JTE
HCMUTE
ISSN: 1859-1272

TẠP CHÍ KHOA HỌC GIÁO DỤC KỸ THUẬT
Trường Đại học Sư phạm Kỹ thuật Thành phố Hồ Chí Minh
Website: https://jte.edu.vn
Email: jte@hcmute.edu.vn

Implementation of a Neural PI Controller for PMSM Drive Systems: FPGA Based Modeling and Experimental Validation

Van Minh Nguyen[1], Van Sang Nguyen[2], Duy Khang Hoang[3], Vu Quynh Nguyen[4*]
[1]*Ho Chi Minh City University of Technology and Education, Vietnam*
[2]*Dong Nai University of Technology, Vietnam*
[3]*Luong The Vinh Gifted Highschool, Vietnam*
[4]*Lac Hong University, Vietnam*

[*]Corresponding author. *Email: vuquynh@lhu.edu.vn*

ARTICLE INFO

Received: 30/01/2025
Revised: 14/02/2025
Accepted: 24/03/2025
Published:

KEYWORDS

PMSM;
PI controller;
Neural network controller;
Sensorless;
Experimental.

ABSTRACT

This paper proposes a self-tuning PI control method using a neural network-based approach (Neural PI Controller - NPIC) for the Permanent Magnet Synchronous Motor (PMSM) drive system. First, the mathematical model of the PMSM is established and thoroughly analyzed to provide a solid theoretical foundation for control design. To enhance system performance and adaptability to dynamic uncertainties, a self-tuning PI controller (PIC) based on a Radial Basis Function Neural Network (RBF-NN) is developed to automatically adjust control parameters in real-time. Subsequently, the algorithm is implemented using Very High Speed Integrated Circuit Hardware Description Language (VHDL) and deployed on an FPGA to validate its functionality. Finally, experimental results demonstrate that the proposed controller enables the sensorless PMSM system to achieve fast and stable speed response with minimal oscillations, even under sudden load variations. These findings confirm the accuracy and effectiveness of the proposed method in real-world applications, providing a promising solution for high-performance PMSM control in industrial and automation systems.

Doi: https://doi.org/10.54644/jte.2025.1825

1. Introduction

PMSMs have been increasingly utilized in industrial automation systems due to their superior efficiency, high-speed motion control, and enhanced precision. However, in practical industrial applications, system performance is often affected by various factors such as disturbances, load variations, and friction forces, leading to degradation in operational quality.

To address these challenges, numerous intelligent control techniques [1] – [6] have been developed and applied in recent years, including fuzzy control, neural network-based control, adaptive fuzzy control, and other advanced control methods to optimize PMSM drive speed regulation. Although PICs are widely employed in industrial control systems due to their simplicity and effectiveness, a major limitation lies in their fixed control parameters, which lack adaptability to system disturbances or load variations [7], [8].

To overcome this limitation, this study proposes a NPIC, in which a Radial Basis Function Neural Network (RBF NN) is employed to identify system parameter variations and compute the appropriate values for dynamically adjusting the Kp and Ki gains of the PIC.

The implementation of this control algorithm involves extensive computational complexity. As a result, FPGA is considered an optimal solution due to its flexible hardware programmability, high processing speed, reduced design cycle, embedded processor integration, and low power consumption [6], [8]. While Digital Signal Processors (DSPs) also provide a viable alternative for intelligent control systems, they typically require a longer development time and substantial hardware resources [9].

FPGA supports two execution approaches: parallel and sequential processing. Parallel processing offers faster computation but demands higher FPGA resource utilization, whereas sequential processing achieves lower computational speed while optimizing hardware efficiency. In this study, the authors employ a finite state machine (FSM) approach to implement the control algorithm, leveraging shared functional blocks to optimize FPGA resource usage [6].

Field Programmable Gate Array (FPGA) technology has emerged as a powerful platform for designing and implementing digital control systems due to its flexibility, high parallel processing capability, and superior computational efficiency. In advanced control applications, FPGA enables the deployment of complex algorithms with minimal latency, thereby enhancing system performance. Specifically, in the study and experimental implementation of intelligent controllers such as the NPIC, utilizing FPGA not only accelerates computation but also ensures high precision in the control process.

In this research, the Cyclone IV EP4CE115F29C7N board from Intel (Altera) is selected as the hardware platform for implementing the NPIC algorithm using the VHDL (VHSIC Hardware Description Language). The Cyclone IV EP4CE115F29C7N FPGA is a reconfigurable logic device featuring 114,480 logic elements (LEs), integrated high-performance DSP multipliers, and embedded RAM blocks, which optimize numerical computations essential for control algorithms. Additionally, this FPGA supports high operating frequencies, ensuring optimal execution time for the algorithm.

Programming the FPGA using VHDL allows for the design of the NPIC control system based on a FSM structure, leveraging the parallel processing capabilities of FPGA to implement various computational modules, including multipliers, adders, comparators, and data shifters. The combination of the Cyclone IV FPGA and FSM-based VHDL programming enhances the efficiency of the control process, ensuring that the NPIC algorithm executes with high accuracy and rapid response time in motor control systems.

2. The design of Neural PI Controller on FPGA

In PMSM drive systems, the PIC is widely employed due to its simple structure and effective control performance. However, variations in load conditions and system disturbances necessitate adaptive tuning of the PIC parameters to maintain optimal performance. To estimate rotor speed without the use of sensors, the Sliding Mode Observer (SMO) is implemented to provide accurate speed feedback. Additionally, the Radial Basis Function Neural Network (RBF NN) is utilized to identify load variations and dynamically adjust the *kp* and *ki* parameters of the PIC. This mechanism enables the system to adapt to changing operating conditions, enhancing robustness and dynamic response in PMSM control. The following section provides a detailed introduction to the design of the controllers used in this study. The structure of controller was shown in Fig.1.

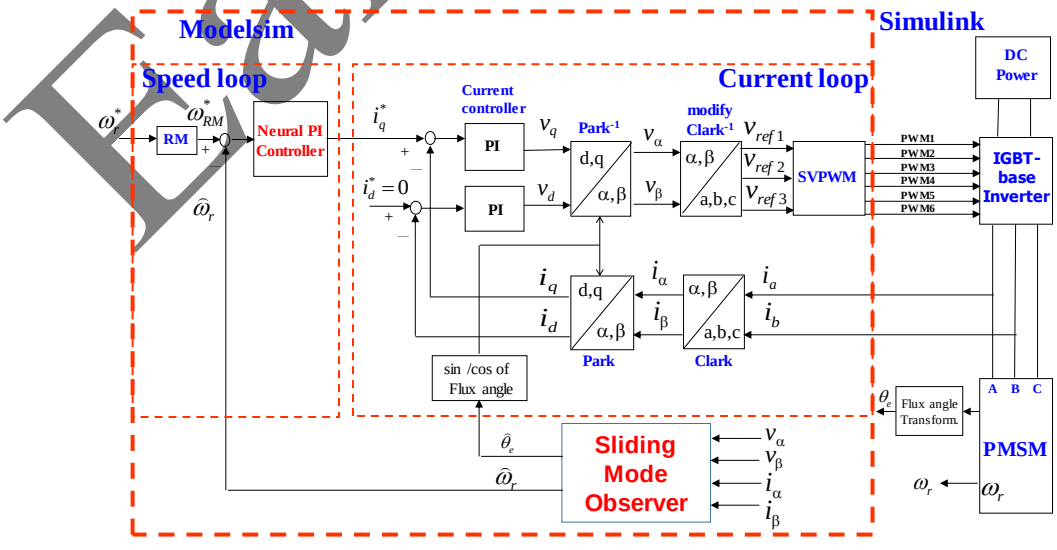

Figure 1. *The sensorless motor speed controller structure using PI algorithm to self-adjust parameters*

2.1. The mathematical model of PMSM

The mathematical model of PMSM on d-q:

$$\frac{di_d}{dt} = -\frac{R_s}{L_d} i_d + \omega_e \frac{L_q}{L_d} i_q + \frac{1}{L_d} \mathrm{v}_d \tag{1}$$

$$\frac{di_q}{dt} = -\omega_e \frac{L_d}{L_q} i_d - \frac{R_s}{L_q} i_q - \omega_e \frac{\lambda_f}{L_q} + \frac{1}{L_q} \mathrm{v}_q \tag{2}$$

Where v_d and v_q represent the voltages along the d-axis and q-axis, respectively; i_d and i_q denote the corresponding d-axis and q-axis currents; R_s is the phase winding resistance; L_d and L_q are the d-axis and q-axis inductances, respectively; ω_r is the rotating speed of the magnetic flux; and λ_f is the permanent magnet flux linkage.

The current control loop in Fig. 1 is designed based on the vector control method. When the i_d current is regulated to zero, the PMSM operates similarly to a DC motor. Consequently, the torque of the PMSM can be expressed by the following equation:

$$T_e = \frac{3P}{4} \lambda_f i_q \triangleq K_t i_q \tag{3}$$

$$K_t = \frac{3P}{4} \lambda_f \tag{4}$$

When the motor is under load, the dynamic equation of the motor is expressed as follows.

$$J_m \frac{d}{dt} \omega_r + B_m \omega_r = T_e - T_L \tag{5}$$

Where T_e is the motor torque, K_t is the torque constant, J_m represents the moment of inertia, B_m is the damping coefficient, T_L is the external load torque, and ω_r is the rotor speed.

2.2. The design of NPIC

Fig. 1 illustrates the structure of the NPIC controller for the PMSM. The system consists of a PIC, a reference model (RM), and a parameter tuning mechanism based on the Radial Basis Function Neural Network (RBF NN). The detailed information is presented as follows.

2.2.1. The PI controller

$$e(k) = \omega_m(k) - \hat{\omega}_r(k) \tag{6}$$

$$u_p(k) = K_p\, e(k) \tag{7}$$

$$\begin{aligned} u_i(k) &= K_i \sum_{j=2}^{k} e(j-1) = K_i \Big(\sum_{j=2}^{k-1} e(j-1) + e(k-1)\Big) \\ &\triangleq K_i (E_i(k-2) + e(k-1)) \;= \mathrm{u}_i(k-1) + K_i e(k-1) \end{aligned} \tag{8}$$

$$i_q^*(k) = u_p(k) + u_i(k) = K_p e(k) + K_i (E_i(k-2) + e(k-1)) \tag{9}$$

Where $E_i(k) = \sum_{j=2}^{k+1} e(j-1)$ and $u_i(k-1) = K_i E_i(k-2)$. Where $\hat{\omega}_r, \omega_m, e$ are the estimated rotor speed, the output of reference model and the error between them, respectively. The K_p, K_i are P

controller gain and *I* controller gain, respectively. The $u_p(k), u_i(k), i_q^*(k)$ are the output of *P* controller only, *I* controller only and the PIC, respectively

2.2.2. The design of Sliding Mode Observer

The SMO is employed to accurately estimate the motor speed (Fig. 1). The detailed structure of the SMO is illustrated in Fig. 2, which consists of a bang-bang controller, a low-pass filter, and a rotor flux position estimator. The mathematical model of the system is described in detail as follows.

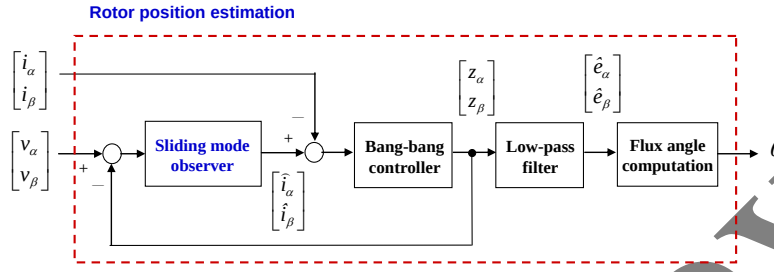

Figure 2. *The Sliding Mode Observer used to estimate the speed of PMSM*

Firstly, rewrite the mathematical mode of PMSM on d-q at equation (1) as follow:

$$\begin{bmatrix} v_d \\ v_q \end{bmatrix} = \begin{bmatrix} R_s + sL & -\omega_e L \\ \omega_e L & R_s + sL \end{bmatrix} \begin{bmatrix} i_d \\ i_q \end{bmatrix} + \begin{bmatrix} 0 \\ \omega_e K_E \end{bmatrix} \tag{10}$$

Where $L \triangleq L_d = L_q$. Transform the equation (5) to $\alpha - \beta$ axis:

$$\begin{bmatrix} v_\alpha \\ v_\beta \end{bmatrix} = \begin{bmatrix} R_s + sL & 0 \\ 0 & R_s + sL \end{bmatrix} \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} + \omega_e K_E \begin{bmatrix} -\sin\theta_e \\ \cos\theta_e \end{bmatrix} \tag{11}$$

Where $[v_\alpha \; v_\beta]^T$ is voltage on fixed coordinate; $[i_\alpha \; i_\beta]^T$ is current on fixed coordinate; θ_e is angular position at magnet flux; *s* is differential operator. In equation 6, the electromotive force (EMF) was defined as:

$$e = \begin{bmatrix} e_\alpha \\ e_\beta \end{bmatrix} \triangleq \omega_e K_E \begin{bmatrix} -\sin\theta_e \\ \cos\theta_e \end{bmatrix} \tag{12}$$

Secondly, for easy to observe EMF, the equation (11) rewrite as:

$$\frac{d}{dt}\begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} = A \cdot \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} + \frac{1}{L}\begin{bmatrix} v_\alpha \\ v_\beta \end{bmatrix} - \frac{1}{L}\begin{bmatrix} e_\alpha \\ e_\beta \end{bmatrix} \tag{13}$$

$$A = \begin{bmatrix} -R_s / L & 0 \\ 0 & -R_s / L \end{bmatrix} \tag{14}$$

Thirdly, the SMO designed as:

$$\frac{d}{dt}\begin{bmatrix} \hat{i}_\alpha \\ \hat{i}_\beta \end{bmatrix} = A \cdot \begin{bmatrix} \hat{i}_\alpha \\ \hat{i}_\beta \end{bmatrix} + \frac{1}{L}\begin{bmatrix} v_\alpha \\ v_\beta \end{bmatrix} - \frac{1}{L}\begin{bmatrix} z_\alpha \\ z_\beta \end{bmatrix} \tag{15}$$

Where the $[\hat{i}_\alpha \; \hat{i}_\beta]^T$ is the estimated current on fixed coordinate. The gain at output of bang-bang controller in Fig. 2:

$$Z = \begin{bmatrix} z_\alpha \\ z_\beta \end{bmatrix} \triangleq k * sign(\begin{bmatrix} \hat{i}_\alpha - i_\alpha \\ \hat{i}_\beta - i_\beta \end{bmatrix}) \tag{16}$$

Where the current difference is determined by the formula $e_{cur} \triangleq [\tilde{i}_\alpha \quad \tilde{i}_\beta]^T = [\hat{i}_\alpha - i_\alpha \quad \hat{i}_\beta - i_\beta]^T$. If k is chosen large enough, the equation 16 can be obtained.

$$e_{cur}^T \dot{e}_{cur} < 0 \tag{17}$$

And SMO goes into sliding mode state, thus $e_{cur} = \dot{e}_{cur} = 0$. Substituting into equation (13) and (15), Z in equation (16) will approach the electromotive force in formula (12).

$$\begin{bmatrix} z_\alpha \\ z_\beta \end{bmatrix} = \begin{bmatrix} e_\alpha \\ e_\beta \end{bmatrix} = \omega_e K_E \begin{bmatrix} -\sin\theta_e \\ \cos\theta_e \end{bmatrix} \tag{18}$$

Fourth, to reduce the oscillations generated by the bang-bang controller, a low-pass filter is added as shown in Fig. 2

$$\frac{\mathrm{d}}{\mathrm{dt}} \begin{bmatrix} \hat{e}_\alpha \\ \hat{e}_\beta \end{bmatrix} = -\omega_0 \begin{bmatrix} \hat{e}_\alpha \\ \hat{e}_\beta \end{bmatrix} + \omega_0 \begin{bmatrix} z_\alpha \\ z_\beta \end{bmatrix} \tag{19}$$

Where $\omega_0 = 2\pi f_0$. Finally, the rotor position $\hat{\theta}_e$ was estimated by:

$$\hat{\theta}_e = \tan^{-1}(-\frac{\hat{e}_\alpha}{\hat{e}_\beta}) \tag{20}$$

To program VHDL for the above expressions, we convert them to discrete form. And use $[\hat{e}_\alpha \ \hat{e}_\beta]^T$ replace for $[z_\alpha \ z_\beta]^T$ to feedback value in SMO; Therefore, the difference equation of the sliding mode observer in formula (15) is adjusted as follows:

$$\begin{bmatrix} \hat{i}_\alpha(n+1) \\ \hat{i}_\beta(n+1) \end{bmatrix} = \begin{bmatrix} \varphi & 0 \\ 0 & \varphi \end{bmatrix} \begin{bmatrix} \hat{i}_\alpha(n) \\ \hat{i}_\beta(n) \end{bmatrix} + \psi \begin{bmatrix} v_\alpha(n) \\ v_\beta(n) \end{bmatrix} - \psi \begin{bmatrix} \hat{e}_\alpha(n) \\ \hat{e}_\beta(n) \end{bmatrix} \tag{21}$$

Where $\phi \triangleq e^{-\frac{R_s}{L}T_s}$, $\psi \triangleq \frac{1}{R_s}(1 - e^{-\frac{R_s}{L}T_s})$. From equation (14), the difference equation of EMF estimation is described as follows

$$\begin{bmatrix} \hat{e}_\alpha(n+1) \\ \hat{e}_\beta(n+1) \end{bmatrix} = \begin{bmatrix} \hat{e}_\alpha(n) \\ \hat{e}_\beta(n) \end{bmatrix} + 2\pi f_0 \begin{bmatrix} z_\alpha(n) - \hat{e}_\alpha(n) \\ z_\beta(n) - \hat{e}_\beta(n) \end{bmatrix} \tag{22}$$

The rotor position is calculated using:

$$\hat{\theta}_e(n) = \tan^{-1}(-\frac{\hat{e}_\alpha(n)}{\hat{e}_\beta(n)}) \tag{23}$$

The sequence of estimating motor speed is as follows:

Step 1: Estimate the current using equation (21).

Step 2: Calculate the current error by $\tilde{i}_\alpha(n) = \hat{i}_\alpha(n) - i_\alpha(n)$ và $\tilde{i}_\beta(n) = \hat{i}_\beta(n) - i_\beta(n)$

Step 3: Calculate the gain factor Z of the SMO using equation (16)

Step 4: Estimate the electromotive force using equation (22).

Step 5: Calculate the rotor position using equation (23)

2.2.3. The design of RBF NN

The RBF NN has a three-layer structure consisting of an input layer, a hidden layer, and an output layer as shown in Fig. 3. RBF NN has three inputs consisting of $i_q^*(k), \omega_r(k-1)$ và $\omega_r(k-2)$, expressed in vector form as follows

$$X = [i_q^*(k), \omega_r(k-1), \omega_r(k-2)]^T \tag{24}$$

Using multivariate Gaussian function for activation in hidden layer of RBF NN:

$$h_r = \exp(-\frac{\|X - c_r\|^2}{2\sigma_r^2}), r = 1,2,3,4,....p \tag{25}$$

Where $c_r = [c_{r1}, c_{r2}, c_{r3}]^T$,. The output of RBF NN in Fig.3 is described as follows:

$$\omega_{rbf} = \sum_{r=1}^{p} w_r h_r \tag{26}$$

Where ω_{rbf} is the output value; w_r and h_r are the weight and output of r^{th} neuron, respectively. The cost function is defined:

$$J = \frac{1}{2}(\omega_{rbf} - \omega_r)^2 \triangleq \frac{1}{2} e_{nn}^2 \tag{27}$$

According to the gradient descent method, the learning algorithm of the weight function, node center and variance is as follows:

$$w_r(k+1) = w_r(k) + \eta e_{nn}(k) h_r(k) \tag{28}$$

$$c_{rs}(k+1) = c_{rs}(k) + \eta e_{nn}(k) w_r(k) h_r(k) \frac{X_s(k) - c_{rs}(k)}{\sigma_r^2(k)} \tag{29}$$

$$\sigma_r(k+1) = \sigma_r(k) + \eta e_{nn}(k) w_r(k) h_r(k) \frac{\|X(k) - c_r(k)\|^2}{\sigma_r^3(k)} \tag{30}$$

Where *r = 1,2,..p, s = 1,2,3*. The Jacobian $\frac{\partial \omega_r}{\partial i_q^*}$ deduced from Fig. 3 and equation (12) as follows:

$$\frac{\partial \omega_r}{\partial i_q^*} \approx \frac{\partial \omega_{rbf}}{\partial i_q^*} = \sum_{r=1}^{p} w_r h_r \frac{c_{r1} - i_q^*(k)}{\sigma_r^2} \tag{31}$$

2.2.4. Reference model - RM:

The Reference Model (RM) serves as a standard model in the motor control system, providing the desired speed trajectory for comparison with actual feedback. The RM is designed to ensure motor speed stability under varying load conditions, including light, normal, and heavy loads. By utilizing the reference model, the controller can adaptively adjust the control signal, enabling the system to achieve

optimal dynamic performance and enhance robustness against load variations. The reference model is a quadratic inertial function.

$$\frac{\omega_m(s)}{\omega_r^*(s)} = \frac{\omega_n^2}{s^2 + 2\varsigma\omega_n s + \omega_n^2} \tag{32}$$

Applying bilinear transformation on matlab, equation (19) is rewritten in discrete form as follows:

$$\frac{\omega_m(z^{-1})}{\omega_r^*(z^{-1})} = \frac{\theta_0 + \theta_1 z^{-1} + \theta_2 z^{-2}}{1 + \varphi_1 z^{-1} + \varphi_2 z^{-2}} \tag{33}$$

And the difference equation is:

$$\omega_m(k) = -\phi_1\omega_m(k-1) - \phi_2\omega_m(k-2) + \theta_0\omega_r^*(k) + \theta_1\omega_r^*(k-1) + \theta_2\omega_r^*(k-2) \tag{34}$$

2.2.5. Mechanism for adjusting coefficients Kp and Ki

The PIC's parameter tuning mechanism is based on minimizing the squared error between the rotor speed and the output of the reference model. The cost function is defined as follows:

$$J_e \triangleq \frac{1}{2}e^2 = \frac{1}{2}(\omega_m - \omega_r)^2 \approx \frac{1}{2}(\omega_m - \hat{\omega}_r)^2 \tag{35}$$

And the parameters of the PIC are adjusted by the formula:

$$\Delta K_p \propto -\frac{\partial J_e}{\partial K_p} = -\alpha\frac{\partial J_e}{\partial K_p} \;\&\; \Delta K_i \propto -\frac{\partial J_e}{\partial K_i} = -\alpha\frac{\partial J_e}{\partial K_i} \tag{36}$$

Where α represents learning rate. Partial Differential Equations J_e of equation (36) shown as

$$\frac{\partial J_e}{\partial K_p} = \frac{\partial J_e}{\partial e}\frac{\partial e}{\partial \omega_r}\frac{\partial \omega_r}{\partial i_q^*}\frac{\partial i_q^*}{\partial K_p} \;\&\; \frac{\partial J_e}{\partial K_i} = \frac{\partial J_e}{\partial e}\frac{\partial e}{\partial \omega_r}\frac{\partial \omega_r}{\partial i_q^*}\frac{\partial i_q^*}{\partial K_i} \tag{37}$$

From equation (6), (9), (35) and (31), we have

$$\frac{\partial J_e}{\partial e} = e,\ \frac{\partial e}{\partial \omega_r} \approx \frac{\partial e}{\partial \hat{\omega}_r} = -1,\ \frac{\partial i_q^*(k)}{\partial K_p} = e(k),\ \frac{\partial i_q^*(k)}{\partial K_i} = E_i(k-2) + e(k-1) \tag{38}$$

$$\frac{\partial \omega_r}{\partial i_q^*} \approx \frac{\partial \hat{\omega}_r}{\partial i_q^*} \approx \frac{\partial \omega_{rbf}}{\partial i_q^*} = \sum_{r=1}^{p} w_r h_r \frac{c_{r1} - i_q^*(k)}{\sigma_r^2} \tag{39}$$

Substituting equation (37)~(39) into equation (36), the PIC parameters are adjusted by the following expression:

$$\Delta K_p(k) = \alpha e^2(k)\sum_{r=1}^{p} w_r h_r \frac{c_{r1} - i_q^*(k)}{\sigma_r^2} \tag{40}$$

$$\Delta K_i(k) = \alpha e(k)(E_i(k-2) + e(k-1))\sum_{r=1}^{p} w_r h_r \frac{c_{r1} - i_q^*(k)}{\sigma_r^2} \tag{41}$$

From equation (8), $E_i(k-2)$ replace by $u_i(k-1)/K_i$ (42)

JTE
HCMUTE
ISSN: 1859-1272

TẠP CHÍ KHOA HỌC GIÁO DỤC KỸ THUẬT
Trường Đại học Sư phạm Kỹ thuật Thành phố Hồ Chí Minh
Website: https://jte.edu.vn
Email: jte@hcmute.edu.vn

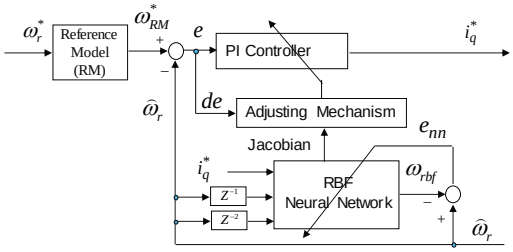

Figure 3. *Structure of PI self-tuning controller*

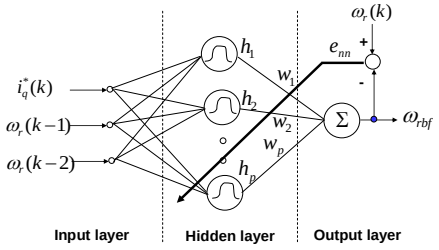

Figure 4. *The structure of RBF NN*

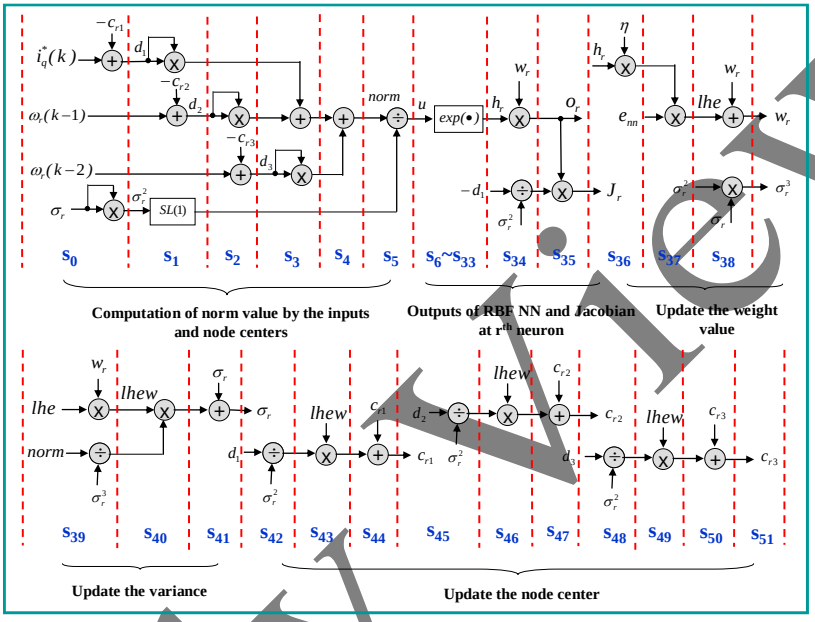

Figure 5. *Computation diagram of the r-th neuron of RBF NN*

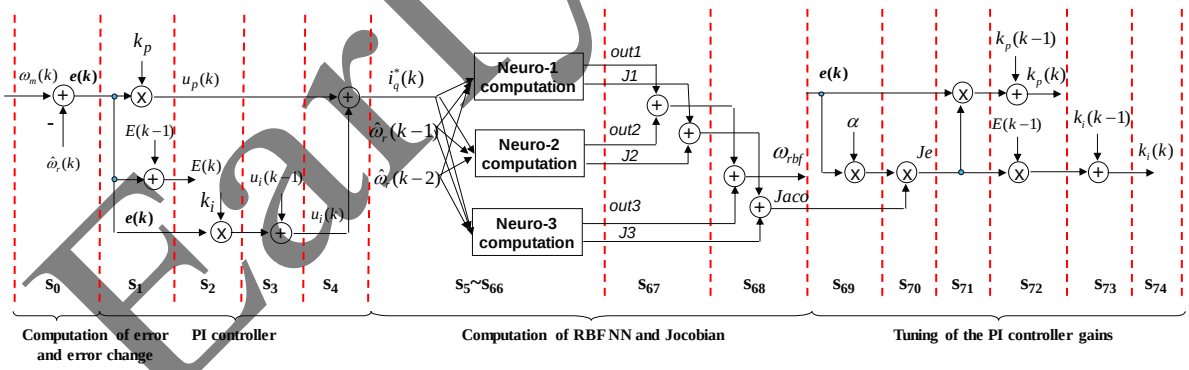

Figure 6. *NPIC algorithm calculation diagram for PMSM motor speed control.*

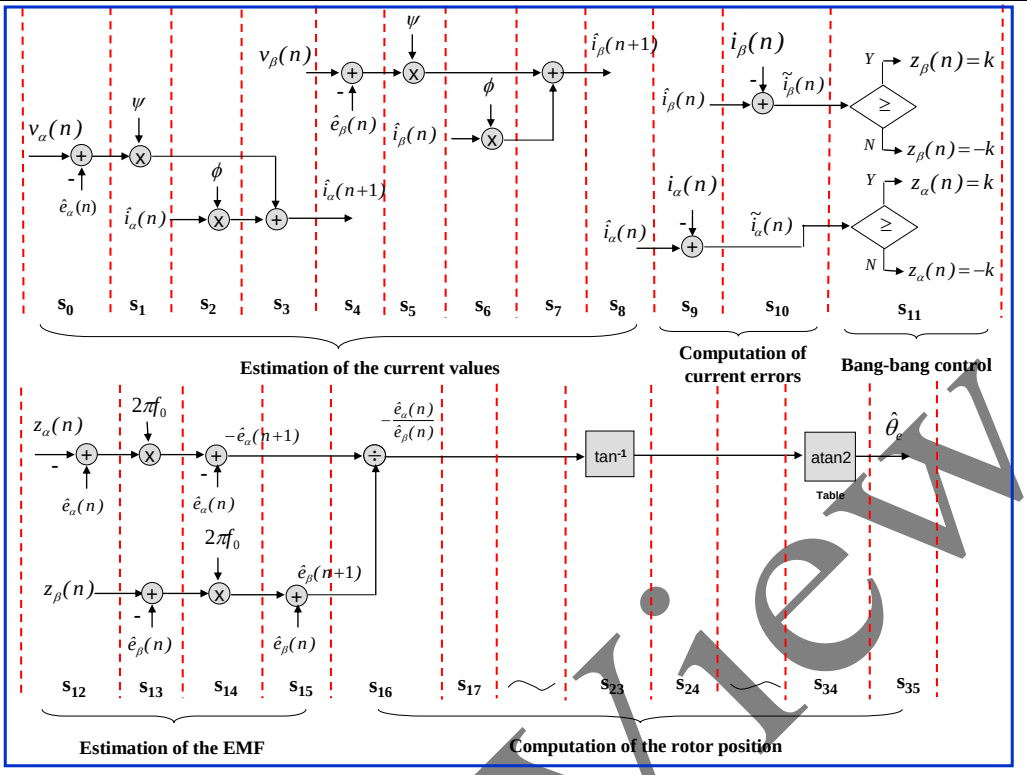

Figure 7. *Rotor speed estimation calculation diagram using SMO algorithm*

3. Algorithm implementation using VHDL

To reduce the hardware resource usage, the FSM method is used to program the algorithm. The implementation of the exponential VHDL programming in equation (25) is done as follows:

$$h_r = \exp(-\frac{\|X - c_r\|^2}{2\sigma_r^2}) \triangleq \exp(-u) \tag{43}$$

With u >4 then the output is approximately zero, so we restrict u to lie in the range $0 \le u < 4$, equation (43) Applying the Tayor series description we have:

$$h_r = \exp(-u) = \sum_{n=0}^{\infty}(-1)^n \frac{u^n}{n!} \approx \sum_{n=0}^{12}(-1)^n \frac{u^n}{n!} \tag{44}$$

To avoid overflow during VHDL programming, we take equation (44) divided by 16 and rewrite it as follows

$$h_r = 16\frac{\exp(-4r)}{16} \approx 16\sum_{n=0}^{12}(-1)^n \frac{4^{n-2}r^n}{n!} \triangleq 16\sum_{n=0}^{12} a_n r^n \tag{45}$$

Where $a_n \triangleq (-1)^n \frac{4^{n-2}}{n!}$ and $a_0 = 0.00625,\ a_1 = -0.25,\ \ldots.,a_{12} = 0.00218909$

The Taylor series expansion for the 12^{th} degree polynomial and equation (30) is rewritten as follows:

$$\begin{aligned} h_r = 16((((((((((((a_{12}r + a_{11})r + a_{10})r + a_9)r + a_8)r + a_7) \\ r + a_6)r + a_5)r + a_4)r + a_3)r + a_2)r + a_1)r + a_0) \end{aligned} \tag{46}$$

Each neuron in Fig. 4 involves complex computations, incorporating Equations (25), (26), (28)–(30), and (31). By applying the previously discussed analytical method, the computational process for the r-

th neuron is described in Fig. 6, where input are $i_q^*(k)$, $\omega_r(k-1)$, $\omega_r(k-2)$ and output are O_r, J_r. Fig. 5 outlines the execution steps: s0–s5 compute the normalized value, s6–s35 determine the exponential function and neuron output, s36–s38 update the weight function, s39–s41 update variance, and s42–s51 update the centroid function. The multipliers and adders are implemented using standard Altera functions. Given a clock frequency of 12.5 MHz (80 ns per cycle), a total of 52 steps completes in only 4.16 µs.

After detailing the computational process for each neuron, this method is further extended to describe the entire NPIC system. In the proposed design, the hidden layer consists of three neurons. The NPIC control structure is depicted in Fig. 6, requiring a total of 75 clock cycles for execution. Although the NPIC algorithm is highly complex, the FSM approach enables efficient VHDL implementation. In Fig. 6, steps s0–s4 compute the speed error and execute the PIC, steps s5–s68 compute the values of the RBF NN and the Jacobian using parallel neurons, while steps s69–s74 adjust the PIC parameters. With a 12.5 MHz clock frequency (80 ns per cycle), the total 75 steps require only 6 µs for execution.

A similar approach is applied to estimate motor speed. Fig. 7 illustrates the detailed execution of each instruction cycle in VHDL, comprising 36 computational steps. Steps s0–s8 compute current values, s9–s10 determine current deviation, s11 executes bang-bang control, steps s12–s15 compute the electromotive force (EMF), and steps s16–s35 estimate rotor position. Given a clock frequency of 12.5 MHz (80 ns per cycle), the total 36 steps complete in only 2.88 µs.

4. Experimental results

Based on the successful simulation in paper [10], we realize this code in the FPGA kit. The experimental system shows in Fig. 8, and the experimental results show in Fig. 9-11. The encoder signal just use to compare with the output signal from SMO.

Fig. 8 illustrates the experimental setup, which comprises an FPGA-based control system (Cyclone IV EP4CE115F29C7N), a motor drive board, an encoder interface board, an ADC module, and a PMSM. The detailed specifications of the PMSM are provided in Table 1. To accurately measure the rotor position, an incremental encoder with a resolution of 2500 pulses per second (pps) is mounted on the motor shaft.

The motor drive system incorporates a three-phase inverter utilizing Insulated Gate Bipolar Transistor (IGBT) modules. The TLP250 optocoupler from Toshiba is employed in the gate driver circuit to ensure reliable signal isolation and efficient switching performance of the IGBT modules. The PWM signals generated by the FPGA serve as the control inputs for the motor drive board.

To facilitate real-time control and system integration, a Nios II soft-core embedded processor is implemented within the FPGA, establishing a System-on-a-Programmable-Chip (SoPC) environment. This architecture enhances computational efficiency and flexibility, enabling seamless execution of advanced control algorithms within the FPGA framework

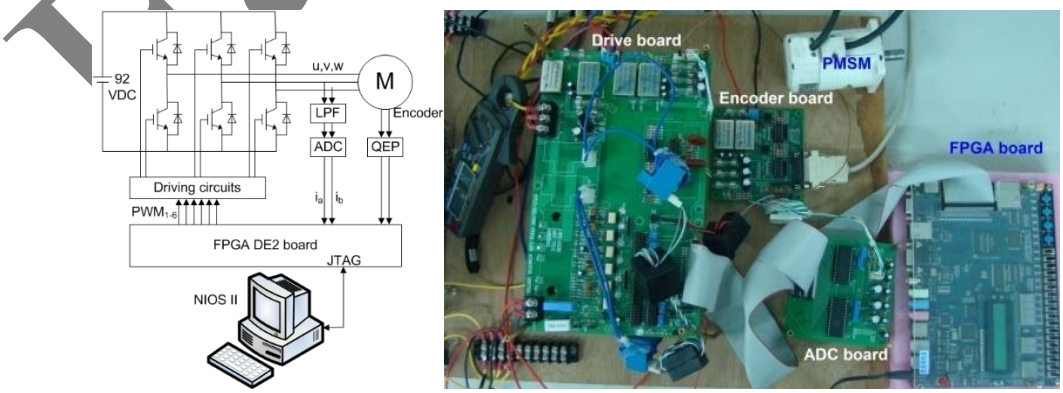

Figure 8. *The experimental system.*

JTE
HCMUTE
ISSN: 1859-1272

TẠP CHÍ KHOA HỌC GIÁO DỤC KỸ THUẬT
Trường Đại học Sư phạm Kỹ thuật Thành phố Hồ Chí Minh
Website: https://jte.edu.vn
Email: jte@hcmute.edu.vn

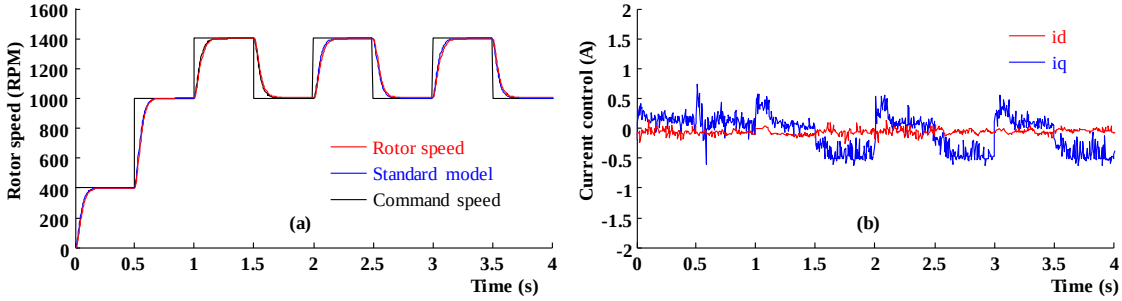

Figure 9. *Only* PIC *without an external load. (a) Rotor speed. (b) Current control.*

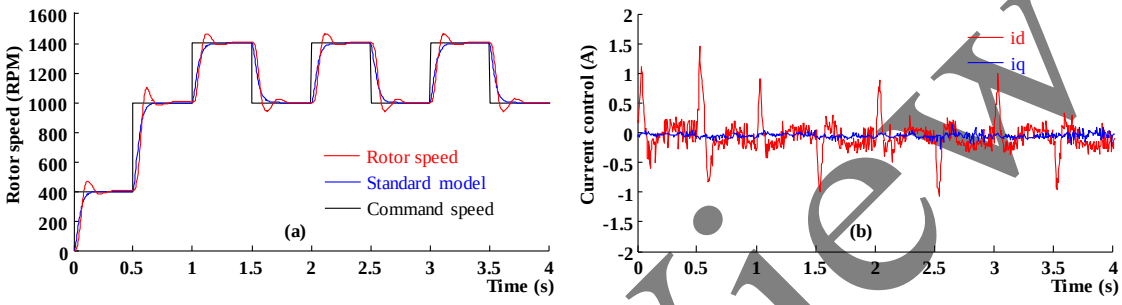

Figure 10. *Only* PIC *with an external load = 5 kg. (a) Rotor speed. (b) Current control.*

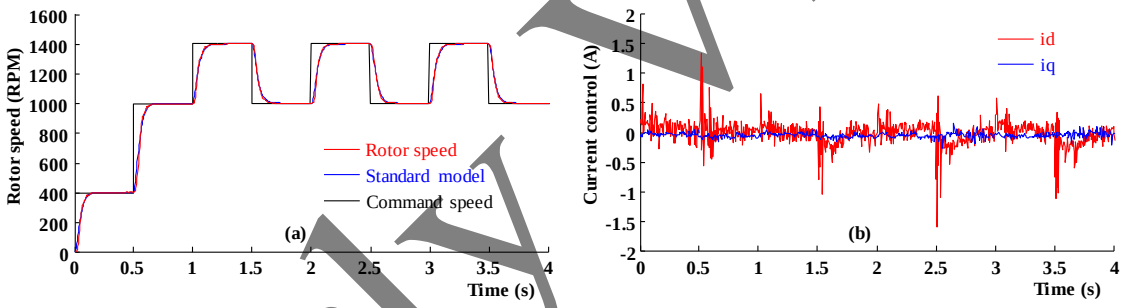

Figure 11. *The NPIC with external load = 5 kg. (a) Rotor speed. (b) Current control*

Figures 9 to 11 illustrate the step response of the PMSM under varying load conditions (0 kg and 5 kg) using conventional PIC and NPIC. The speed reference signal follows a square wave profile with a period of 0.5 s and transitions between speed levels of 0 rpm → 400 rpm → 1000 rpm → 1400 rpm → 1000 rpm → 1400 rpm → 1000 rpm → 1400 rpm → 1000 rpm, serving as the test input.

In Fig. 9a, under a light-load condition (0 kg), the PIC exhibits excellent dynamic performance, achieving a rise time of 0.1 s without overshoot and maintaining zero steady-state error. However, under heavy-load conditions (5 kg) with the same PIC, the system's response deteriorates, as shown in Fig. 10a, where a 48 rpm overshoot is observed, indicating reduced robustness.

To address this issue, a NPIC is implemented, as depicted in Fig. 11a. The application of the NPIC significantly enhances the speed response, effectively eliminating overshoot while ensuring a more stable performance. Although Figures 9b, 10b, and 11b reveal slight ripples in the measured current due to Electromagnetic Interference (EMI) effects, the experimental results confirm that the current follows the reference command with high accuracy.

Furthermore, these findings not only validate the efficacy of the vector control strategy but also demonstrate its ability to decouple the PMSM dynamics. The consistency between the simulation and experimental results further substantiates the effectiveness and accuracy of the proposed control algorithm for PMSM applications.

5. Conclusion

This paper has presented the NPIC for speed regulation of PMSM and successfully validated its effectiveness through both simulation (as demonstrated in [10]) and experimental implementation in this study.

To address system instability, a RBF NN was employed to model the system dynamics and adaptively adjust the PIC parameters in real-time. During the VHDL implementation, the entire algorithm was efficiently realized using the FSM approach, significantly optimizing FPGA resource utilization.

The obtained simulation and experimental results confirm that the proposed NPIC algorithm enables precise and rapid speed tracking, ensuring robust performance even under significant load variations, thereby demonstrating its potential for practical PMSM control applications.

Table 1. *The parameters of PMSM*

Parameters	Value
Stator resistor	1.3Ω
Stator inductance	6.3mH
Pole pairs	4
Inertia	J=0.000108 kg*m2
Friction factor	F=0.0013 N*m*s
Voltage constant	52.2 V_peak/1000rpm
Torque constant	0.43169 N.m/A_peak

Acknowledgments

This work belongs to the project in 2025 funded by the Lac Hong University, Vietnam National University Ho Chi Minh City and Dong Nai University of Technology, Vietnam.

Conflict of Interest

The authors declare no conflict of interest.

REFERENCES

[1] A. A. Nada and M. A. Bayoumi, "Development of embedded fuzzy control using reconfigurable FPGA technology," *Automatika*, vol. 65, no. 2, pp. 609–626, 2024, doi: 10.1080/00051144.2024.2313904.

[2] S. Mahfoud, A. Derouich, N. El Ouanjli, M. A. Mossa, M. S. Bhaskar, N. K. Lan, and N. V. Quynh, "A new robust direct torque control based on a genetic algorithm for a doubly-fed induction motor: Experimental validation," *Energies*, vol. 15, p. 5384, 2022, doi: 10.3390/en15155384.

[3] S. Mahfoud, A. Derouich, N. El Ouanjli, N. V. Quynh, and M. A. Mossa, "A new hybrid ant colony optimization based PID of the direct torque control for a doubly fed induction motor," *World Electric Vehicle Journal*, vol. 13, p. 78, 2022, doi: 10.3390/wevj13050078.

[4] X. Jiang, Y. Wang, and J. Dong, "Speed regulation method using genetic algorithm for dual three-phase permanent magnet synchronous motors," *CES Transactions on Electrical Machines and Systems*, vol. 7, no. 2, pp. 171–178, Jun. 2023, doi: 10.30941/CESTEMS.2023.00013.

[5] H. Echeikh, M. A. Mossa, N. V. Quynh, A. A. Ahmed, and H. H. Alhelou, "Enhancement of induction motor dynamics using a novel sensorless predictive control algorithm," *Energies*, vol. 14, p. 4377, 2021, doi: 10.3390/en14144377.

[6] J. Lee, H. Kim, B. S. Kim, S. Jeon, J. C. Lee, and D. S. Kim, "Implementing binarized neural network processor on FPGA-based platform," in *Proc. IEEE 4th Int. Conf. Artif. Intell. Circuits Syst. (AICAS)*, Incheon, Republic of Korea, 2022, pp. 469–471, doi: 10.1109/AICAS54282.2022.9869997.

[7] B. Bairwa, M. Murari, M. Shahapur, K. M. R, and M. F. Khan, "Speed control of BLDC motor using PI controller," in *Proc. Int. Conf. Adv. Technol. (ICONAT)*, Goa, India, 2023, pp. 1–6, doi: 10.1109/ICONAT57137.2023.10080074.

[8] D. M. Kumar, M. Cirrincione, H. K. Mudaliar, M. di Benedetto, A. Lidozzi, and A. Fagiolini, "Development of a fractional PI controller in an FPGA environment for a robust high-performance PMSM electrical drive," in *Proc. IEEE 12th Energy Convers. Congr. Expo. – Asia (ECCE-Asia)*, Singapore, 2021, pp. 2427–2431, doi: 10.1109/ECCE-Asia49820.2021.9479450.

[9] A. Panda, S. Kahare, and S. K. Gawre, "DSP TMS320F28377S based speed control of DC motor," in *Proc. IEEE Int. Students' Conf. Elect., Electron. Comput. Sci. (SCEECS)*, Bhopal, India, 2020, pp. 1–4, doi: 10.1109/SCEECS48394.2020.133.

[10] M. Nguyen Van and N. V. Quynh, "Speed control for sensorless PMSM based self-tuning PI controller," in *Proc. Vietnam Int. Conf. Exhib. Control Autom. (VCCA)*, 2024, pp. 1–8.

Van Minh Nguyen, born in 1984, obtained a Master's degree in Electrical Engineering from Lac Hong University in 2017. He is currently working at Vietnam National University, Ho Chi Minh City. Since 2024, he has been a doctoral researcher at Ho Chi Minh City University of Technology and Education, focusing on electric drive control, modeling and simulation, renewable energy, and robotics.
Email: minhnv.ncs@hcmute.edu.vn. ORCID: https://orcid.org/0009-0003-2666-4438

Van Sang Nguyen, born in 1983, graduated from the major of electrification and power supply at Ho Chi Minh City University of Technical Education in 2007 and graduated with a Master's degree in electrical engineering at Lac Hong University in 2017. He is currently a lecturer at the Department of Electrical and Electronic Engineering, Faculty of Engineering, Dong Nai University of Technology.
Email: nguyenvansang@dntu.edu. ORCID: https://orcid.org/0009-0001-2855-6962

Duy Khang Hoang, born in 2008, currently studying grade 12th in Luong The Vinh gifted highschool, his interested studying fields include computer science, modeling and simulation and English language studies.
Email: hoangduykhang2018@gmail.com. ORCID: https://orcid.org/0009-0009-3247-2313

Vu Quynh Nguyen, born in 1979, earned a Ph.D. in Electrical Engineering from Southern Taiwan University of Science and Technology (Taiwan) in 2013. He was conferred the title of Associate Professor in 2020 and is currently the Vice President of Lac Hong University. His primary research interests include control engineering, renewable energy, and robotics.
Email: vuquynh@lhu.edu.vn. ORCID: https://orcid.org/0000-0002-1479-4791