Research Article

FPGA Realization of Sensorless PMSM Speed Controller Based on Extended Kalman Filter

Ying-Shieh Kung,¹ Nguyen Vu Quynh,² Nguyen Trung Hieu,¹ and Jin-Mu Lin¹

¹ Department of Electrical Engineering, Southern Taiwan University of Science and Technology, 1 Nan-Tai Street, Yong-Kang District, Tainan City 710, Taiwan

² Department of Electrical Engineering, Lac Hong University, 10 Huynh Van Nghe Street, Buu Long District, Bien Hoa City 084, Dong Nai Province, Vietnam

Correspondence should be addressed to Ying-Shieh Kung; kung@mail.stust.edu.tw

Received 9 September 2013; Accepted 22 October 2013

Academic Editor: Teen-Hang Meen

Copyright © 2013 Ying-Shieh Kung et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Based on extended Kalman filter (EKF), the design and FPGA implementation of a sensorless control intellectual property (IP) for permanent magnet synchronous motor (PMSM) drive are presented in this paper. Firstly, the mathematical model for PMSM is derived and the vector control is built up. Secondly, the rotor flux angle (FA) and rotor speed which are estimated by using EKF are described. These estimated values are feedforwarded to the current loop for vector control and to the speed loop for speed control. Thirdly, the very-high-speed IC hardware description language (VHDL) is adopted to describe the behavior of the sensorless speed control IP which includes the circuits of space vector pulse width modulation (SVPWM), coordinate transformation, EKF, and PI controller. Finally, to evaluate the effectiveness and correctness of the proposed system, a cosimulation work performed by Simulink and ModelSim is firstly conducted. Then, an experimental system by FPGA chip and motor driving board is set up to further validate the performance of the proposed EKF-based sensorless speed control IP.

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. However, conventional motor control needs a speed sensor or an optical encoder to measure the rotor speed and feedback it to the controller for ensuring the precision speed control. Such sensor presents some disadvantages such as drive cost, machine size, reliability, and noise immunity. In recent years, a sensorless control without position and speed sensors for PMSM drive become a popular research topic in the literature [1–7]. Those sensorless control strategies have sliding mode observer (SMO), extended Kalman filter (EKF), reduced-order EKF, and so forth. The EKF is basically a full-order stochastic observer for the recursive optimum state estimation of a nonlinear dynamic system in real time by using signals that are in noisy environment [7]. Comparing with SMO, EKF can directly

estimate the angular speed and it has high convergence rate which can give a more rapid speed response [5]. However, EKF requires heavy online 4×4 matrix computing; therefore, the complex computation becomes a challenge for a fixed-point processor system. In realization, a fixed-point processor using digital signal processor (DSP) and field programmable gate array (FPGA) both can provide a good solution in this issue. Particularly, FPGA with programmable hardware feature, fast computation ability, shorter design cycle, embedding processor, low power consumption, and higher density is better for the implementation of the digital system [8, 9] than DSP.

Recently, a cosimulation 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 [10–12]. The EDA Simulator Link [13] provides a cosimulation interface between MATLAB or Simulink and HDL simulators-ModelSim [14]. Using it you

can verify a VHDL, Verilog, or mixed-language implementation against your Simulink model or MATLAB algorithm [13]. 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 [13]. In this paper, a cosimulation by EDA Simulator Link is applied to sensorless speed control for PMSM drive. The PMSM, inverter, and speed command are performed in Simulink and the EKF algorithm; current vector controller and speed PI controller described by VHDL code are executed in ModelSim. After successful verification in simulation, an FPGA-based experimental system is established with configuration as in Figure 1 for realizing the proposed IP code again, and experiment results will validate the effectiveness of the sensorless speed control system of PMSM drive.

2. System Description of PMSM Drive and Sensorless Speed Control

The sensorless speed control block diagram for PMSM drive is shown in Figure 1. The modeling of PMSM, the EKF-based rotor flux angle (FA) and rotor speed estimation, and the PI controller are introduced as follows.

2.1. Mathematical Model of PMSM. The mathematical model of a PMSM is described, in two-axis d - q synchronous rotating reference frame, as follows:

$$\begin{aligned} \frac{di_d}{dt} &= -\frac{r_s}{L_d} i_d + \omega_e \frac{L_q}{L_d} i_q + \frac{1}{L_d} v_d, \\ \frac{di_q}{dt} &= -\omega_e \frac{L_d}{L_q} i_d - \frac{r_s}{L_q} i_q - \omega_e \frac{\lambda_f}{L_q} + \frac{1}{L_q} v_q, \end{aligned} \quad (1)$$

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

The current loop control of PMSM drive in Figure 1 is based on a vector control approach. That is, if the i_d is forced to 0, the PMSM will be decoupled, and controlling a PMSM is like controlling a DC motor. After decoupling, the motor's torque is proportional to i_q :

$$T_e = \frac{3N_p}{4} \lambda_f i_q \triangleq K_t i_q. \quad (2)$$

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

$$J_m \frac{d}{dt} \omega_r + B_m \omega_r = T_e - T_L, \quad (3)$$

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

2.2. Extended Kalman Filter (EKF). For a motor stochastic nonlinear equation, it can be written in the following form:

$$\dot{x}(t) = f[x(t)] + Bu(t) + \sigma(t), \quad (4)$$

$$y(t) = h[x(t)] + \mu(t), \quad (5)$$

where $x(t)$, $u(t)$, and $y(t)$ are system state, system input, and system output, respectively. The $\sigma(t)$ and $\mu(t)$ represent system noise and measurement noise which are zero-mean white Gaussian distribution with covariance $Q(t)$ and $R(t)$, respectively. Once a nominal solution to a motor nonlinear equation can be found in (4)-(5), the linearized perturbation equation is

$$\delta \dot{x}(t) = F(x(t)) \delta x(t) + B \delta u(t) + \sigma(t), \quad (6)$$

$$\delta y(t) = H(x(t)) \delta x + \mu(t), \quad (7)$$

where the Jacobian and output matrices are defined as follows:

$$\begin{aligned} F(x(t)) &= \left. \frac{\partial f}{\partial x} \right|_{x=x(t)}, \\ H(x(t)) &= \left. \frac{\partial h}{\partial x} \right|_{x=x(t)}. \end{aligned} \quad (8)$$

After discretization with sampling time T_c , (6) becomes

$$\begin{aligned} x(t_n) &= \Phi(t_n, t_{n-1}, x(t_{n-1})) x(t_{n-1}) \\ &+ \int_{t_{n-1}}^{t_n} \Phi(t_n, t_{n-1}, x(t_{n-1})) B d\tau \cdot u(t_{n-1}) + v(t_{n-1}), \end{aligned} \quad (9)$$

where $\Phi(t_n, t_{n-1}, x(t_{n-1}))$ is an exponential matrix and its computation can be simplified by

$$\Phi(t_n, t_{n-1}, x(t_{n-1})) \cong I + FT_c. \quad (10)$$

Further,

$$\int_{t_{n-1}}^{t_n} \Phi(t_n, t_{n-1}, x(t_{n-1})) B d\tau \cong BT_c. \quad (11)$$

Therefore, the discrete model of (6)-(7) becomes

$$x(n) = (I + FT_c) x(n-1) + BT_c \cdot u(n-1) + v(n-1), \quad (12)$$

$$y(n) = H \cdot x(n) + \xi(n-1), \quad (13)$$

where $t_n = nT_c$ and $t_{n-1} = (n-1)T_c$.

The EKF is an optimal estimator which searches the cost function $J = \sum_{n=1}^m E\{\tilde{x}^2(n)\}$ at the least square sense. The $\tilde{x}(n)$ is defined by $\tilde{x}(n) = \hat{x}(n) - x(n)$, which is the difference of estimation of state $\hat{x}(n)$ and system state $x(n)$. The EKF algorithm is described by the following two-step recursive equations.

FIGURE 1: Architecture of FPGA-based sensorless PMSM drive system.

Step 1 (prediction step). From (4) and using a simple rectangular integration,

$$\hat{x}_{n|n-1} = \hat{x}_{n-1} + (f[\hat{x}_{n-1}] + B \cdot u_{n-1}) T_c \quad (14)$$

or from (12)

$$\hat{x}_{n|n-1} = \Phi_{n-1} \hat{x}_{n-1} + B T_c \cdot u_{n-1}. \quad (15)$$

The covariance is updated by

$$P_{n|n-1} = \Phi_{n-1} P_{n-1} \Phi_{n-1}^T + Q_d. \quad (16)$$

Step 2 (innovation step). Consider

$$\hat{x}_n = \hat{x}_{n|n-1} + K_n (y_n - H \hat{x}_{n|n-1}), \quad (17)$$

$$P_n = P_{n|n-1} - K_n H P_{n|n-1}. \quad (18)$$

The Kalman gain is calculated by

$$K_n = P_{n|n-1} H^T [H P_{n|n-1} H^T + R]^{-1}. \quad (19)$$

Figure 2 shows the procedure of EKF with two step above, after measurement update, it will be changed to time update.

2.3. Design of the Sensorless PMSM Based on EKF. The circuit equation of PMSM on the d - q rotating coordinate in (1) is reformulated as

$$\begin{bmatrix} v_d \\ v_q \end{bmatrix} = \begin{bmatrix} r_s + sL_s & -\omega_e L_s \\ \omega_e L_s & r_s + sL_s \end{bmatrix} \begin{bmatrix} i_d \\ i_q \end{bmatrix} + \begin{bmatrix} 0 \\ \omega_e \lambda_f \end{bmatrix}, \quad (20)$$

where $L_s \triangleq L_d = L_q$. Transforming (20), the circuit equation of PMSM on the α - β fixed coordinate can be derived by the following equation:

$$\begin{bmatrix} v_\alpha \\ v_\beta \end{bmatrix} = \begin{bmatrix} r_s + sL_s & 0 \\ 0 & r_s + sL_s \end{bmatrix} \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} + \omega_e \lambda_f \begin{bmatrix} -\sin \theta_e \\ \cos \theta_e \end{bmatrix}, \quad (21)$$

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; and s is differential operator. If we choose the $x(t) = [i_\alpha, i_\beta, \omega_e, \theta_e]^T$ as the state variable of the PMSM mathematical model, then (21) can be expanded to the following formulation:

$$\frac{d}{dt} \begin{bmatrix} i_\alpha \\ i_\beta \\ \omega_e \\ \theta_e \end{bmatrix} = \begin{bmatrix} -\frac{r_s i_\alpha}{L_s} + \frac{\omega_e \lambda_f}{L_s} \sin \theta_e \\ -\frac{r_s i_\beta}{L_s} - \frac{\omega_e \lambda_f}{L_s} \cos \theta_e \\ 0 \\ \omega_e \end{bmatrix} + \begin{bmatrix} \frac{1}{L_s} & 0 \\ 0 & \frac{1}{L_s} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} v_\alpha \\ v_\beta \end{bmatrix} \triangleq f + B \begin{bmatrix} v_\alpha \\ v_\beta \end{bmatrix}, \quad (22)$$

$$\begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} i_\alpha \\ i_\beta \\ \omega_e \\ \theta_e \end{bmatrix} \triangleq h, \quad (23)$$

FIGURE 2: Demonstration of the EKF operation.

where $x(t) = [i_\alpha \ i_\beta \ \omega_e \ \theta_e]^T$, $y(t) = [i_\alpha \ i_\beta]$, and $u(t) = [v_\alpha \ v_\beta]^T$. From (8) and (22), the Jacobian and output matrices can be obtained:

$$F(x(t)) = \left. \frac{\partial f}{\partial x} \right|_{x=x(t)} = \begin{bmatrix} -\frac{r_s}{L_s} & 0 & \frac{\lambda_f}{L_s} \sin \theta_e & \frac{\omega_e \lambda_f}{L_s} \cos \theta_e \\ 0 & -\frac{r_s}{L_s} & -\frac{\lambda_f}{L_s} \cos \theta_e & \frac{\omega_e \lambda_f}{L_s} \sin \theta_e \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \quad (24)$$

$$H(x(t)) = \left. \frac{\partial h}{\partial x} \right|_{x=x(t)} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}. \quad (25)$$

Further, substituting (24) into (10), the exponential matrix is shown as follows:

$$\begin{aligned} & \Phi(t_n, t_{n-1}, x(t_{n-1})) \\ & \cong I + FT_c \\ & = \begin{bmatrix} 1 - \frac{r_s T_c}{L_s} & 0 & \frac{\lambda_f T_c}{L_s} \sin \theta_e & \frac{\omega_e \lambda_f T_c}{L_s} \cos \theta_e \\ 0 & 1 - \frac{r_s T_c}{L_s} & -\frac{\lambda_f T_c}{L_s} \cos \theta_e & \frac{\omega_e \lambda_f T_c}{L_s} \sin \theta_e \\ 0 & 0 & 1 & 0 \\ 0 & 0 & T_c & 1 \end{bmatrix} \\ & \triangleq \begin{bmatrix} G & 0 & \phi_{13} & \phi_{14} \\ 0 & G & \phi_{23} & \phi_{24} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & T_c & 1 \end{bmatrix}. \end{aligned} \quad (26)$$

As the results, the EKF algorithm in (14)–(19) can be carried out to estimate the state value. The initial values of Q_d , R , and P_0 need to be chosen. Through the recursive calculation, the

state value of $\hat{x}(n) = [\hat{i}_\alpha(n), \hat{i}_\beta(n), \hat{\omega}_e(n), \hat{\theta}_e(n)]^T$ is estimated at each sampling period; then the rotor speed can be derived by

$$\hat{\omega}_r(n) = \frac{\hat{\omega}_e(n)}{N_p}. \quad (27)$$

Finally, a summary for the estimation of rotor FA and rotor speed based on EKF is presented as follows.

Step 1. Set the initial values of Q_d , R , P_0 , and $n = 1$.

Step 2. Measure the values of $i_\alpha(n)$, $i_\beta(n)$, $v_\alpha(n)$, and $v_\beta(n)$ from PMSM system.

Step 3. Estimate the temporary state variables from (14). In addition, refer to (22), and the scalar form of the prediction equation can be expressed as follows:

$$\begin{aligned} \hat{i}_\alpha(n | n-1) &= \left(1 - T_c \frac{r_s}{L_s}\right) \hat{i}_\alpha(n-1) \\ &+ \frac{\hat{\omega}_e(n-1) T_c \lambda_f \sin(\hat{\theta}_e(n-1))}{L_s} \\ &+ \frac{T_c v_\alpha(n-1)}{L_s}, \\ \hat{i}_\beta(n | n-1) &= \left(1 - T_c \frac{r_s}{L_s}\right) \hat{i}_\beta(n-1) \\ &- \frac{\hat{\omega}_e(n-1) T_c \lambda_f \cos(\hat{\theta}_e(n-1))}{L_s} \\ &+ \frac{T_c v_\beta(n-1)}{L_s}, \\ \hat{\omega}_e(n | n-1) &= \hat{\omega}_e(n-1), \\ \hat{\theta}_e(n | n-1) &= \hat{\theta}_e(n-1) + \hat{\omega}_e(n-1) T_c. \end{aligned} \quad (28)$$

Step 4. Calculate the Φ_{n-1} from (26).

FIGURE 3: Internal architecture of a sensorless speed control IP in FPGA.

Step 5. Obtain the temporary covariance matrix $P_{n|n-1}$ from (16). Because matrix $P_{n|n-1}$ is symmetric matrix by $p_{ij} = p_{ji}$, it can be chosen by the following form:

$$P_{n|n-1} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \\ p_{41} & p_{42} & p_{43} & p_{44} \end{bmatrix}. \quad (29)$$

Hence, from (16), individual elements of $P_{n|n-1}$ could be updated as follows:

$$\begin{aligned} p_{11} &\Leftarrow G^2 p_{11} + 2G\phi_{14}p_{14} + 2G\phi_{13}p_{13} + 2\phi_{14}\phi_{13}p_{34} \\ &\quad + \phi_{14}^2 p_{44} + \phi_{13}^2 p_{33} + q_{11}, \\ p_{12} &\Leftarrow G^2 p_{12} + G\phi_{14}p_{24} + G\phi_{13}p_{23} \\ &\quad + \phi_{23}(Gp_{13} + \phi_{13}p_{33} + \phi_{14}p_{34}) \\ &\quad + \phi_{24}(Gp_{14} + \phi_{13}p_{34} + \phi_{14}p_{44}), \\ p_{13} &\Leftarrow Gp_{13} + \phi_{13}p_{33} + \phi_{14}p_{34}, \\ p_{14} &\Leftarrow (Gp_{13} + \phi_{13}p_{33} + \phi_{14}p_{34})T_c \\ &\quad + (Gp_{14} + \phi_{13}p_{34} + \phi_{14}p_{44}), \\ p_{22} &\Leftarrow 2G(\phi_{23}p_{23} + \phi_{24}p_{24}) + G^2 p_{22} + \phi_{23}^2 p_{33} \\ &\quad + \phi_{24}^2 p_{44} + 2\phi_{23}\phi_{24}p_{34} + q_{22}, \end{aligned}$$

$$\begin{aligned} p_{23} &\Leftarrow Gp_{23} + \phi_{23}p_{33} + \phi_{24}p_{34}, \\ p_{24} &\Leftarrow (Gp_{23} + \phi_{23}p_{33} + \phi_{24}p_{34})T_c + Gp_{24} \\ &\quad + \phi_{24}p_{44} + \phi_{23}p_{34}, \\ p_{33} &\Leftarrow p_{33} + q_{33}, \\ p_{34} &\Leftarrow T_c p_{33} + p_{34}, \\ p_{44} &\Leftarrow p_{44} + T_c^2 p_{33} + 2T_c p_{34} + q_{44}. \end{aligned}$$

(30)

Step 6. Calculate the Kalman gain from (19). Using (25) and (29), the formulation is further simplified as follows. Firstly,

$$\begin{aligned} &[HP_{n|n-1}H^T + R]^{-1} \\ &= \left[\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \\ p_{41} & p_{42} & p_{43} & p_{44} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \right. \\ &\quad \left. + \begin{bmatrix} R_{11} & 0 \\ 0 & R_{22} \end{bmatrix} \right]^{-1} \\ &= \left[\begin{bmatrix} p_{11} & p_{12} \\ p_{21} & p_{22} \end{bmatrix} + \begin{bmatrix} R_{11} & 0 \\ 0 & R_{22} \end{bmatrix} \right]^{-1}, \end{aligned}$$

(31)

FIGURE 4: State diagram of FSM for describing the EKF-based algorithm.

where R represents the uncertainties of measurement variable which are very difficult to know. Hence, R is often chosen as unity matrix for simplifying the task but does not affect the outcome (31). Since $P_{12} = P_{21}$, (31) can be rewritten in the following form:

$$\begin{aligned} & [HP_{n|n-1}H^T + R]^{-1} \\ &= \frac{1}{P_{11}P_{22} + P_{11} + P_{22} - P_{12}^2 + 1} \begin{bmatrix} P_{22} + 1 & -P_{12} \\ -P_{12} & P_{11} + 1 \end{bmatrix}. \end{aligned} \quad (32)$$

To normalize the input value and to prevent the numerical overflow condition which occurred during computation, (32) is normalized and those values are all divided by 4. Therefore, (32) becomes

$$[HP_{n|n-1}H^T + R]^{-1} = \frac{1}{M} \begin{bmatrix} \frac{P_{22} + 1}{4} & \frac{-P_{12}}{4} \\ \frac{-P_{12}}{4} & \frac{P_{11} + 1}{4} \end{bmatrix} \triangleq \begin{bmatrix} t_{11} & t_{12} \\ t_{21} & t_{22} \end{bmatrix}, \quad (33)$$

where $M = (p_{11}p_{22} + p_{11} + p_{22} - p_{12}^2 + 1)/4$. Then, from (19), the Kalman gain can be calculated by

$$\begin{aligned} K_n &\triangleq \begin{bmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \\ k_{31} & k_{32} \\ k_{41} & k_{42} \end{bmatrix} \\ &= \begin{bmatrix} p_{11} & p_{12} \\ p_{21} & p_{22} \\ p_{31} & p_{32} \\ p_{41} & p_{42} \end{bmatrix} \begin{bmatrix} t_{11} & t_{12} \\ t_{21} & t_{22} \end{bmatrix} \\ &= \begin{bmatrix} p_{11}t_{11} + p_{12}t_{21} & p_{11}t_{12} + p_{12}t_{22} \\ p_{21}t_{11} + p_{22}t_{21} & p_{21}t_{12} + p_{22}t_{22} \\ p_{31}t_{11} + p_{32}t_{21} & p_{31}t_{12} + p_{32}t_{22} \\ p_{41}t_{11} + p_{42}t_{21} & p_{41}t_{12} + p_{42}t_{22} \end{bmatrix}. \end{aligned} \quad (34)$$

FIGURE 6: State diagram of FSM for (a) the calculation of matrix K_n and (b) the update of covariance matrix P_n .

Step 7. Tune the present state variables from (17), and its scalar form is shown as follows:

$$\begin{aligned}\hat{i}_\alpha(n) &= \hat{i}_\alpha(nm-1) + k_{11}\tilde{i}_\alpha(n) + k_{12}\tilde{i}_\beta(n), \\ \hat{i}_\beta(n) &= \hat{i}_\beta(nm-1) + k_{21}\tilde{i}_\alpha(n) + k_{22}\tilde{i}_\beta(n),\end{aligned}$$

$$\hat{\omega}_e(n) = \hat{\omega}_e(nm-1) + k_{31}\tilde{i}_\alpha(n) + k_{32}\tilde{i}_\beta(n),$$

$$\hat{\theta}_e(n) = \hat{\theta}_e(n | n-1) + k_{41}\tilde{i}_\alpha(n) + k_{42}\tilde{i}_\beta(n),$$

(35)

where k_{ij} is the element in Kalman gain K_n . $\tilde{i}_\alpha(n) = i_\alpha(n) - \hat{i}_\alpha(n | n-1)$ and $\tilde{i}_\beta(n) = i_\beta(n) - \hat{i}_\beta(n | n-1)$.

FIGURE 7: The Simulink/ModelSim cosimulation architecture for sensorless speed control of PMSM drive.

Step 8. Update the present covariance matrix P_n from (18), where $K_n HP_{n|n-1}$ can be directly obtained as follows:

$$K_n HP_{n|n-1} = \begin{bmatrix} k_{11}P_{11} + k_{12}P_{21} & k_{11}P_{12} + k_{12}P_{22} & k_{11}P_{13} + k_{12}P_{23} & k_{11}P_{14} + k_{12}P_{24} \\ k_{21}P_{11} + k_{22}P_{21} & k_{21}P_{12} + k_{22}P_{22} & k_{21}P_{13} + k_{22}P_{23} & k_{21}P_{14} + k_{22}P_{24} \\ k_{31}P_{11} + k_{32}P_{21} & k_{31}P_{12} + k_{32}P_{22} & k_{31}P_{13} + k_{32}P_{23} & k_{31}P_{14} + k_{32}P_{24} \\ k_{41}P_{11} + k_{42}P_{21} & k_{41}P_{12} + k_{42}P_{22} & k_{41}P_{13} + k_{42}P_{23} & k_{41}P_{14} + k_{42}P_{24} \end{bmatrix}. \quad (36)$$

Step 9. Calculate the rotor speed from (27); then set $n = n + 1$ and go back to Step 2.

3. Design of Sensorless Speed Controller IP

Figure 3 illustrates the internal architecture of the proposed FPGA implementation of a sensorless speed control IP for PMSM drive system. The internal circuit comprises a Nios II embedded processor IP and an application IP. The Nios II processor is depicted to generate the speed command, collect the response data, and communicate with external device. All programs in Nios II processor are coded in the C language. The application IP includes mainly a circuit of an EKF-based rotor FA and rotor speed estimation, a speed PI controller, a circuit for current controllers and coordinate transformation (CCCT), a SVPWM circuit, and an ADC interface circuit. The sampling frequencies of the speed control loop and current control loop are designed with 2 kHz and 16 kHz, respectively. The frequency divider generates 50 Mhz (Clk) and 12.5 Mhz (Clk-step) clock to supply all circuits in Figure 3. The internal circuit of CCCT performs

the function of two PI controllers, table look-up for sin/cos function, and the coordinate transformation for Clark, Park, inverse Park, and modified inverse Clarke. The detailed circuit design of CCCT, speed PI controller, ADC interface, and SVPWM generation in Figure 3 refers to [8]. Herein, only the digital hardware circuit design of the EKF-based rotor FA and rotor speed estimation is described. To reduce the FPGA resource usage, a finite-state machine (FSM) is employed to model the EKF-based algorithm whose design procedure and mathematical formulation are presented in the previous section. The design result is shown in Figure 4 which uses one adder, one multiplier, one divider, two sin/cos look-up table, shifters, registers, and so forth and manipulates 131 steps machine to carry out the overall computation. The data type adopts 16-bit length with Q15 format and 2's complement operation. The multiplier, adder, and divider apply Altera LPM (Library Parameterized Modules) standard. In Figure 4, the steps $s_0 \sim s_{12}$ execute the calculation of Jacobian matrix and predict temporary state variables; $s_{13} \sim s_{68}$ perform the computation of temporary covariance matrix; steps $s_{69} \sim s_{70}$ are for state error calculation; steps $s_{71} \sim s_{120}$ update of

FIGURE 8: PMSM running at low speed (200 rpm) and inverse speed (± 200 rpm) condition. (a) Speed step response, (b) current response, and (c) actual rotor FA and estimated rotor FA.

FIGURE 9: PMSM running at wide-range high speed (2000 rpm) and inverse speed (± 1000 rpm) condition. (a) Speed step response, (b) current response, AND (c) actual rotor FA and estimated rotor FA.

the present covariance matrix P_n and calculate the Kalman gain K_n ; $s_{121} \sim s_{130}$ describe the present state tuning as well as executing the computation of rotor FA and rotor speed. According to (30), the computation of temporary covariance matrix $P_{n|n-1}$ in Figure 4 is described in Figure 5. Further, according to (18), (33)-(34), and (36), the update of the present covariance matrix P_n and calculation the Kalman gain K_n in Figure 4 are illustrated in Figure 6. The operation of each step in Figure 4 is 80 ns (12.5 MHz) in FPGA; therefore a total of 131 steps need only 10.48 μ s operation times. The FPGA (Altera) resource usages of EKF-based rotor FA and rotor speed estimation in Figure 4 is 4,158 LEs (logic elements) and 2,890 RAM bits. Additionally, the FPGA

resource usages of the circuits of CCCT, speed PI controller, ADC interface, and SVPWM in Figure 3 are 864 LEs and 24,576 RAM, 2,043 LEs and 0 RAM bits, 136 LEs and 0 RAM bits, and 1,221 LEs and 0 RAM bits, respectively.

4. Simulink/ModelSim Cosimulation and Its Simulation Results

The Simulink/ModelSim cosimulation architecture for sensorless PMSM speed control system is shown in Figure 7. The SimPowerSystem-Blockset in the Simulink executes the PMSM and the IGBT-based inverter. The EDA simulator link for ModelSim executes the cosimulation using VHDL code

In simulation, the designed PMSM parameters used in Figure 7 are as follows: pole pairs are 4, stator phase resistance is 1.3Ω , stator inductance is 6.3 mH , inertia is $J = 0.000108 \text{ kg}\cdot\text{m}^2$, and friction factor is $F = 0.0013 \text{ N}\cdot\text{m}\cdot\text{s}$. PI gains in speed control loop are designed with $K_p = 0.2442$ and $K_i = 0.00458$. To evaluate the proposed controller performance, PMSM running at low speed (200 rpm) and inverse speed (from 200 rpm to -200 rpm or vice versa) condition is firstly considered, and its simulation results regarding as the speed step response, the current response as well as the actual rotor flux angle (FA) and the estimated rotor flux angle are shown in Figure 8. It shows that the motor speed gives a good dynamic response performance with a

FIGURE 11: (a) Step speed response of actual rotor speed and estimated rotor speed and (b) actual rotor FA and estimated rotor FA.

FIGURE 12: (a) Step speed response of actual rotor speed and estimated rotor speed and (b) current response.

little overshoot and 25 ms rising time. It also shows that the estimated rotor speed can track actual rotor speed well. Figure 8(b) shows a successful vector control because i_d is controlled to zero. Figure 8(c) shows that the estimated rotor FA can fast track the actual rotor FA even in the inverse speed command condition. Further, another simulation case while the PMSM running at wide-range high speed (from 0 rpm $\rightarrow$ 1000 rpm $\rightarrow$ 2000 rpm $\rightarrow$ 1000 rpm $\rightarrow$ -1000 rpm $\rightarrow$ -2000 rpm) and inverse speed (from 1000 rpm to -1000 rpm) condition is tested, and its simulation results are shown in Figure 9. It presents that not only at speed tracking but also at rotor FA tracking, the estimated value adopted by EKF method can give a good follow to the real rotor value. However, except in the initial condition, the actual rotor speed gives a fast step response by 18 ms rising time, near 0 mm steady-state value, and maximum 17.5% overshoot.

5. Experimental System and Results

After confirming the correctness of the proposed EKF-based sensorless control IP by simulation, the VHDL codes are directly applied to the experimental FPGA-based sensorless PMSM drive system. The block diagram and real experimental system are depicted in Figure 10. 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 $r_s = 0.63 \Omega$, $L = 2.77$ mH, and 4 pole pairs. The input voltage, continuous current, rating torque, rating speed, and continuous power of the PMSM are 220 V, 12 A, 2.3 N-m, 3000 rpm, and 750 W, respectively. 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 multicore 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, 1 Mbyte SRAM, 16 M byte SDRAM, and 4 Gbytes memory outside of the chip. In implementation, except the VHDL codes of CCCT, EKF, and PI controller, the VHDL code of ADC interface circuit should be added and integrated and then downloaded into FPGA. The PI gains of speed control loop in experiment are designed with $K_p = 0.512$ and $K_i = 0.021$.

In the experimental case, PMSM running at low speed (200 rpm) and inverse speed (from 200 rpm to -200 rpm or vice versa) condition are firstly considered, and its experiment results regarding the speed step response as well as the actual rotor FA and the estimated rotor FA are shown in Figure 11. Figure 11(a) shows that the motor speed gives a good dynamic response performance with a little oscillation and overshoot and it has 15 ms rising time. Figure 11(b) shows that the estimated FA can fast track the actual rotor FA even in the inverse speed command condition. Further, another experimental case while the PMSM running at medium speed (from 0 rpm $\rightarrow$ 300 rpm $\rightarrow$ 600 rpm $\rightarrow$ 900 rpm $\rightarrow$ 600 rpm $\rightarrow$ 900 rpm) condition is evaluated, and its experiment results are shown in Figure 12. The Figure 12(a) presents a fast speed tracking with 15 ms rising time and no occurrence of overshoot condition. Figure 12(b) shows a successful vector control due to the i_d being controlled to zero. Therefore, from the simulation results in Figures 8 and 9 and the experimental results in Figures 11 and 12, it is demonstrated that the EKF-based estimation algorithm used in sensorless PMSM drive and the behavior description by using VHDL are effective and correct.

6. Conclusions

This study has presented a sensorless PMSM drive based on EKF and successfully demonstrated its performance through cosimulation by using Simulink/ModelSim and implementation by using FPGA. In realization aspect, the VHDL is used to describe the behavior of EKF algorithm, and FSM is used to reduce the FPGA resource usage; therefore, it only needs 4,158 LEs and 2,890 RAM bits. In computational power aspect, the operation time to complete the computation of EKF algorithm is only $10.48\ \mu\text{s}$, which is less than the $62.5\ \mu\text{s}$ (16 KHz) sampling time in current control loop. In simulated and experimental results, it shows that the use of EKF in sensorless PMSM drive can accurately estimate the rotor FA and rotor speed, and it can give a good step response performance in case of low speed control, inverse speed control, and high speed control as well.

Acknowledgment

This work was supported by the National Science Council of China under Grant no. NSC 101-2221-E-218-039.

References

- [1] V. C. Ilioudis and N. I. Margaris, "PMSM sensorless speed estimation based on sliding mode observers," in *Proceedings of the 39th IEEE Annual Power Electronics Specialists Conference (PESC '08)*, pp. 2838–2843, June 2008.
- [2] S. Chi, Z. Zhang, and L. Xu, "Sliding-mode sensorless control of direct-drive PM synchronous motors for washing machine applications," *IEEE Transactions on Industry Applications*, vol. 45, no. 2, pp. 582–590, 2009.
- [3] V. Delli Colli, R. di Stefano, and F. Marignetti, "A system-on-chip sensorless control for a permanent-magnet synchronous motor," *IEEE Transactions on Industrial Electronics*, vol. 57, no. 11, pp. 3822–3829, 2010.
- [4] S. Bolognani, R. Oboe, and M. Zigliotto, "Sensorless full-digital pmsm drive with ekf estimation of speed and rotor position," *IEEE Transactions on Industrial Electronics*, vol. 46, no. 1, pp. 184–191, 1999.
- [5] M. C. Huang, A. J. Moses, and F. Anayi, "The comparison of sensorless estimation techniques for PMSM between extended Kalman filter and flux-linkage observer," in *Proceedings of the 21st Annual IEEE Applied Power Electronics Conference and Exposition (APEC '06)*, pp. 654–659, March 2006.
- [6] W. Wang, M. Zhang, and Q. Wu, "Application of reduced-order extended kalman filter in permanent magnet synchronous motor sensorless regulating system," in *Proceedings of the International Conference on Digital Manufacturing and Automation (ICDMA '10)*, pp. 271–274, December 2010.
- [7] J.-S. Jang, B.-G. Park, T.-S. Kim, D. M. Lee, and D.-S. Hyun, "Parallel reduced-order extended Kalman Filter for PMSM sensorless drives," in *Proceedings of the 34th Annual Conference of the IEEE Industrial Electronics Society (IECON '08)*, pp. 1326–1331, November 2008.
- [8] Y.-S. Kung and M.-H. Tsai, "FPGA-based speed control IC for PMSM drive with adaptive fuzzy control," *IEEE Transactions on Power Electronics*, vol. 22, no. 6, pp. 2476–2486, 2007.
- [9] L. Idkhajine and E. Monmasson, "Design methodology for complex FPGA-based controllers—application to an EKF sensorless AC drive," in *Proceedings of the 19th International Conference on Electrical Machines (ICEM '10)*, pp. 1–6, September 2010.
- [10] H.-H. Chou, Y.-S. Kung, N. Vu Quynh, and S. Cheng, "Optimized FPGA design, verification and implementation of a neuro-fuzzy controller for PMSM drives," *Mathematics and Computers in Simulation*, vol. 90, pp. 28–44, 2013.
- [11] Y. S. Kung and T. H. Nguyen, "Simulink/Modelsim Co-simulation of EKF-based Sensorless PMSM drives," in *Proceedings of IEEE International on Power Electronics and Drive Systems (PEDS '13)*, pp. 709–713, 2013.
- [12] 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 '10)*, pp. 223–227, August 2010.
- [13] The Mathworks, "Matlab/Simulink Users Guide, Application Program Interface Guide," 2004.
- [14] Modeltech, "ModelSim Reference Manual," 2004.