux control is also lost for a time period of 1.4sec as in Fig. 7a. Also another adverse effect on the drive performance due rotor resistance change is that the decoupled control is lost which is the most desirable feature of the vector controlled drive. This is evident from the Fig. 7c.
From the results it can be observed that the performance of the drive is satisfactory with FPGA implemented NL-MRAS Rr estimator. The actual torque of the motor tracks the reference torque and the rotor flux tracks the command value which are clear from the Fig.8b. Also it retains the decoupled control as in Fig.7d.

Table VII. Flux response for different variations in Rr with DSP and FPGA based estimator

<table>
<thead>
<tr>
<th>Rr change</th>
<th>With DSP based estimator</th>
<th>With FPGA based estimator</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Maximum deviation in $\lambda_r$ (Wb)</td>
<td>Maximum deviation in $\lambda_f$ (%)</td>
</tr>
<tr>
<td>40% step change</td>
<td>1.06</td>
<td>22.77</td>
</tr>
<tr>
<td>80% step change</td>
<td>1.19</td>
<td>32.2</td>
</tr>
<tr>
<td>100% step change</td>
<td>1.21</td>
<td>34.4</td>
</tr>
<tr>
<td>200% step change</td>
<td>1.60</td>
<td>77.7</td>
</tr>
</tbody>
</table>

Table VIII. Torque response for different variations in Rr with DSP and FPGA based estimator

<table>
<thead>
<tr>
<th>Rr change</th>
<th>With DSP based estimator</th>
<th>With FPGA based estimator</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Maximum deviation in $T_s$ (Wb)</td>
<td>Maximum deviation in $T_s$ (%)</td>
</tr>
<tr>
<td>40% step change</td>
<td>6.23</td>
<td>24.6</td>
</tr>
<tr>
<td>80% step change</td>
<td>7.12</td>
<td>42.4</td>
</tr>
<tr>
<td>100% step change</td>
<td>7.52</td>
<td>56.4</td>
</tr>
<tr>
<td>200% step change</td>
<td>8.1</td>
<td>62</td>
</tr>
</tbody>
</table>

The ratings of the induction motor used for simulation studies are given in the appendix. For wide range of variations in Rr, the flux and the torque responses are tabulated in Table VII and VIII respectively with DSP and FPGA based Rr estimator.

CONCLUSION

The new enhanced neural learning strategy for the rotor resistance estimator has been proposed in this paper. It is compared with conventional learning strategies in terms of time taken for estimation, accuracy of estimation and complexity and the results are presented. The proposed constraint based BP is shown to be the most appropriate learning strategy for the neural based rotor resistance estimator as it is computationally less rigorous resulting in lower estimation time.

The proposed Rr estimator is implemented using DSP and FPGA. The DSP implementation is carried out using F2812 processor. The coding is done using code composer studio and the time taken for estimation is found to be 1.4s. To further improve the speed of estimation the estimator is implemented using FPGA. The bit precision in FPGA is important as higher bit precision leads to higher accuracy but also higher cost. To identify the optimal bit precision the analysis is carried out and optimal bit precision is arrived to be 16 bits. In FPGA the time taken for implementation is obtained to be 0.0089s. Both DSP and FPGA estimate Rr with good accuracy but the FPGA is 157 times faster than the DSP. As Rr varies online the speed of estimation is crucial for the drive performance.

System comprising of induction motor, inverter, Rr estimator is built for IFOC. The performance of MLI is superior to two level inverters. In this paper a 7-level CHBMLI fed IFOC drive scheme has been studied and shown to be suitable inverter for IFOC when used with RES.

The performance of the 7 level CHBMLI fed IFOC drive scheme has been studied with and without Rr estimator. The performance is obtained for the drive scheme with DSP and FPGA implemented Rr estimator. As the estimation time for DSP is larger it is observed that the instantaneous torque control is lost and the rotor flux increases beyond the command value resulting in loss of the decoupled control. For large changes the system may become unstable. When Rr estimation is carried out using FPGA, it is observed that both the torque and flux track the reference retaining the decoupled control. Hence FPGA is concluded to be the most suitable digital hardware for Rr estimation. The performance of the drive is really enhanced with FPGA based Rr estimator. To sum up this work will form a basis for the embedded control of electric drives.

REFERENCES


