DESIGN AND SIMULATION OF A SPEED CONTROL IC FOR PMSM DRIVE BASED ON NEURAL FUZZY CONTROL

Ying-Shieh Kung [1], Nguyen Vu Quynh[2], Hsin-Hung Chou[3], Chiu-Pao Tien[4], Chih-Nan Yen[5]

[1,2,5] Department of Electrical Engineering, Southern Taiwan University, Taiwan
e-mail: kung@mail.stut.edu.tw[1], vuquynh@lhu.edu.vn[2], m9820211@mail.stut.edu.tw[5]
[3,4] Mechanical and Systems Research Laboratories, Industrial Technology Research Institute, Taiwan
e-mail: carin6969@itri.org.tw[3], ChiuPaoTien@itri.org.tw[4]

Abstract – The work presents a neural fuzzy control (NFC) for speed control IC of permanent synchronous motor (PMSM) drive system. Firstly, a mathematic model of the PMSM drive is derived; then to increase the performance of the PMSM drive system, a fuzzy controller (FC) which its parameters are adjusted by a radial basis function neural network (RBF NN) is applied to the speed controller for coping with the effect of the system dynamic uncertainty and the external load. Secondly, the Verilog hardware description language (Verilog HDL) is adopted to describe the behaviour of the speed control IC which includes the circuits of space vector pulse width modulation (SVPWM), coordinate transformation, NFC, etc. Thirdly, the simulation work is performed by MATLAB/Simulink and ModelSim co-simulation mode, provided by Electronic Design Automation (EDA) Simulator Link. The PMSM, inverter and speed command are performed in Simulink and the speed control IC of PMSM drive is executed in ModelSim. Finally, the co-simulation results validate the effectiveness of the proposed NFC-based speed control system.

Keywords – PMSM, Neural fuzzy control, Verilog HDL, ModelSim, Matlab, Simulink, Co-simulation.

1. INTRODUCTION

PMSM has been increasingly used in many automation control fields as actuators, due to its advantages of superior power density, high-performance motion control with fast speed and better accuracy. But in industrial applications, there are many uncertainties, such as system parameter uncertainty, external load disturbance, friction force, unmodeled uncertainty, etc. which always diminish the performance quality of the pre-design of the motor driving system. To cope with this problem, in recent years, many intelligent control techniques [1-2], such as fuzzy control, neural networks control, adaptive fuzzy control and other control method, have been developed and applied to the speed control of servo motor drives to obtain high operating performance. Although fuzzy control has been successful applied in several industrial automation, however, it is not an easy task to obtain an optimal set of fuzzy membership functions and rules in FC. In this paper, a neural fuzzy controller (NFC) is proposed and a RBF NN is used to identify the plant dynamic and provided more accuracy plant information for parameters tuning of FC.

In implementation, although the execution of NFC requires many computations, FPGA can provide a solution in this issue. Especially, FPGA with programmable hard-wired feature, fast computation ability, shorter design cycle, embedding processor, low power consumption and higher density is better for the implementation of the digital system [2-3] than DSP.

Recently, a co-simulation work by Electronic Design Automation (EDA) Simulator Link has been gradually applied to verify the effectiveness of the Verilog and VHDL code in the motor drive system [4-7]. The EDA Simulator Link [8] provides a co-simulation interface between MALTAB or Simulink and HDL simulators-ModelSim [9]. Using it you can verify a VHDL, Verilog, or mixed-language implementation against your Simulink model or MATLAB algorithm [8]. Therefore, EDA Simulator Link lets you use MATLAB code and Simulink models as a test bench that generates stimulus for an HDL simulation and analyzes the simulation's response [8]. Therefore, in this paper, the co-simulation by EDA Simulator Link is applied. The PMSM, inverter and speed command are performed in Simulink and the speed control IC described by Verilog HDL code is executed in ModelSim. Finally, some simulations results validate the effectiveness of the proposed NFC-based speed control system of PMSM drive.

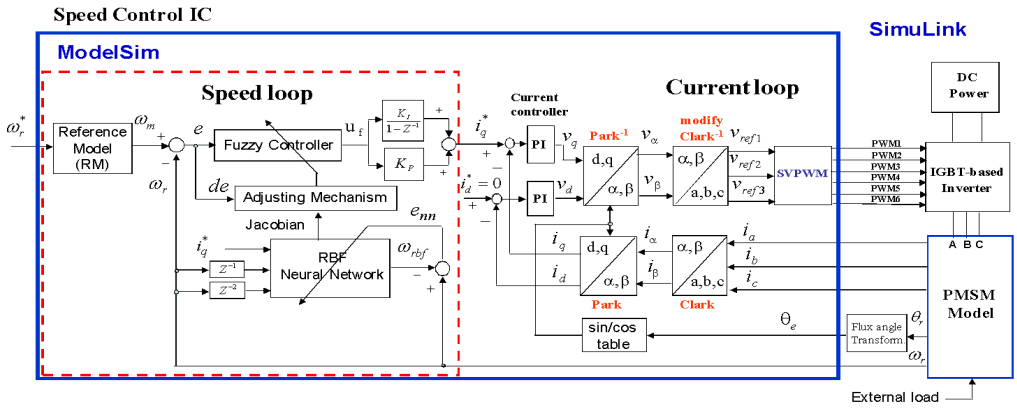

Fig.1 The simulation architecture of NFC-based speed control for PMSM drive

2. System Description of PMSM Drive and Speed Controller Design

The simulation architecture of NFC-based speed control for PMSM drive is shown in Fig. 1. The modelling of PMSM and the algorithm of the fuzzy controller using RBF NN are introduced as follows:

2.1. Mathematical Model of PMSM

The typical mathematical model of a PMSM is described, in two-axis *d-q* synchronous rotating reference frame, as follows

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

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

where v_d, v_q are the d and q axis voltages; i_d, i_q, are the d and q axis currents, R_s is the phase winding resistance; L_d, L_q are the d and q axis inductance; ω_e is the rotating speed of magnet flux; λ_f is the permanent magnet flux linkage.

The current loop control of PMSM drive in Fig.1 is based on a vector control approach. That is, if the i_d is controlled to 0 in Fig.1, the PMSM will be decoupled and controlling a PMSM like to control a DC motor. Therefore, after decoupling, the torque of PMSM can be written as the following equation,

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

with

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

Considering the mechanical load, the overall dynamic equation of PMSM drive system is obtained by

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

where T_e is the motor torque, K_t is torque constant, J_m is the inertial value, B_m is damping ratio, T_L is the external torque, ω_r is rotor speed.

2.2. Design of Neural Fuzzy Controller (NFC)

The dash rectangular area in Fig. 1 presents the architecture of an NFC for the PMSM drive. It consists of a FC, a reference model and a RBF NN based parameter adjusting mechanism. Detailed description of these is as follows.

(1) *Fuzzy controller (FC):*

The FC in this study uses singleton fuzzifier, triangular membership function, product-inference rule and central average defuzzifier method. In Fig. 1, the tracking error e and the error change de are defined by

$$e(k) = \omega_m(k) - \omega_r(k) \tag{6}$$

$$de(k) = e(k) - e(k-1) \tag{7}$$

Where u_f represents the output of the FC and ω_m is the output of reference model. The design procedure of FC algorithm is as follows. Firstly, in Fig.1, the e and de are taken as the input variable of FC, and their linguist variables are defined as E and dE. Each linguist value of E and dE are based on the symmetrical triangular membership function. Secondly, the computation of the membership degree for e and de are done. However, the only two linguistic values can be excited in any input value. Thirdly, the selection of the initial FC rules refers to the dynamic response characteristics, such as,

$$IF\ e\ is\ A_i\ and\ \Delta e\ is\ B_j\ THEN\ u_f\ is\ c_{j,i}, \tag{8}$$

where i and j are from 0 to 6, A_i and B_j are fuzzy number, and $c_{j,i}$ is real number. Finally, to construct

the fuzzy system $u_f(e,de)$, the singleton fuzzifier, product-inference rule, and central average defuzzifier method is adopted and the output of the fuzzy inference can be obtained by the following expression:

$$u_f(e,de)=\frac{\sum_{n=i}^{i+1}\sum_{m=j}^{j+1}c_{m,n}[\mu_{A_n}(e)*\mu_{B_m}(de)]}{\sum_{n=i}^{i+1}\sum_{m=j}^{j+1}\mu_{A_n}(e)*\mu_{B_m}(de)}\triangleq\sum_{n=i}^{i+1}\sum_{m=j}^{j+1}c_{m,n}*\xi_{n,m} \quad (9)$$

where $\xi_{n,m}\triangleq\mu_{A_n}(e)*\mu_{B_m}(de)$.

(2) *RADİAL BASİS FUNCTİON NEURAL NETWORK*

The RBF NN adopted here is a three-layer architecture which is shown in Fig. 2 and comprised of one input layer, one hidden layer and one output layer.

The RBF NN has three inputs by $i_q^*(k)$, $\omega_r(k-1)$, $\omega_r(k-2)$ and its vector form is represented by

$$X=[i_q^*(k),\omega_r(k-1),\omega_r(k-2)]^T \quad (10)$$

Furthermore, the multivariate Gaussian function is used as the activated function in hidden layer of RBF NN, and its formulation is shown as follows.

$$h_r=\exp(-\frac{\|X-c_r\|^2}{2\sigma_r^2}),r=1,2,3,4,....q \quad (11)$$

where $c_r=[c_{r1},c_{r2},c_{r3}]^T$ and σ_r denote the node center and node variance of r^{th} neuron, and $\|X-c_r\|$ is the norm value which is measured by the inputs and the node center at each neuron. And the network output in Fig. 2 can be written as

$$\omega_{rbf}=\sum_{r=1}^{q}w_r h_r \quad (12)$$

where ω_{rbf} is the output value; w_r and h_r are the weight and output of r^{th} neuron, respectively.

The instantaneous cost function is defined as follows.

$$J=\frac{1}{2}(\omega_{rbf}-\omega_r)^2\triangleq\frac{1}{2}e_{nn}^2 \quad (13)$$

According to the gradient descent method, the learning algorithm of weights, node center and variance are as follows:

$$w_r(k+1)=w_r(k)+\eta e_{nn}(k)h_r(k) \quad (14)$$

$$c_{rs}(k+1)=c_{rs}(k)+\eta e_{nn}(k)w_r(k)h_r(k)\frac{X_s(k)-c_{rs}(k)}{\sigma_r^2(k)} \quad (15)$$

$$\sigma_r(k+1)=\sigma_r(k)+\eta e_{nn}(k)w_r(k)h_r(k)\frac{\|X(k)-c_r(k)\|^2}{\sigma_r^3(k)} \quad (16)$$

where $r=1,2,..q$, $s=1,2,3$ and η is a learning rate. Further, the $\frac{\partial\omega_r}{\partial i_q^*}$ is Jacobian transformation and can be derived from Fig.2

$$\frac{\partial\omega_r}{\partial i_q^*}\approx\frac{\partial\omega_{rbf}}{\partial i_q^*}=\sum_{r=1}^{q}w_r h_r\frac{c_{r1}-i_q^*(k)}{\sigma_r^2} \quad (17)$$

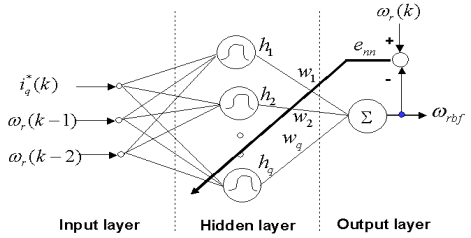

Fig.2 RBF NN

(3) *Adjusting mechanism of fuzzy controller*

The gradient descent method is used to derive the NFC control law in Fig. 1. The adjusting mechanism of FC parameters is to minimize the square error between the rotor speed and the output of the reference model. The instantaneous cost function is firstly defined by

$$J_e\triangleq\frac{1}{2}e^2=\frac{1}{2}(\omega_m-\omega_r)^2 \quad (18)$$

and the parameters of c_{mn} are adjusted according to

$$\Delta c_{m,n}\propto-\frac{\partial J_e}{\partial c_{m,n}}=-\alpha\frac{\partial J_e}{\partial c_{m,n}} \quad (19)$$

where α represents learning rate. Secondly, the chain rule is used, and the partial differential equation for J_e in (18) can be written as

$$\frac{\partial J_e}{\partial c_{m,n}}=-e\frac{\partial\omega_r}{\partial u_f}\frac{\partial u_f}{\partial c_{m,n}} \quad (20)$$

Further, from (9) and using the Jocobian formulation from (17), we can respectively get

$$\frac{\partial u_f(k)}{\partial c_{m,n}(k)}=d_{n,m} \quad (21)$$

and,

$$\frac{\partial\omega_r}{\partial u_f}\approx(K_P+K_I)\frac{\partial\omega_{rbf}}{\partial i_q^*}=(K_P+K_I)\sum_{r=1}^{q}w_r h_r\frac{c_{r1}-i_q^*(k)}{\sigma_r^2} \quad (22)$$

Therefore, substituting (21) and (22) into (20), the parameters $c_{m,n}$ of fuzzy controller described in (9) can be adjusted by the following expression.

$$\Delta c_{m,n}(k) = \alpha e(k)(K_p + K_i) d_{n,m} \sum_{r=1}^{q} w_r h_r \frac{c_{r1} - i_q^*(k)}{\sigma_r^2} \quad (23)$$

with $m = j, j+1$ and $n = i, i+1$.

3. Design of Speed Control IC

The internal architecture of the proposed speed control IC for PMSM drive is shown in Fig. 3. The inputs of this control IC are speed command ω_r^*, rotor speed ω_r, flux angle θ_e, measured three-phase currents (i_a, i_b, i_c), and the output is PWM command. The speed control IC mainly includes a NFC-based speed controller, a current controller and coordinate transformation (CCCT), a SVPWM generation, frequency divider etc. The sampling frequency of current and speed control is designed with 16 kHz and 2kHz, respectively. The input clock is 50MHz and the frequency divider generates 50 Mhz (*Clk*), 25 MHz (*Clk-step*), 16 kHz (*Clk-cur*) and 2 kHz (*Clk-sp*) clock to supply all modules of the speed control IC. All modules in Fig.3 are described by Verilog HDL and simulated in ModelSim. An FSM (Finite state machine) is employed to model the NFC-based speed controller in Fig. 3 and it is shown in Fig. 4, which uses adders, multipliers and registers, etc. and manipulates 100 steps machine to carry out the overall computation. Although the algorithm of the NFC is high complexity, the FSM can give a very adequate modelling and easily be described by Verilog HDL. In Fig.4, steps s_0~s_5 execute the computation of reference model output; steps s_6~s_7 are for the computation of speed error and error change; steps s_8~s_{12} execute the fuzzification and look-up fuzzy table; s_{13}~s_{21} are for the defuzzification; s_{22}~s_{25} are the computation of current command; s_{26}~s_{89} describe the computation of RBF NN and Jacobian transformation; finally s_{90}~s_{99} execute the tuning of fuzzy rule parameters. The operation of each step in Fig.4 can be completed within 40ns (25 MHz clock); therefore total 100 steps only need 4μs operational times. In Fig. 3, the design of others components except NFC-based speed controller refer to [2]. The FPGA resource usages of CCCT, SVPWM and NFC controller in Fig.3 are 1,020 ALUTs (Adaptive Look-Up-Table), 1,326 ALUTs and 19,225 ALUTs, respectively.

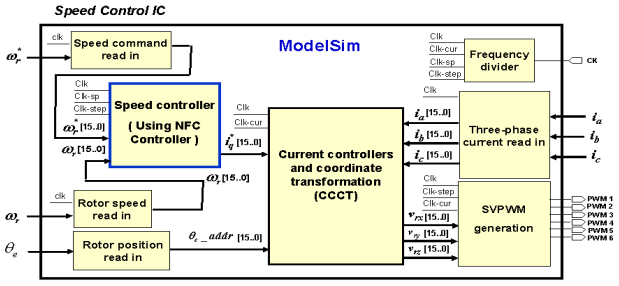

Fig. 3 Internal circuit of the proposed speed control IC for PMSM drive

4. Simulation Results

The co-simulation architecture for NFC-based speed control of PMSM drive is shown in Fig.5. The PMSM, inverter and speed command are implemented in Simulink program, and the speed control IC of PMSM drive is realized in Verilog HDL code using ModelSim program. The SimPowerSystem blockset in the Simulink executes the PMSM and the inverter. The EDA simulator link for ModelSim executes the co-simulation using Verilog HDL code running in ModelSim program. The designed PMSM parameters used in simulation of Fig.5 are that pole pairs is 4, stator phase resistance is 1.3Ω, stator inductance is 6.3mH, inertia is J=0.000108 kg*m^2 and friction factor is F=0.0013 N*m*s.

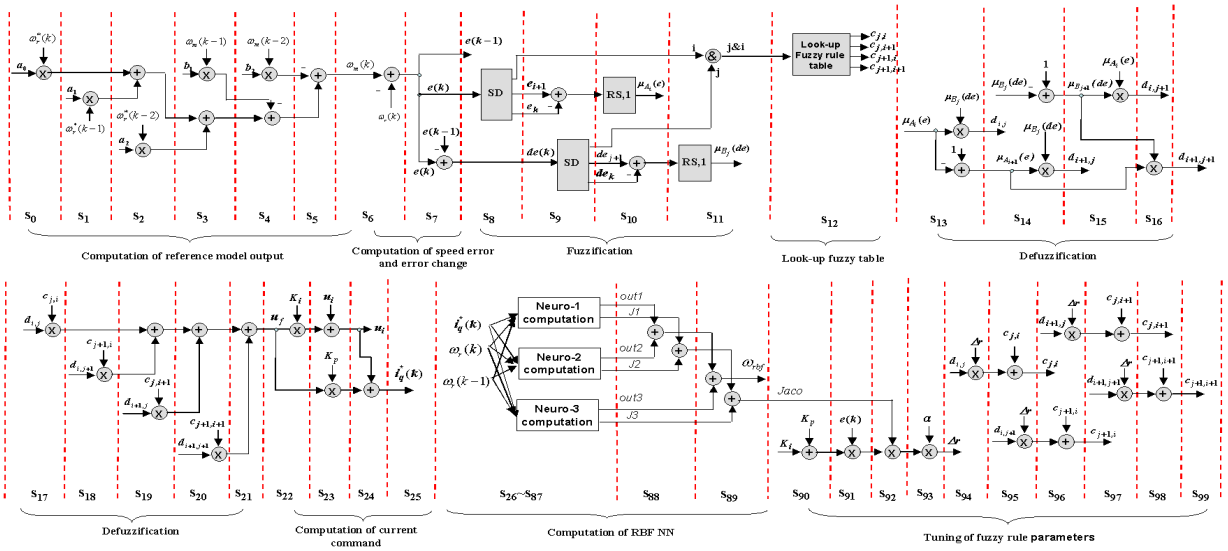

Fig. 4 State diagram of an FSM for describing the NFC in speed loop controller of PMSM drive

The simulation is used to evaluate the effectiveness of the proposed control algorithm. Three tested cases with different PMSM parameters are considered to evaluate the controller performance, in which

Case 1: (Normal-load condition)

$$J=0.000108, \quad F=0.0013 \tag{24}$$

Case II: (Light-load condition)

$$J=0.000108/3, \quad F=0.0013/3 \tag{25}$$

Case III: (Heavy-load condition)

$$J=0.000108*3, \quad F=0.0013*3 \tag{26}$$

The co-simulation is carried out in Fig.5. The control objective is to control the rotor speed of PMSM to track the output of the reference model. In the case of the FC design, the membership function and the fuzzy rule table are designed as Fig.6. In the NFC design, except that the parameters of the $c_{i,j}$ in Fig. 5 can be tuned using (23), others are the same as the FC. Herein, the learning rate α is set as 0.03. Square wave with period of 0.05 second and magnitude of 300rpm is used as a tested input command. To compare the tracking performance of the aforementioned two controllers at various system conditions, the system parameters are initially designed at the normal-load condition (Case I) and the controller is adopted by FC only, and the simulation result is shown in Fig. 7 which presents a good following response. However, when the system parameters change to the light-load (Case II) and heavy-load (Case III) condition, the results in Figs. 8~9 show that the response becomes worse with oscillation occurred in light-load condition and slow response occurred in heavy-load condition. To cope with this problem, a NFC is adopted and its simulation results are shown in Figs. 10~11. Due to the $c_{i,j}$ parameters in FC can be tuned to reduce the error between the rotor speed and the output of reference model, the rotor speed can accurately track well in Figs. 10~11. Therefore, the simulation results in Figs. 7 to 11 demonstrate that the proposed NFC-based speed control IC for PMSM drive is effective and robust.

5. Conclusions

This study has been presented a NFC-based speed control IC for PMSM drive and successfully demonstrated its performance through co-simulation by using Simulink and ModelSim. After confirming the effective of Verilog HDL code of speed control IC, we will realize this code in the experimental FPGA-based PMSM drive system for further verifying its function in the future work.

6. References

[1] B.K. Bose, "Expert system, fuzzy logic, and neural network applications in power electronics and motion control," Proc. IEEE, vol. 82, no. 8, 1994, pp. 1303-1323.

[2] Y.S. Kung and M.H. Tsai, "FPGA-based speed control IC for PMSM drive with adaptive fuzzy control," IEEE Trans. on Power Electronics, vol. 22, no. 6, pp. 2476-2486, Nov. 2007.

[3] E. Monmasson and M. N. Cirstea, "FPGA design methodology for industrial control systems – a review" IEEE Trans. Ind. Electron., vol. 54, no.4, pp.1824-1842, Aug. 2007.

[4] M. F. Castoldi, G. R. C. Dias, M. L. Aguiar and V. O. Roda, "Chopper-Controlled PMDC motor drive using VHDL code," in Proceedings of the 5th Southern Conference on Programmable Logic, pp. 209~212, 2009.

[5] M. F. Castoldi and M. L. Aguiar, "Simulation of DTC strategy in VHDL code for induction motor control," in Proceedings of the IEEE International Symposium on Industrial Electronics (ISIE), pp.2248-2253, 2006.

[6] J. L´azaro, A. Astarloa, J. Arias, U. Bidarte and A. Zuloaga, "Simulink/Modelsim simulable VHDL PID core for industrial SoPC multiaxis controllers," in Proceedings of the IEEE Industrial Electronics 32nd Annual Conference (IECON), pp.3007-3011, 2006.

[7] Y. Li , J. Huo, X. Li, J. Wen, Y. Wang and B. Shan, "An open-loop sin microstepping driver based on FPGA and the Co-simulation of Modelsim and Simulink," in Proceedings of the International Conference on Computer, Mechatronics, Control and Electronic Engineering (CMCE), pp. 223-227, 2010.

[8] The Mathworks, Matlab/Simulink Users Guide, Application Program Interface Guide, 2004

[9] Modeltech, ModelSim Reference Manual, 2004.

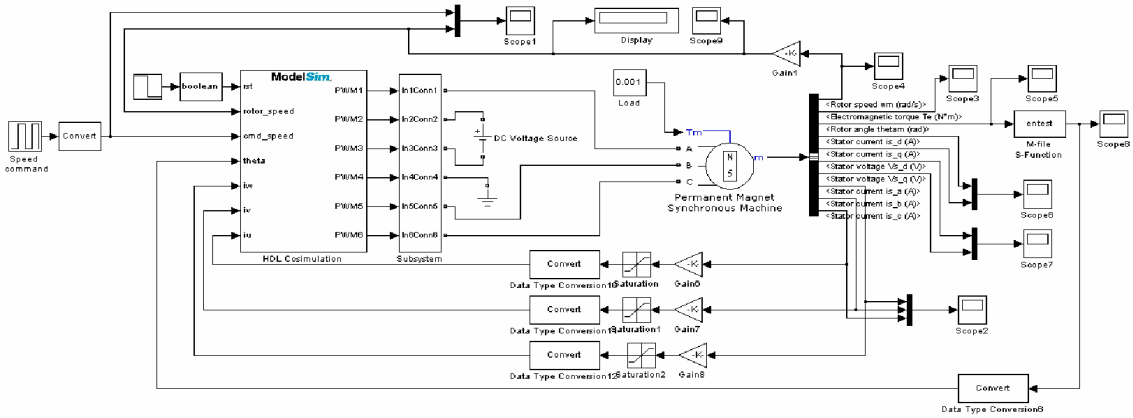

Fig. 5 Simulink and ModelSim co-simulation architecture for NFC-based speed control of PMSM drive

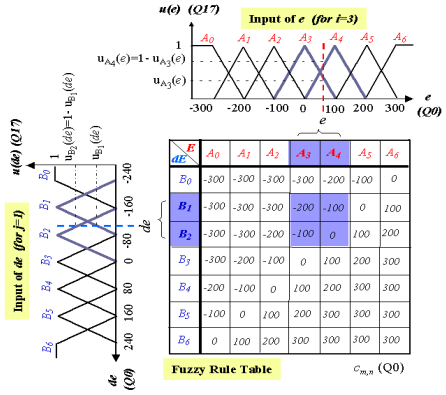

Fig. 6 Fuzzy membership function and the initial fuzzy-rule table

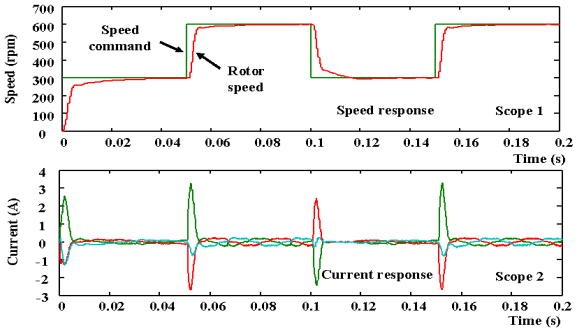

Fig. 7 Simulation result when FC is used and PMSM is operated at normal-load condition

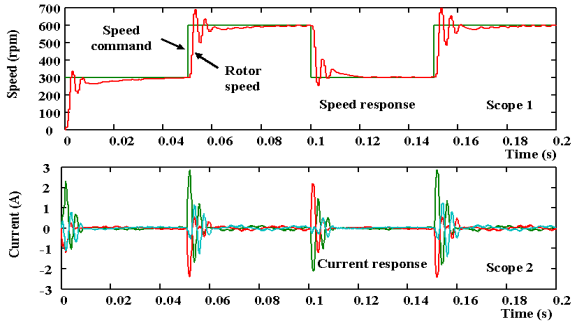

Fig. 8 Simulation result when FC is used and PMSM is operated at light-load condition

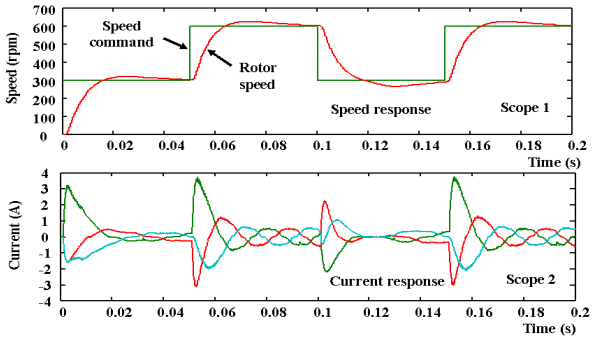

Fig. 9 Simulation result when FC is used and PMSM is operated at heavy-load condition

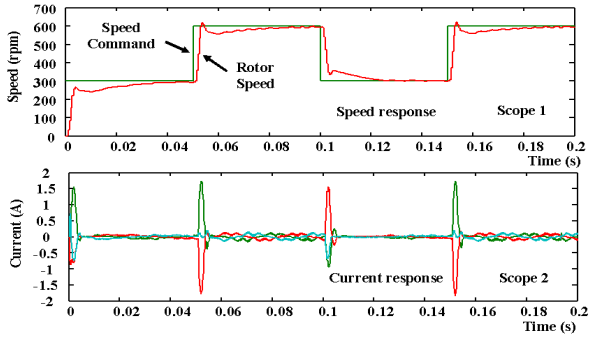

Fig. 10 Simulation result when NFC is used and PMSM is operated at light-load condition

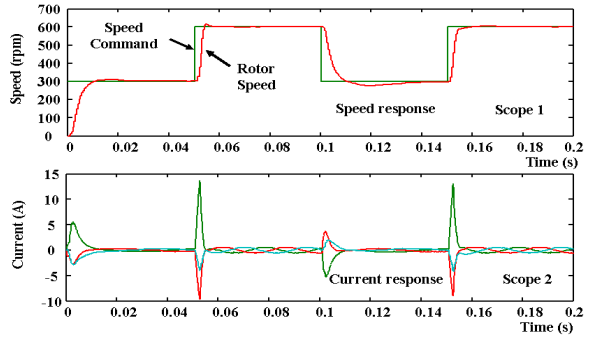

Fig. 11 Simulation result when NFC is used and PMSM is operated at heavy-load condition