第十屆台灣電力電子研討會暨展覽會
台灣　中壢市　2011 年 9 月 2 日

Design of Speed Control IC for PMSM Drive from Simulink/Modelsim Co-Simulation to FPGA Implementation

[1]Ying-Shieh Kung, [2]Nguyen Trung Hieu, [3]Nguyen Vu Quynh,
[4]Chung-Chun Huang and [5]Liang-Chiao Huang
[1, 2, 3] Department of Electrical Engineering, Southern Taiwan University, Tainan, Taiwan
[4, 5] Green Energy and Environment Research Laboratories, Industrial Technology Research Institute, Hsinchu, Taiwan
Email：[1] kung@mail.stut.edu.tw, [2] m992b202@webmail.stut.edu.tw, [3] vuquynh@lhu.edu.vn,
[4]cchuang@itri.org.tw, [5] liang.qiao@itri.org.tw

Abstract

Permanent magnetic synchronous motor (PMSM) has been increasingly used in many high performance applications due to its advantages of high power density, high power factor and efficiency. The design and implementation of a fuzzy-control based speed control IC for PMSM from Simulink/Modelsim co-simulation to field programmable gate array (FPGA) realization is presented in this paper. Firstly, a SVPWM scheme, vector control method and fuzzy controller are derived and applied in the speed control IC of PMSM drive. Secondly, the Very-High-Speed IC Hardware Description Language (VHDL) is adopted to describe the behavior of the aforementioned control algorithms. To evaluate the effectiveness and correctness of the proposed speed control IC, a co-simulation work performed by Matlab/Simulink and Modelsim is firstly conducted. Then, an experimental system by FPGA chip, Nios processor and motor driving board is set up to further validate the performance of the proposed speed control IC. Finally, the results in simulation and in experiment will be compared and discussed.

Key words: Simulink/Modelsim co-simulation; Fuzzy controller; VHDL, FPGA, PMSM, SVPWM.

I. Introduction

PMSM has been used in many automation fields such as robot, metal cutting machines, precision machining, etc., because its advantages of superior power density, high performance motion control with fast speed and better accuracy. Therefore, PMSM's controller plays a very important role. Up to now, there are many control methods have been studied to control the speed of PMSM such as adaptive control, PID control, intelligent control etc. Many conventional controls use Digital Signal Processor (DSP) in most studies. Unfortunately, DSP suffers from long time of development and exhaust resources of CPU [1]. The novel technology of FPGA with great advantages of programmable hard-wired feature, fast computation ability, shorter design cycle, embedded processor and low power consumption and higher density on the other hand can provide an alternative solution for these issues and is more suitable for the implementation of Digital System [2-3] than conventional DSPs.

Recently, a co-simulation work by Electronic Design Automation (EDA) Simulator Link has been gradually applied to verify the effectiveness of the VHDL and Verilog code in the motor drive system [4-7]. The EDA Simulator Link [8] provides a co-simulation interface between Matlab/Simulink and HDL simulators ModelSim [9]. Using this software we can verify a VHDL, Verilog, or mixed-language implementation against your Simulink model or MATLAB algorithm [8]. Therefore, EDA Simulator Link allows us to use Matlab code and Simulink models as a test bench that generate stimulus for an HDL simulation and analyzes the simulation's response [8].

In this paper, a co-simulation by EDA Simulator Link is applied to fuzzy-control based speed control IC for PMSM drive. The Simulink/Modelsim co-simulation architecture is shown in Fig.1. The PMSM, inverter and speed command are performed in Simulink and the proposed speed control IC described by VHDL code is executed in ModelSim. Some simulation results via Simulink/Modelsim co-simulation environment will present the correctness of the aforementioned VHDL code. Further, those codes will be directly downloaded to the FPGA to validate its effectiveness by experiment.

II. System Description of PMSM Drive and Fuzzy Speed Controller Design

(A) The Mathematical Model of PMSM

The typical mathematical model of PMSM is derived in two-axis *d-q* synchronous rotating reference frame, as follows [10]:

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

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

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 inductances; ω_e is the rotating speed of magnet flux; K_E is the permanent magnet flux linkage.

The configuration of the current loop for a PMSM are shown in Fig.1, and it includes PI controller, *Clarke*, modified $Clarke^{-1}$, *Park*, $Park^{-1}$ coordinate transformation, SVPWM, current detection pulse signal detection of the encoder, etc. After using vector control (control i_d^* to 0 in Fig.1), it will make the non-linear and coupling characteristics of PMSM become decouple [10]. Thus, the torque magnitude control of PMSM is only need to control the current in the direction of q-axis. The transformations between stationary a-b-c frame, stationary α-β frame and synchronously rotating d-q frame are shown in Fig.2, where $\bar{f}_s$ is a space vector refer to current, voltage or flux. The rotor speed of PMSM is synchronous with the stator electrical speed, thus we have ω_e=(P/2)ω_r and θ_e=(P/2) θ_r, where ω_e is synchronous with the rotor

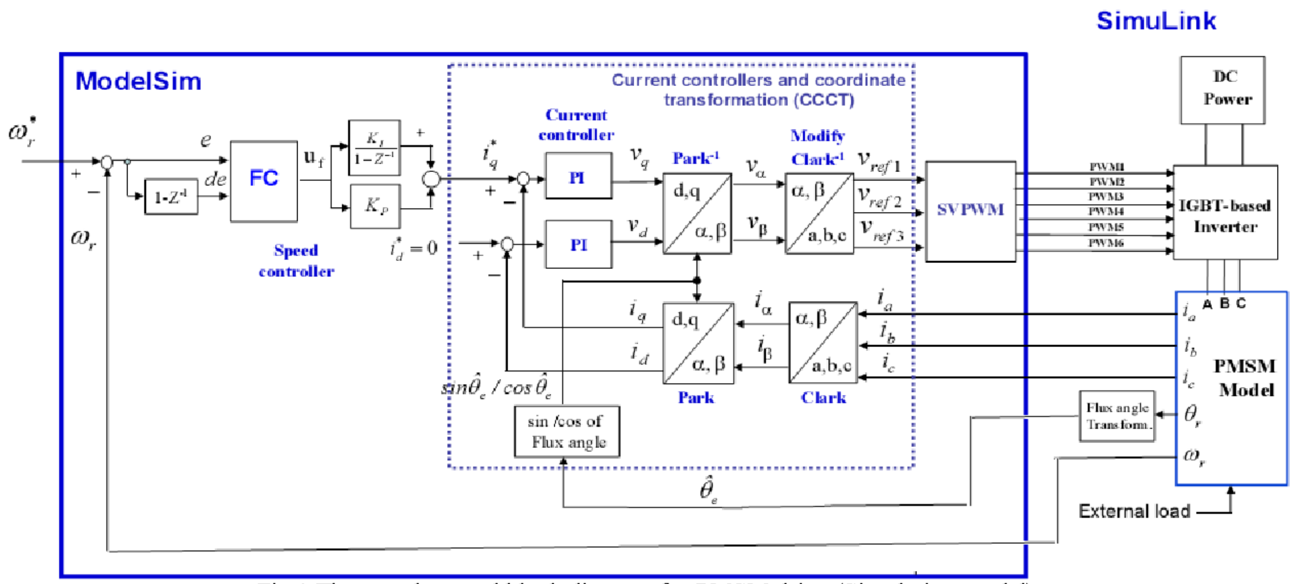


Fig.1 The speed control block diagram for PMSM drive (Simulation model)

speed, θ_e is electrical angle and P is number of magnetic pole. In Fig. 2, the coordinate transformations between stationary a-b-c frame, stationary α-β frame and synchronously rotating d-q frame are described as follows:

■ *Clarke* transformation: stationary a-b-c frame to stationary α-β frame.

$$\begin{bmatrix} f_\alpha \\ f_\beta \end{bmatrix} = \begin{bmatrix} \frac{2}{3} & \frac{-1}{3} & \frac{-1}{3} \\ 0 & \frac{1}{\sqrt{3}} & \frac{-1}{\sqrt{3}} \end{bmatrix} \begin{bmatrix} f_a \\ f_b \\ f_c \end{bmatrix} \qquad (3)$$

■ *Clarke*$^{-1}$ transformation: stationary α-β frame to stationary a-b-c frame.

$$\begin{bmatrix} f_a \\ f_b \\ f_c \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \frac{-1}{2} & \frac{\sqrt{3}}{2} \\ \frac{-1}{2} & \frac{-\sqrt{3}}{2} \end{bmatrix} \begin{bmatrix} f_\alpha \\ f_\beta \end{bmatrix} \qquad (4)$$

■ *Park* transformation: stationary α-β frame to synchronously rotating d-q frame.

$$\begin{bmatrix} f_{ds} \\ f_{qs} \end{bmatrix} = \begin{bmatrix} \cos\theta_e & \sin\theta_e \\ -\sin\theta_e & \cos\theta_e \end{bmatrix} \begin{bmatrix} f_\alpha \\ f_\beta \end{bmatrix} \qquad (5)$$

■ *Park*$^{-1}$ transformation: synchronously rotating d-q frame to stationary α-β frame.

$$\begin{bmatrix} f_\alpha \\ f_\beta \end{bmatrix} = \begin{bmatrix} \cos\theta_e & -\sin\theta_e \\ \sin\theta_e & \cos\theta_e \end{bmatrix} \begin{bmatrix} f_{ds} \\ f_{qs} \end{bmatrix} \qquad (6)$$

Therefore, after using vector control, the torque of PMSM is only dependent on i_q as the follow:

$$T_e = \frac{3P}{4} K_E i_q \triangleq K_t i_q \qquad (7)$$

When this condition is satisfied, the PMSM will be decoupled and we can control PMSM as control a DC motor. With the consideration of the mechanical load, the overall dynamic equation of PMSM drive system can be compute by:

$$J_m \frac{d}{dt}\omega_r + B_m \omega_r = T_e - T_L \qquad (8)$$

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

(B) Space Vector Pulse Width Modulation (SVPWM)

SVPWM is a special switching scheme of a 3-phase power converter with the six power transistors. According to the ON/OFF switching of upper transistors in 3-phase power converter, there have eight possible combinations. The eight vectors are called basic space vectors and they are denoted by U_0, U_{60}, U_{120}, U_{180}, U_{240}, U_{300}, O_{000} and O_{111}, which are shown in Fig.3. Using the *Clarke* transformation, the project values in α-β axis for six basic space vectors can be obtained and are also shown in Fig.3. The SVPWM technique is applied to approximate the reference voltage U_{out}, and it combines of the switching pattern with the basic space vectors. Therefore, the motor-voltage vector U_{out} will be located at one of the six sectors (S3, S1, S5, S4, S6, S2) at any given time. Thus, for any PWM period, it can be approximated by the vector sum of two vector components lying on the two adjacent basic vectors, as the following:

$$U_{out} = \frac{T_1}{T} U_X + \frac{T_2}{T} U_{X+60} + \frac{T_0 (O_{000}\ or\ O_{111})}{T}, \qquad (9)$$

where $T_0 = T - T_1 - T_2$ and T is half of PWM period.

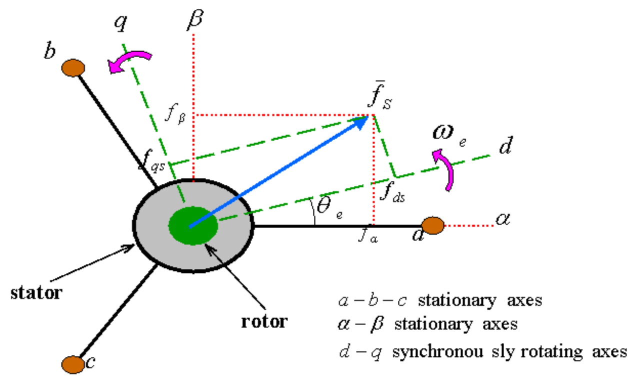


Fig. 2 Transformation between stationary axes and synchronously rotating axes

■ *Calculation of T_1 and T_2*

Any output voltage can be projected into each adjacent basic vector in SVPWM strategy. For example, the output voltage vector U_{out} in the sector S_3 can be the combination of U_0 and U_{60} shown in Fig.3. Therefore, the calculation of T_1 and T_2 can be shown as the followings,

$$U_0 = \frac{2}{3}V_{DC}\vec{\alpha} \tag{10}$$

$$U_{60} = \frac{1}{3}V_{DC}\vec{\alpha} + \frac{1}{\sqrt{3}}V_{DC}\vec{\beta} \tag{11}$$

If we substitute (10)~(11) into (9), we obtain

$$U_{out} = \frac{T_1}{T}U_0 + \frac{T_2}{T}U_{60} = \frac{T_1}{T}(\frac{2}{3}V_{DC}\vec{\alpha}) + \frac{T_2}{T}(\frac{V_{DC}}{3}\vec{\alpha} + \frac{V_{DC}}{\sqrt{3}}\vec{\beta}) \tag{12}$$
$$\triangleq V_\alpha\vec{\alpha} + V_\beta\vec{\beta}$$

where

$$V_\beta = \frac{T_2}{\sqrt{3}T}V_{DC} \tag{13}$$

$$V_\alpha = \frac{2}{3}\frac{T_1}{T}V_{DC} + \frac{T_2}{3T}V_{DC} = \frac{2}{3}\frac{T_1}{T}V_{DC} + \frac{1}{\sqrt{3}}V_\beta \tag{14}$$

$$\Rightarrow T_1 = \frac{T}{2V_{DC}}(3V_\alpha - \sqrt{3}V_\beta),\quad T_2 = \sqrt{3}\frac{T}{V_{DC}}V_\beta \tag{15}$$

In the similar way, the T_1 and T_2 in other sector can be derived and be rearranged in Table 1, which T_X, T_Y and T_Z are represented as the followings:

$$T_X = \sqrt{3}\frac{T}{V_{DC}}V_\beta \tag{16}$$

$$T_Y = \frac{T}{2V_{DC}}(3V_\alpha + \sqrt{3}V_\beta) \tag{17}$$

$$T_Z = \frac{T}{2V_{DC}}(-3V_\alpha + \sqrt{3}V_\beta) \tag{18}$$

If it is at the saturation condition $T_1 + T_2 > T$, the T_1 and T_2 should be modified as:

$$T_{1SAT} = T_1\frac{T}{T_1 + T_2} \tag{19}$$

$$T_{2SAT} = T_2\frac{T}{T_1 + T_2} \tag{20}$$

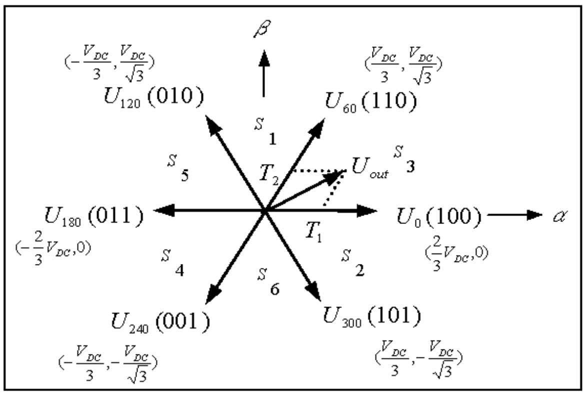


Fig. 3 Basic vector space and switching patterns

■ *Determination of the duty cycles and CMPRx*

After the calculation of T_1 and T_2, it has to re-transfer it to the duty cycles and CMPx values to generate the PWM output signals for controlling the power transistor switching time in Fig. 3. First, the duty cycles are defined as Ta_{on}, Tb_{on}, Tc_{on}, and calculated as the followings:

$$Ta_{on} = (T-T_1-T_2)/2 = T_0/2 \tag{21}$$

$$Tb_{on} = Ta_{on} + T_1 \tag{22}$$

$$Tc_{on} = Tb_{on} + T_2 \tag{23}$$

Then, the CMPR1~CMPR3 values can be obtained in Table 2 depend on the sector number. For example in S3 sector, CMPR1~CMPR3 are Ta_{on}, Tb_{on} and Tc_{on}, respectively and its output waveforms PWM1~PWM3 are depicted in Fig. 4 with the duty time at U_0 (100), U_{60} (110) and zero vector (O_0 and O_{111}) be T_1, T_2, T_0, respectively.

Table 1 T_1 and T_2 in all specific sectors

	S_3	S_1	S_5	S_4	S_6	S_2
T_1	$-T_Z$	T_Z	T_X	$-T_X$	$-T_Y$	T_Y
T_2	T_X	T_Y	$-T_Y$	T_Z	$-T_Z$	$-T_X$

Table 2 Assigning the duty cycle to CMPx in any sector

Sector	S_3	S_1	S_5	S_4	S_6	S_2
CMPR1	Ta_{on}	Tb_{on}	Tc_{on}	Tc_{on}	Tb_{on}	Ta_{on}
CMPR2	Tb_{on}	Ta_{on}	Ta_{on}	Tb_{on}	Tc_{on}	Tc_{on}
CMPR3	Tc_{on}	Tc_{on}	Tb_{on}	Ta_{on}	Ta_{on}	Tb_{on}

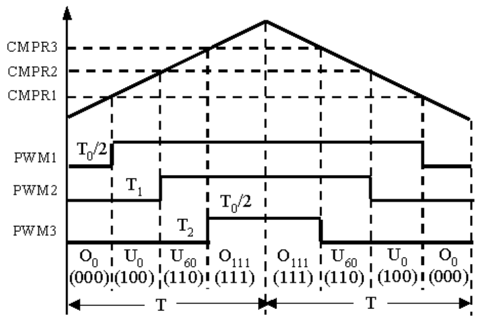


Fig. 4 Sector S_3 PWM patterns and duty cycle

■*Determination of the sector*

To determine the sector, we first modified the $Clarke^{-1}$ transformation as the following,

$$\begin{bmatrix} V_{rfx} \\ V_{rfy} \\ V_{rfz} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{1}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix} \begin{bmatrix} v_\beta \\ v_\alpha \end{bmatrix} \tag{24}$$

then, the output waveforms of V_{rfx}, V_{rfy} and V_{rfz} for sinusoid wave inputs (V_α, V_β) can be calculated and shown in Fig. 5. They can determine the sector according to the following rules:

If $V_{rfx} > 0$ then a=1 else a=0
If $V_{rfy} > 0$ then b=1 else b=0
If $V_{rfz} > 0$ then c=1 else c=0
Sector = a+2b+4c. (25)

From equations (16)~(18) and (24), we have.

$$\begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} = \frac{\sqrt{3}T}{V_{DC}} \begin{bmatrix} V_{rfx} \\ -V_{rfz} \\ -V_{rfy} \end{bmatrix}. \tag{26}$$

Therefore, T_x, T_y and T_z can be derived directly from V_{rfx}, V_{rfy} and V_{rfz}.

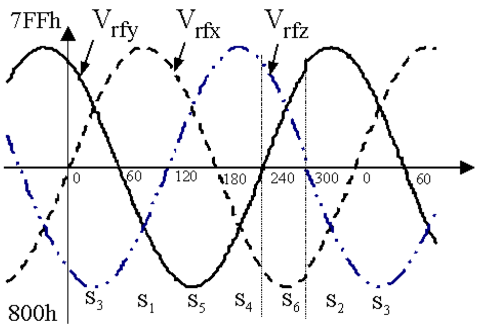


Fig. 5 3-phase sinusoid output waveform

■ *Computation procedures of SVPWM design*

SVPWM design is summary as following procedures:

Step 1: Determination of the sector according to the rule shown in (25), where V_{rfx}, V_{rfy}, V_{rfz} are the input signals of the SVPWM block circuit in Fig.1.

Step 2: Calculation of T_X, T_Y and T_Z from (26).

Step 3: Determination of T_1 and T_2 from Table 1. If it is at the saturation condition, we can use (19) and (20) to modify the T_1 and T_2.

Step 4: Determination of the duty cycle Ta_{on}, Tb_{on} Tc_{on} from (21)-(23).

Step 5: Assignment of the duty cycles to CMPR1, CMPR2 and CMPR3 from Table 2.

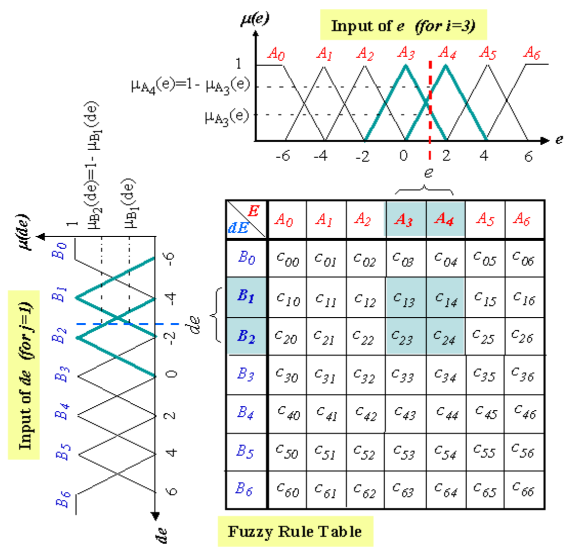
(C) Design Fuzzy controller (FC)

The fuzzy controller 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

$$e(n) = \omega_r^*(n) - \omega_r(n) \tag{27}$$

$$de(n) = e(n) - e(n-1) \tag{28}$$

and u_f represents the output of the fuzzy controller. The ω_r^* is the speed command. The design of the fuzzy controller is as follows:



E / dE	A_0	A_1	A_2	A_3	A_4	A_5	A_6
B_0	c_{00}	c_{01}	c_{02}	c_{03}	c_{04}	c_{05}	c_{06}
B_1	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}	c_{15}	c_{16}
B_2	c_{20}	c_{21}	c_{22}	c_{23}	c_{24}	c_{25}	c_{26}
B_3	c_{30}	c_{31}	c_{32}	c_{33}	c_{34}	c_{35}	c_{36}
B_4	c_{40}	c_{41}	c_{42}	c_{43}	c_{44}	c_{45}	c_{46}
B_5	c_{50}	c_{51}	c_{52}	c_{53}	c_{54}	c_{55}	c_{56}
B_6	c_{60}	c_{61}	c_{62}	c_{63}	c_{64}	c_{65}	c_{66}

Fig. 6 The designed fuzzy controller

■ Take e, de and u_f as the input and output variable of fuzzy controller and define their linguist values E and dE in Fig.2 by $\{A_0, A_1, A_2, A_3, A_4, A_5, A_6\}$ and $\{B_0, B_1, B_2, B_3, B_4, B_5, B_6\}$, respectively. Each linguist value of E and dE are based on the symmetrical triangular membership function.

■ Compute the membership degree of e and de. Figure 6 shows that the only two linguistic values are excited and gave a non-zero membership in any input value, and the membership degree $\mu_{A_i(e)}$ can be derived, in which the error e is located between e_i and e_{i+1}, two linguist values of A_i and A_{i+1} are excited, and the membership degree is obtained by

$$\mu_{A_i}(e) = \frac{e_{i+1} - e}{2} \quad \text{and} \quad \mu_{A_{i+1}}(e) = 1 - \mu_{A_i}(e) \tag{29}$$

where $e_{i+1} \triangleq -6 + 2*(i+1)$. Similar results can be obtained in computing the membership degree $\mu_{B_j(de)}$.

■ Select the initial fuzzy control rules, such as,

$$IF\ \ e\ \ is\ A_i\ \ and\ \ \Delta e\ is\ B_j\ \ THEN\ \ u_f\ \ is\ \ c_{j,i} \tag{30}$$

where i and j = 0~6, A_i and B_j are fuzzy number, and $c_{j,i}$ is real number.

■ Construct the fuzzy system $u_f(e,de)$ by using the singleton fuzzifier, product-inference rule, and central average defuzzifier method. Therefore, the (30) can be replaced by the following expression:

$$u_f(e,de) = \frac{\sum_{n=i}^{i+1}\sum_{m=j}^{j+1} c_{m,n}[\mu_{A_n}(e)*\mu_{B_m}(de)]}{\sum_{n=i}^{i+1}\sum_{m=j}^{j+1} \mu_{A_n}(e)*\mu_{B_m}(de)} \triangleq \sum_{n=i}^{i+1}\sum_{m=j}^{j+1} c_{m,n}*d_{n,m} \tag{31}$$

where $d_{n,m} \triangleq \mu_{A_n}(e)*\mu_{B_m}(de)$. And those $c_{m,n}$ denote the value of the singleton fuzzier.

III Simulink/Modelsim Co-Simulation of PMSM Drive

To confirm the effectiveness of the proposed architecture of fuzzy speed control block diagram for PMSM in Fig.1, simulation and experiment work are demonstrated in this and next section. In simulation work, the generation of SVPWM function is firstly tested, and its Simulink/Modelsim co-simulation block diagram is constructed in Fig.7. The SimPowerSystem blockset in Simulink executes the RC circuits and the EDA simulator link for ModelSim executes the co-simulation using VHDL code which runs the function of inverse Park, modified inverse Clarke and SVPWM algorithm. The data type herein adopts 12-bit length with Q11 format and 2's complement operation. The switching frequency and dead-band in SVPWM are designed with 16kHz and 1.2μs, respectively. In simulation, the IP/OP of Modelsim block in Fig.7 is that the input signal v_q is set to a voltage vary from 0.1 to 0.5 with period being 0.1 sec, the v_d is chosen by zero and the address is generated from 0~360 degree with increment by one, then PWM1~PWM6 outputs can be generated through the computation in Modelsim. Further, the PWM1,3,5 are serial connected to RC circuit (R=10Ω, C=47μF), respectively. The co-simulation is running in Simulink, and the results show the measured PWM1,3,5 as a saddle waveform in Fig.8.

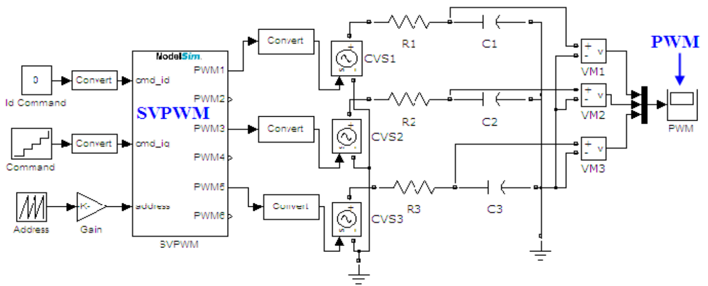


Fig. 7 Simulink/Modelsim co-simulation of SVPWM

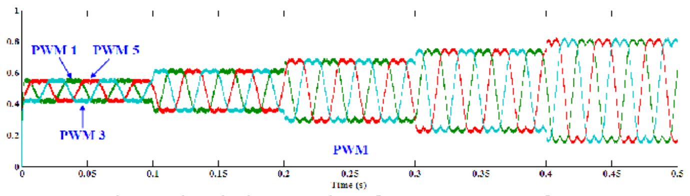


Fig. 8 Simulation results of SVPWM waveform

Having confirmed the effectiveness of the SVPWM generation, the dynamic performance of PMSM drive is evaluated while the current vector control and the fuzzy controller is applied in speed control loop of Fig. 1, and its Simulink/Modelsim co-simulation block diagram is constructed in Fig. 9. The SimPowerSystem blockset in the Simulink executes the PMSM and the inverter. The EDA simulator link for ModelSim executes the co-simulation using VHDL code running in ModelSim program with two works. The work-1 of ModelSim in Fig.9 performs the function of ~~[speed estimation and]~~ speed loop fuzzy controller, and the work-2 executes the function of current controller and coordinate transformation (CCCT) and SVPWM. The sampling frequency of current and speed control is designed with 16 kHz and 2kHz, respectively. The clocks of 50MHz and 12.5MHz will supply all works of ModelSim. Data type used in work-1 adopts 16-bit length with Q15 format, but in work-2 is 12-bit length with Q11 format. The numerical operations all consider 2's complement operation. The multiplier and adder apply Altera LPM (Library Parameterized Modules) standard. The FPGA resource usages in work-1 and work-2 of ModelSim in Fig.9 are 1,873 LEs (Logic Elements) and 0RAM bits as well as 1,767 LEs and 98,304 RAM bits respectively. In simulation, the designed PMSM parameters used in simulation of Fig.9 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. The simulation work of the step speed response is tested. The motor speeds command is designed with speed step varying from 0rpm→500rpm → 1000rpm → 1500rpm → 2000rpm → 1500rpm, and the results for actual rotor speed, two-axis currents (i_d and i_q) response and three-phase currents response are measured from scope-1 to scope-3 of Fig.9 and shown in Fig.10. From Fig.10(a), the rising time and steady-state value are about 14ms and near 0mm, which presents a good speed following response without overshoot occurred. From Fig.10(b), the value of measured current in d-axis (i_d) is approach to zero, and it reveal that a good decoupling effect after using vector current control in Fig.1.

IV. Experimental System and Results

After confirming the correctness of the proposed controller's VHDL codes by simulation, the codes are directly applied to the experimental FPGA-based PMSM drive system. The overall experimental system is depicted in Fig. 11. The main devices include a PMSM, a DE2 board with Altera CycloneII FPGA, a motor driver and a power supplier. The parameters of the PMSM are: r_s = 0.63Ω, L =2.77mH and 4 pole pairs. The input voltage, continuous current, rating torque, rating speed and continuous power of the PMSM are 220V, 12A, 2.3 N-m, 3000rpm and 750W, respectively. An incremental encoder with 2500ppr is attached at the PMSM. The Altera CycloneII EP2C35 chip adopted in the design possesses 33,216 LEs, maximum 475 available I/O pins, 483,840 RAM, and 35 embedded multipliers. The chip can be embedded with a Nios II multi-core processor that is equipped with several 32-bit CPU, flexibility of core size, 1 to 16 Mbytes of flash memory in the available memory chip, 1Mbyte SRAM, 16 M byte SDRAM, and 4Gbytes memory outside of the chip. In implementation, except the aforementioned controller's VHDL codes, the VHDL code for ADC and QEP interface circuits should be added and integrated; then downloaded into FPGA. In Fig.11(a), the FPGA will read the pulses from encoder and the motor current from ADC, through the computation of current controller and speed controller, the 6-channel PWM signals will be generated to drive the PMSM to a specified speed.

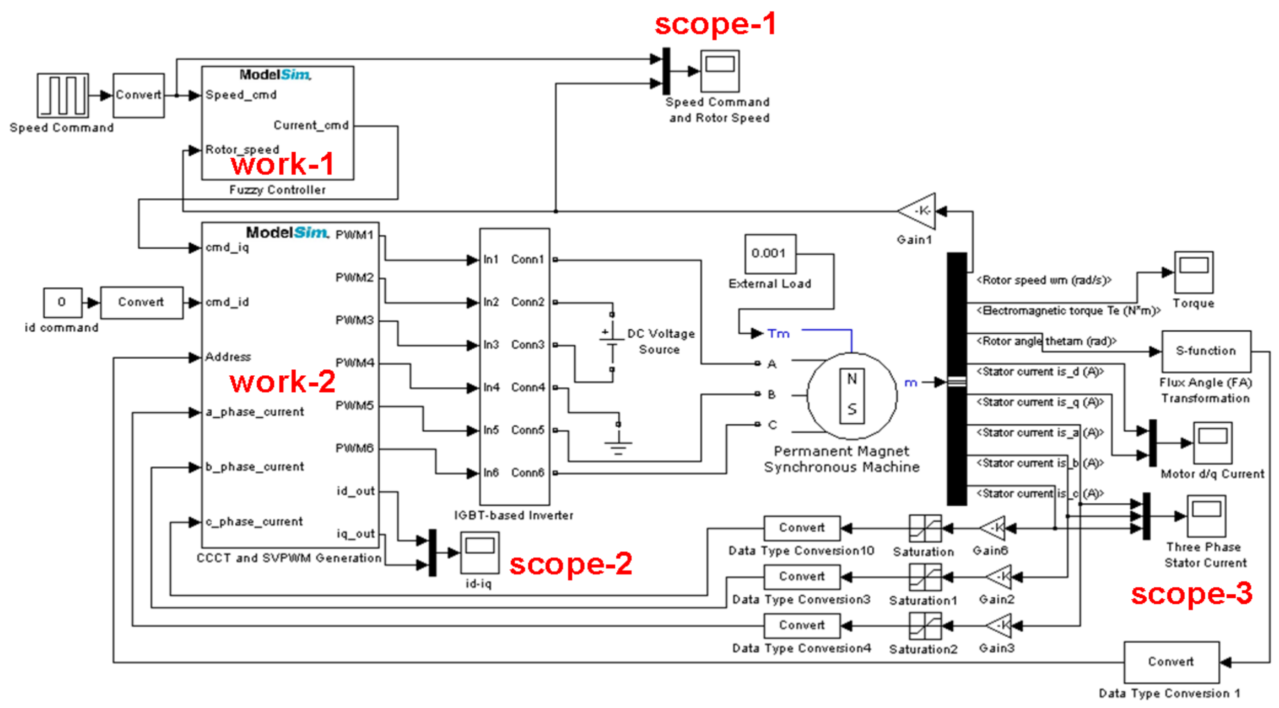


Fig. 9 Simulink/ModelSim co-simulation architecture for the speed control of PMSM drive

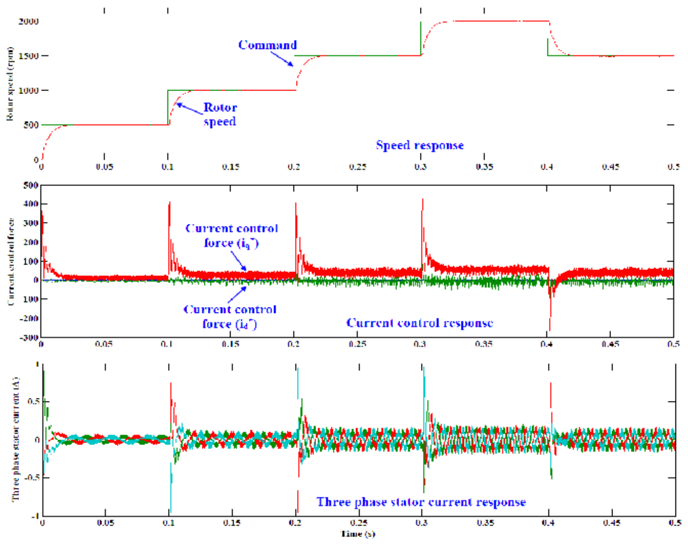
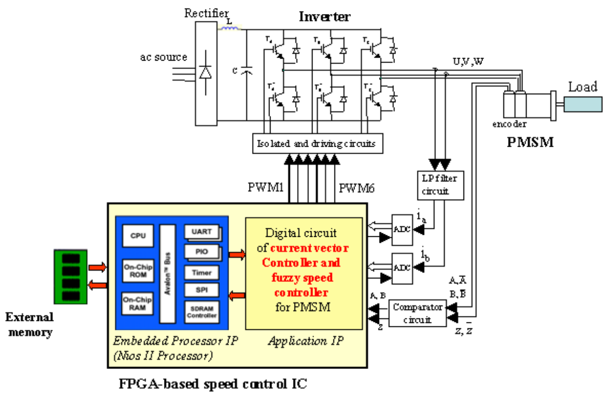
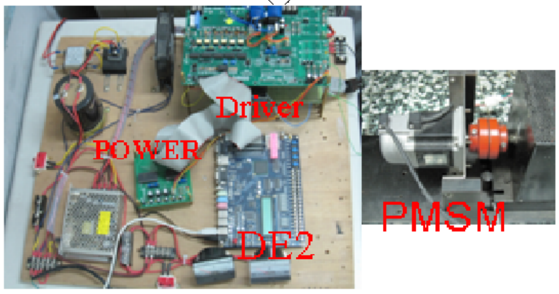


Fig. 10 Simulation results of step speed respond, two-axis current respond and three phase current response

Similar with speed command in simulation work, herein, the step speed command is designed by varying from 0rpm→500rpm→1000rpm→1500rpm→2000rpm→1500 rpm→0rpm with 1 second period time, and the results for actual rotor speed and control effort response are shown in Fig.12. From Fig.12(a), the rising time and steady-state value are about 250ms and near 0mm, which presents a good speed following response without overshoot occurred. The speed response trends in Fig.10(a) by simulation and in Fig.12(a) by experiment are very similar. The response time in simulation is faster than in experiment because we adopt different PMSM specification and controller parameter gains which can accelerate the overall simulation time in Simulink. However, the simulations and experiments shown in Fig.8, Fig.10 and Fig.12, demonstrate the effectiveness and correctness of the proposed FC-based speed control IC for PMSM.



(a)



(b)

Fig.11 Experimental System (a) block diagram (b) Real system

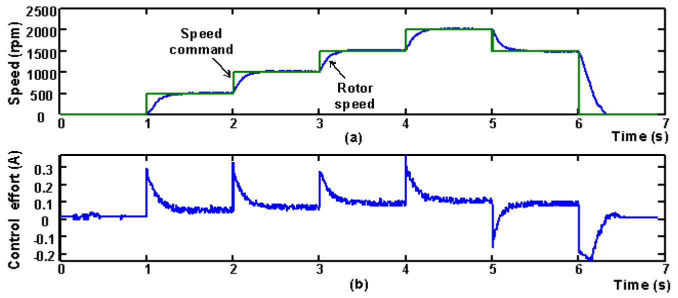


Fig. 12 Experimental results of (a) step speed respond (b) control effort response

V. Conclusion

This study has been successfully presented a speed control IC for PMSM drive and demonstrated its performance from Simulink/ModelSim co-simulation to FPGA implementation. Via this design methodology, the VHDL code of the speed control IC can be fast developed, and the risk in experimental development can be greatly reduced.

References

[1] Z.Zhou, T. Li, T. Takahahi and E. Ho, "FPGA realization of a high-performance servo controller for PMSM," *in Proceeding of the 9th IEEE Application Power Electronics conference and Exposition*, 2004, vol.3, pp. 1604-1609.
[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.
[10] Y.S. Kung, C.S. Chen, K.I. Wong, M.H. Tsai "Development of a FPGA-based control IC for PMSM drive with adaptive fuzzy control" *in Proceedings of the Industrial Electronics Society, IECON 2005. 31st Annual Conference of IEEE* pp 1544 – 1549, 2005