366 Journal of Electrical Engineering & Technology Vol. 7, No. 3, pp. 366~375, 2012
http://dx.doi.org/10.5370/JEET.2012.7.3.366

Design and Temporal Analysis of Hardware-in-the-loop Simulation for Testing Motor Control Unit

Chinchul Choi*, Kangseok Lee* and Wootaik Lee†

Abstract – This paper describes a design and temporal analysis of a hardware-in-the-loop (HIL) simulation environment for testing a motor control unit (MCU). The design concepts and main characteristics including unavoidable time delays of each component module are described. From temporal analysis results according to the module integration method, an appropriate solution is proposed to fix and minimize time delays. In order to verify the effectiveness of the proposed solution, the HIL test results are compared with the results of experiments and an offline simulation.

Keywords: Hardware-in-the-loop (HIL) Simulation, Motor, Control, Model, Time delay

1. Introduction

With increasing demand for environmentally friendlier and higher fuel economy vehicles, automotive companies are focusing on electric vehicles, hybrid electric vehicles, and fuel-cell electric vehicles. In developing these vehicles, key challenges include achieving high efficiency, ruggedness, small size, and low cost in the power converter and electrical machines, as well as in the associated electronics [1, 2]. Great emphasis has recently been placed on the efficiency and optimal utilization of permanent magnet motors for a variety of automotive applications, such as hybrid propulsion, electric power steering, braking, engine cooling fans, fuel/water pumps, air-conditioning compressors, and so on [3, 4].

Such systems require motor control units (MCUs), which incorporate more complex control schemes such as high performance sensorless operation, fault tolerance, advanced diagnostics, and self-tuning capabilities [5] .The increasing complexity of the MCU has made efficient design and test environments indispensible. Engineers have found quantitative simulations to be a vital tool for designing and testing a variety of control functions. A hardware-in-the-loop (HIL) simulation is a kind of real-time simulation, and has been used to test controllers thoroughly, efficiently, and safely [6, 7].

There are various approaches for HIL simulation of electric drives. These approaches can be mainly categorized according to two points of view, interface and model level [8, 9].

According to the interface level between the controller and the HIL simulator, there are three approaches:

- Signal level [10, 11]
- Electrical level [12, 13]
- Mechanical level

The signal level is very flexible, because it provides full access to modify the real-time model. However, the other levels are dependent on the power level and entail the same risks when testing using actual plants.

The two common approaches for modeling are [8, 9]:

- Oversampling model
- Average-value (Mean-value) model.

The oversampling model requires computation that is more than 10 times faster than the PWM period [9]. This allows for the simulation of high frequency ripples on the current waveform, which are caused by the PWM switching effects. Because of the higher execution rate, the oversampling model is computationally more intensive. A FPGA based model implementation is an appropriate solution to satisfy the requirement of fast execution time [8]. The FPGA requires conversion of a floating point model (generally used in offline simulation) into a fixed point model. This converting process makes it difficult to modify and add features of the model, e.g., the model implementation for fault simulation is very complicated.

On the average-value model, the average values of the PWM signals are sampled once per PWM period. The main advantage of the average value approach compared to the oversampling approach is reduced computational load. Therefore, the average value model is more useful for fast analysis of system level behaviors than for obtaining detailed switching effects. Also, this approach allows the use of the floating point model, which is used in offline simulations. Additionally, a low-cost configuration can be achieved [10].

On the other hand, the average value approach has some delays depending on the integration method [9, 10]. A temporal analysis and design process are required to determine and minimize these delays.

This paper describes the design and temporal analysis of

† Corresponding Author: Dept. of Control and Instrumentation Engineering, Changwon National University, Korea. (wootaik@changwon.ac.kr)
* Graduate School of Control and Instrumentation Engineering, Changwon National University, Korea. (startchrap@changwon.ac.kr, ks1549@gmail.com)
Received: April 11, 2011; Accepted: April 6, 2012

the signal and average value level HIL simulation environment for testing the MCU. The main characteristics of each component module are described. From the temporal analysis results according to the module integration method, an appropriate solution is then proposed to fix or minimize the time delays. In order to verify the effectiveness of the proposed solution, the HIL simulation results are compared with experiments and offline simulation results.

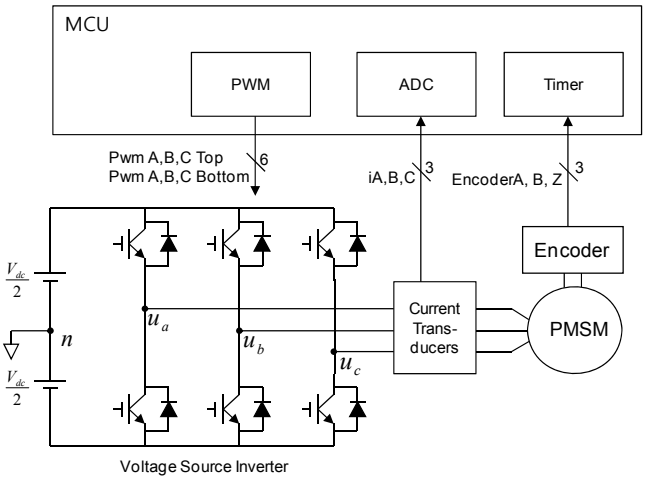

Fig. 1. A general PMSM control system.

2. System Configuration

Fig. 1 shows a general PMSM control system that consists of a MCU, VSI (Voltage Source Inverter), PMSM, and sensors. Current transducers and a position sensor such as an incremental encoder are indispensible for fundamental PMSM vector control. The MCU samples the sensor signals and then the calculated results are outputted as low power PWM pulse signals to operate the VSI. According to the PWM signals of the MCU, the VSI converts the DC power source to 3-phase AC power for the PMSM.

Fig. 2 shows the configuration of the signal level HIL simulator, which replaces the actual VSI and PMSM hardware as real-time models for testing functions of the MCU. The HIL simulator consists of I/O interface modules and a real-time processor. Detailed features of each component are described in the following subsections.

2.1 Motor Control Unit (MCU)

The modern MCUs run not only the basic vector control algorithm for high performance drives, but also perform additional advanced functions such as fault detection and tolerance, diagnostics, and so on. Digital signal processors (DSPs) are widely used as the main CPU (central

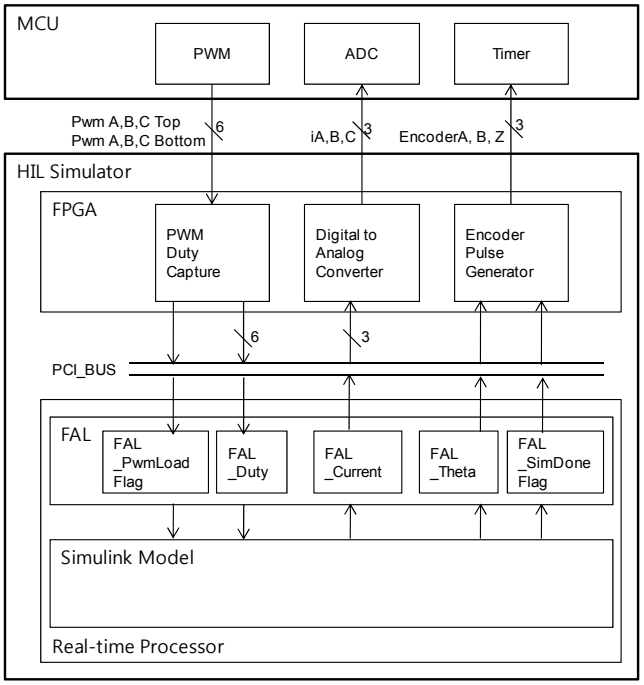

Fig. 2. The HIL simulation configuration.

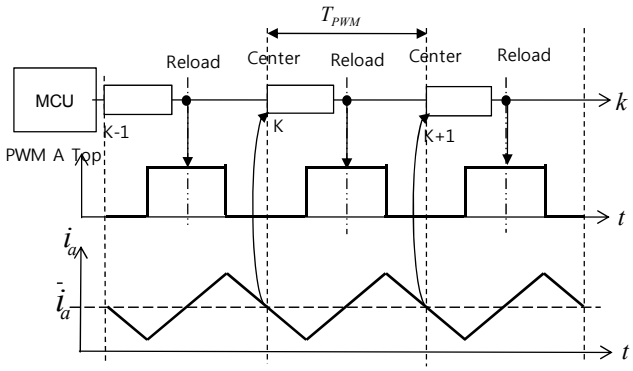

Fig. 3. Typical behaviors of the MCU.

processing unit) of MCUs because of the DSP's specific peripherals for electric drives [14, 15]. Fig. 3 describes the typical time sequence of sensing, computation, and the PWM output. From [14] and [15], we can assume behaviors of the MCU as follows.

The vector control is computed every PWM period, which is typically in a range of 5 kHz to 20 kHz. Sampling of the current is done at the center of the PWM period to eliminate most of the harmonics caused by the PWM and to capture the average value of the current during the sampling time. The position information is also sampled with current at the same time by counting pulse signals of the incremental encoder. Using this sampled information, the MCU calculates the next duty cycle of each PWM signal. The calculated duty cycle is only updated at the reload time.

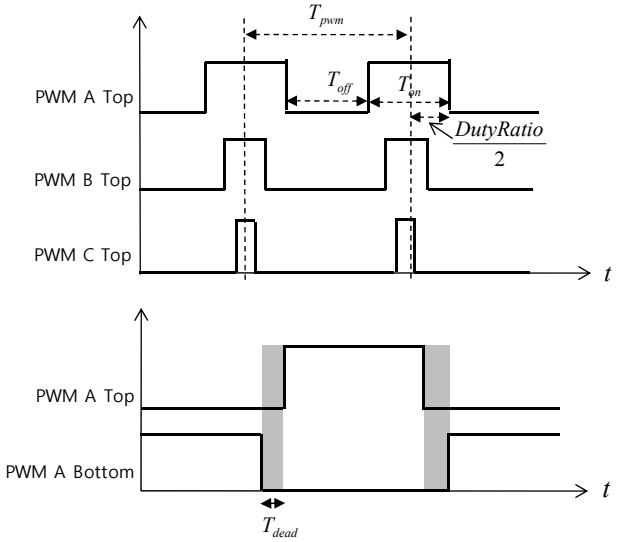

Fig. 4. PWM signals for the PMSM control

2.2 PWM duty capture

The duty cycle is computed by taking the capturing relative time at each rising and falling edge of the PWM signal. Fig. 4 shows the center-aligned PWM signals fed to the VSI for the PMSM drives. Two PWM signals for the top and bottom switch of each phase have a complementary relationship and include dead-time to prevent the two switches being turned on together.

If the PWM period and dead-time are known, the duty cycle can be computed by only using three PWM signals for the top switches. For other cases, all PWM signals are required for computing the dead-time.

$$\begin{cases} DutyRatio_x = T_{on\_x} / T_{pwm} \\ DutyRatio_x = 1 - \left(T_{off\_x} / T_{pwm} \right) \end{cases} \quad (1)$$

$$T_{pwm} = T_{on\_x} + T_{off\_x} \quad (2)$$

$$T_{dead} = \left| T_{on\_x\_top} - T_{off\_x\_bottom} \right| \quad (3)$$

where x is a, b, or c. Even if design details of the MCU are unknown (e.g. when black box test), the PWM period and dead-time can be obtained with little effort to check the PWM output signals. Therefore, we propose that the use of three PWM signals of the top switches is more appropriate. Under these assumptions, the duty cycle can also be obtained within a half PWM period by use of the simple formula given as Eq. (4).

$$\frac{DutyRatio_x}{2} = T_{on\_x} - \frac{(T_{pwm} - T_{off\_x})}{2} \quad (4)$$

2.3 Encoder pulse generator

The encoder pulse trains are generated by use of the rotor position information obtained from the real-time processor, as shown in Fig. 5. For accurate generation of the encoder pulses, high resolution position information is required. Fig. 6 shows an extrapolation and interpolation method for creating the position information with high resolution. The dotted line indicates the extrapolation method using the M-th position and speed information. The solid line indicates the linear interpolation method. The information obtained from M-1 to M is linearly interpolated in the M to M+1 region. θ_n^{ext} is the extrapolated position information and θ_n^{int} is the interpolated information.

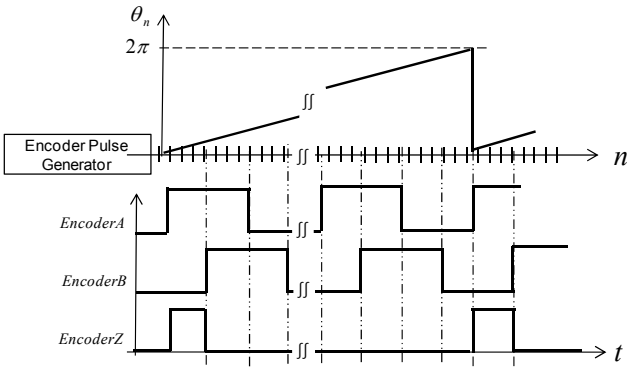

Fig. 5. Encoder pulses generation.

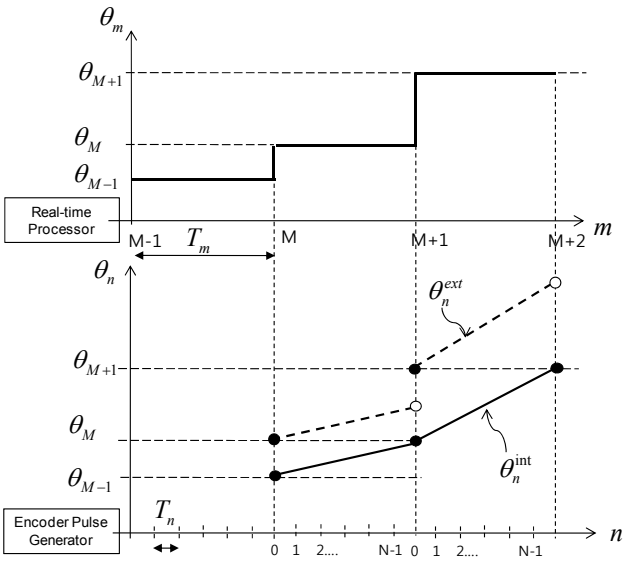

Fig. 6. Generating the high resolution position information.

$$\theta_n^{ext} = \theta_m + \left(\frac{\theta_m - \theta_{m-1}}{N} \right) \cdot n \quad (5)$$

$$\theta_n^{\text{int}} = \theta_{m-1} + \left(\frac{\theta_m - \theta_{m-1}}{N} \right) \cdot n \quad (6)$$

$$\begin{cases} \theta_0^{ext} = \theta_m & \theta_N^{ext} = 2\theta_m - \theta_{m-1} \\ \theta_0^{\text{int}} = \theta_{m-1} & \theta_N^{\text{int}} = \theta_m \end{cases} \quad (7)$$

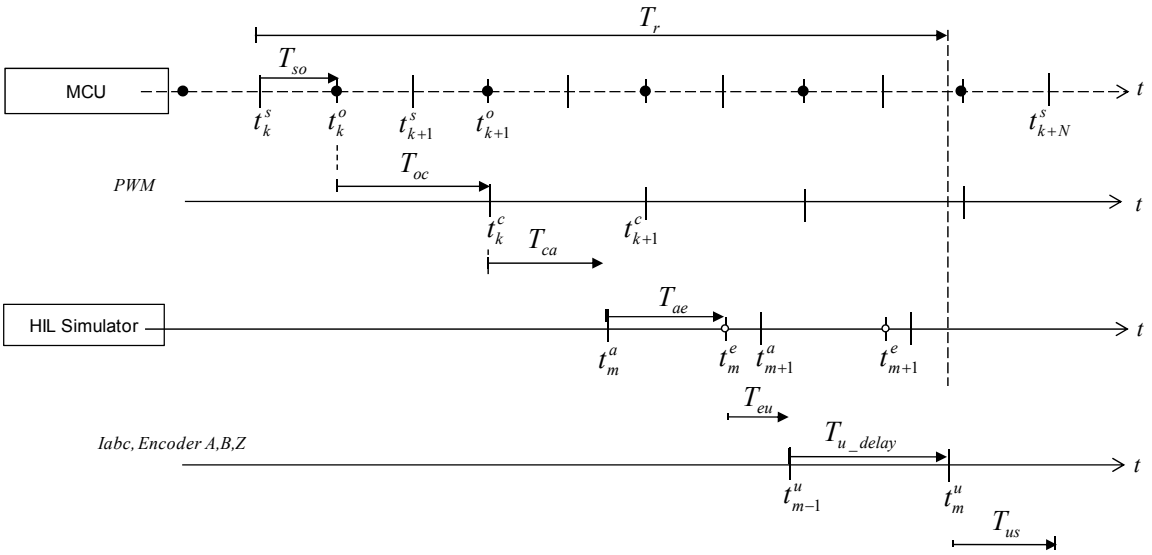

Fig. 7. Time sequence diagram for integration of the MCU and HIL simulator

where N is the integer result of T_m over T_n, and n is the circularly counting value from 0 to N-1.

The extrapolation method produces a discontinuous point at a transient state (when $\theta_{m+1} - \theta_m \neq \theta_m - \theta_{m-1}$ or $\theta_{m+1} \neq 2\theta_m - \theta_{m-1}$). The interpolation method is free to discontinuous but causes a 1-step time delay. The generated encoder pulse with discontinuous causes drifts of the position information between the real-time processor and the MCU [16]. If these drifts are accumulated in excess of a half cycle of the rotated position, the synchronous drive can break. Although a 1-step delay can be endured, interpolation is recommended for stable operation.

2.4 Digital to analog converter

This module outputs the analog signals, which are converted from the current information of the real-time processor in order to be fitted to the current transducer's characteristics. Although the current information can be used as soon as the computation of the real-time processor is completed, the current information must be synchronized with the position information.

2.5 FPGA abstraction layer (FAL)

This software layer converts fixed-point signals of the FPGA to floating-point signals of the real-time model, and vice versa. By use of this layer, the offline simulation model can be directly used as the real-time model.

2.6 Simulink model

The models dealt with in this paper are implemented under the MATLAB/Simulink environment, which is basically a digital simulation program for nonlinear dynamic systems. Simulink is widely used as one of the de facto standard tools for efficient system level modeling of complex systems including motor drives [17, 18]. These offline models can be used as the real-time model for the proposed HIL simulation. This paper uses a well-known linear PMSM model which is implemented by circuit based modeling toolboxes (SimPowerSystems) in MATLAB /Simulink [18]. The real-time processor (National Instruments (NI) PXI-8106 with 2.16GHz Intel dual core) requires 45μ second in the worst case in order to execute the model. This worst case execution time (WCET) occupies 72 % of a PWM period, 62.5μ second in this paper. According to the test scope and purpose, more complex models included harmonic model or thermal model can be used. However, it is important that the WCET do not exceed a PWM period in the presented HIL system.

3. Temporal Design and Analysis

Based on the design results of each module this section deals with the interface and integration of modules for the HIL simulation. Fig. 7 shows the general time sequence diagram for integration of the MCU and the HIL simulator. We define the event t as an occurrence at a point in time and the duration, T as the time interval between two events.

The MCU samples at t_k^s and then the calculated results are output as the PWM signals at the reload time, t_k^o. The PWM duty capture module computes the duty cycle during the effective region, T_{oc} for the outputted PWM signals at t_k^o. The captured duty cycle is then transferred to the real-time processor at t_k^c. At t_k^c, the real-time processor is activated, and model execution is completed at t_m^e. As soon as model execution is completed, the real-time

processor transfers the executed results to the encoder pulse generator and the DAC module. These two modules update l-step delayed results. Additional delay (T_{eu}) can be inserted intentionally. The total response time from sensing of the MCU to updating I/O interface of the HIL simulator is calculated as Eqs. (8).

$$\begin{aligned} T_r &= t_m^u - t_k^s \\ &= T_{so} + T_{oc} + T_{ca} + T_{ae} + T_{eu} + T_{u\_delay} \end{aligned} \tag{8}$$

Some kinds of duration are constant with use of the suggested module design. T_{so} constantly consumes time equal to a half PWM period, as assumed earlier for the MCU operation. T_{oc} is determined by selecting the duty capture method regardless of whether a full or half PWM period is used. T_{u_delay} is the PWM period, which is the same as the sampling period.

$$\begin{cases} T_{so} = 0.5T_{pwm} \\ T_{oc} = T_{pwm} \ or \ 0.5T_{pwm} \\ T_{u\_delay} = T_{pwm} \end{cases} \tag{9}$$

In the following subsections, the total response time is calculated according to the system configuration method.

3.1 Asynchronous configuration

Each module asynchronously operates with its own fixed-time step. The propagation delay T_{ca} cannot exceed T_{pwm} if the execution time T_{ae} of the real-time processor is kept within T_{pwm} . There is no reason to select T_{eu} such that the sum of T_{eu} and T_{ae} exceeds T_{pwm} .

According to the range of these durations, the minimum and maximum response times are calculated as Eqs. (11) and (12), respectively.

$$\begin{cases} 0 \le T_{ca} < T_{pwm} \\ 0 < T_{ae} < T_{pwm} \\ 0 \le T_{eu} < \left(T_{pwm} - T_{ae}\right) \end{cases} \tag{10}$$

$$\begin{aligned} T_{r\_\min} &= T_{so} + T_{oc} + T_{ca\_\min} + T_{ae\_\min} + T_{eu\_\min} + T_{u\_delay} \\ &= 0.5T_{pwm} + T_{pwm} + 0 + 0 + 0 + T_{pwm} \\ &= 2.5T_{pwm} \end{aligned} \tag{11}$$

$$\begin{aligned} T_{r\_\max} &= T_{so} + T_{oc} + T_{ca\_\max} + T_{ae\_\max} + T_{eu\_\max} + T_{u\_delay} \\ &= 0.5T_{pwm} + T_{pwm} + T_{pwm} + T_{ae} + \left(T_{pwm} - T_{ae}\right) + T_{pwm} \\ &= 4.5T_{pwm} \end{aligned} \tag{12}$$

According to T_{ca} , T_{ae} , and T_{eu} , the total response time varies in a range from 2 to 4.5 steps. In this configuration, the k-th information can be lost when the k+1-th information is used in the real-time processor before the k-th information [19-21], because some jitter of each event and duration is unavoidable.

3.2 Synchronous configuration

After the information of the precedent phase is valid, the following phase is triggered at fixed timing for synchronization. Two approaches are possible for the synchronization. First one uses a hardware connection for the synchronization in addition to six PWM signals. And the second calculates the synchronization timing from the PWM signals. The second method, which is used in this paper, does not have an additional wire but requires a fast computation and sampling rate in order to avoid jitters The PWM duty capture module, which is implemented by FPGA, calculates the synchronization timing and transfers this event signal to the real-time processor. As soon as the PWM duty cycle is captured and valid, the real-time processor is triggered and activated for execution (i.e. $t_k^c = t_m^a$). Additional delay (T_{eu}) is added in order to fix the update timing at the sampling or output timing of the MCU (i.e. $T_{eu} = \left(T_{pwm} - T_{ae}\right)$). Under these conditions, the total response time is calculated as Eq. (12).

$$\begin{aligned} T_r &= T_{so} + T_{oc} + T_{ae} + (T_{pwm} - T_{au}) + T_{u\_delay} \\ &= 0.5T_{pwm} + T_{pwm} + T_{pwm} + T_{pwm} \\ &= 3.5T_{pwm} \end{aligned} \tag{13}$$

If T_{oc} is $0.5T_{pwm}$, T_r becomes $3T_{pwm}$.

Although the synchronous configuration may occasionally have more delay than the asynchronous configuration, the system has a fixed-step delay, which makes the system more deterministic. If it is assumed that T_{ae} is larger than $0.5T_{pwm}$ and T_{oc} consumes $0.5T_{pwm}$, the synchronous configuration has the same delay as the asynchronous configuration with 3-step delay. In addition, all modules are operated in consecutive order without sample rejection in the synchronous configuration. Fig. 8 shows the time sequence for the proposed HIL simulation configuration.

4. Test Results

Several tests are carried out to validate the HIL simulation environment by comparison with actual experimental and offline simulation results. To perform the offline simulation, a model of the drive system including the MCU is assembled in MATLAB/Simulink. The MCU model takes into account physical implementation of the actual MCU, such as the various sampling rates and fixed

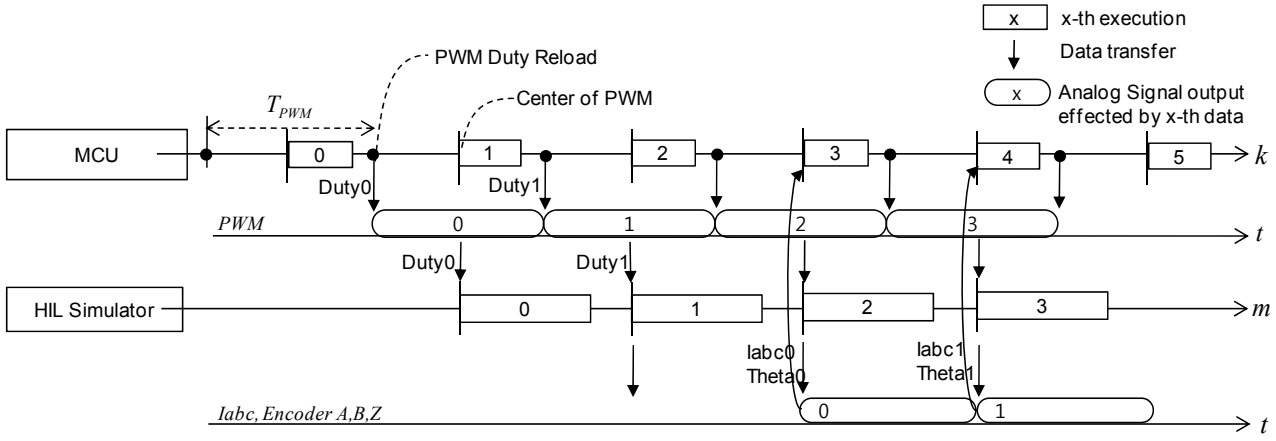

Fig. 8. Time sequence diagram for the proposed HIL simulation configuration.

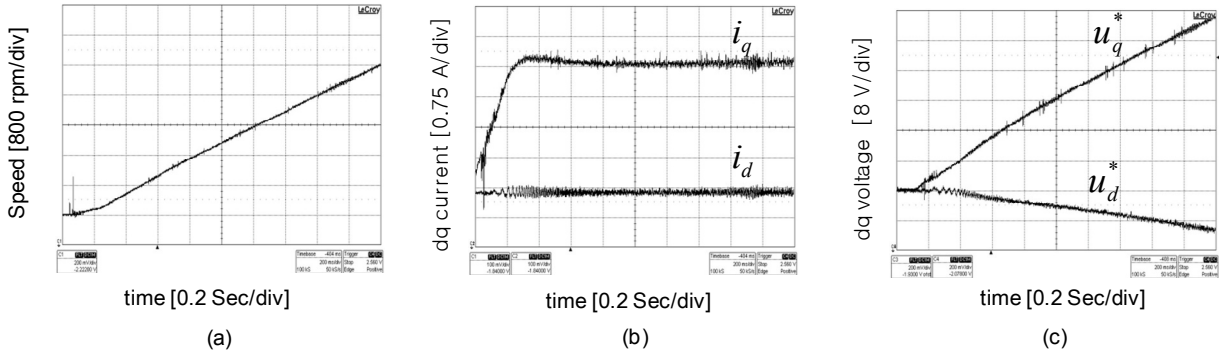

Fig. 9. Experiment results: (a) motor speed [800rpm/div]; (b) measured dq current [0.75A/div]; (c) dq voltage references [8V/div]

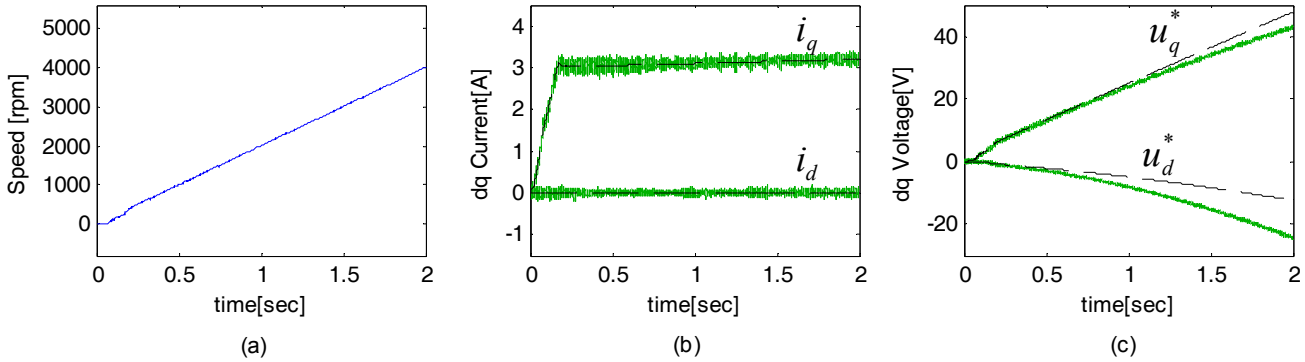

Fig. 10. Offline(dashed line) and HIL test results(Solid line): (a) motor speed; (b) measured dq current; (c) dq voltage references

point arithmetic.

On the other hand, the real MCU is interfaced to the HIL simulator for the HIL simulation. The vector control algorithm is implemented in a digital signal processor (DSP) 56F8367 manufactured by Freescale Semiconductor Incorporation. The specifications and parameters of the digital controller and 0.5kW PMSM are described in Table 1.

The first test scenario is that the speed reference increases constantly from 0 to 4000 rpm for two seconds. Fig. 9 shows the experimental results and Fig. 10 shows the offline simulation results and HIL simulation results. The experiment and offline simulation yield similar results but the HIL simulation shows voltage deviations with an increase of speed.

The second test scenario for both the offline and HIL simulation consists of stepped changes of the speed reference, as 1000, 4000, and 8000 rpm. In order to regulate the motor speed according to the step reference,

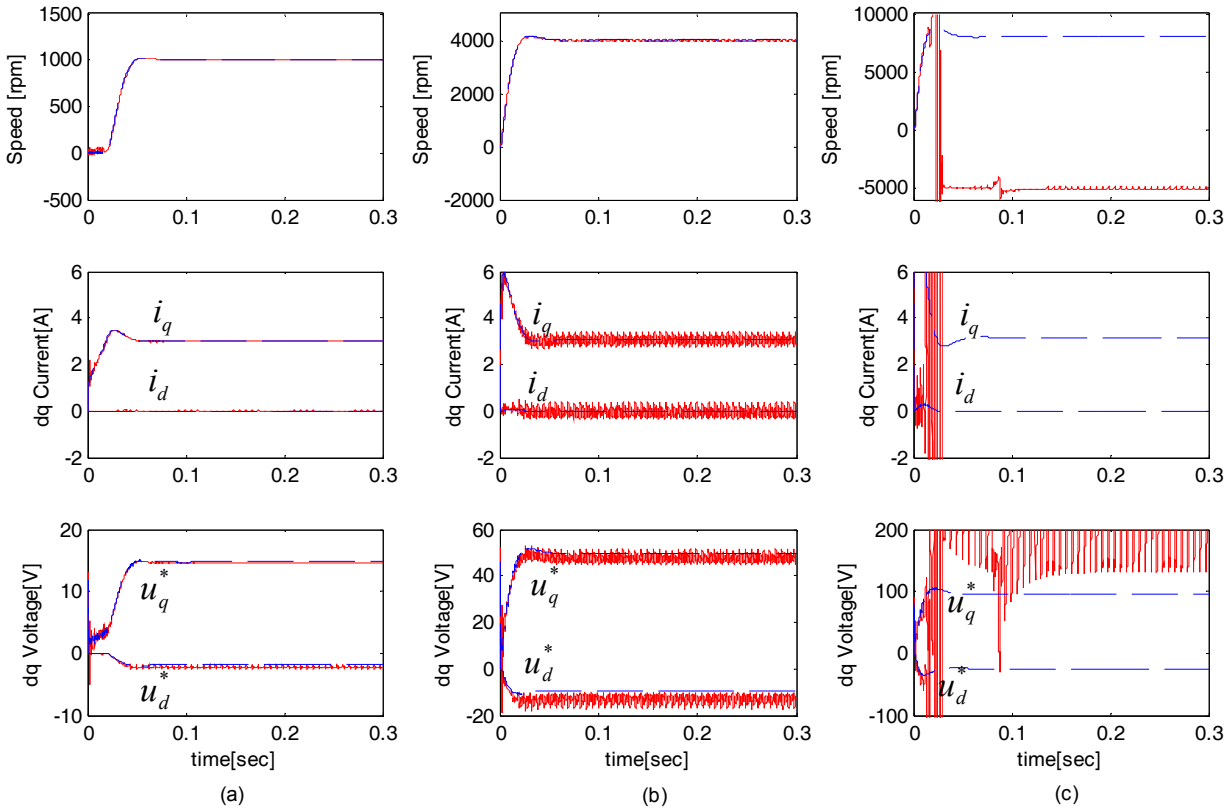

Fig. 11. Step response comparison between offline simulation (dashed line) and HIL simulation (solid line) with an asynchronous configuration (a) 1000rpm, (b) 4000rpm, (c) 8000 rpm

Table 1. Rating and parameters of PMSM

Descriptions	Value	Unit
DC Bus Voltage	180	V
Current (peak)	3	A
Speed	8000	Rpm
Load torque	0.5	Nm
Stator resistance	0.98	Ω
Stator inductance	2.3	mH
Rotor flux linkage	6.55	mWb
Number of pole	4	EA

the speed PI controller outputs the reference of the q-axis current. The reference of the d-axis current is always zero in these cases. And then each for d and q-axis current PI controller outputs the reference voltage of each axis to regulate the motor current.

Fig. 11 shows the results of a comparison between the offline simulation and HIL simulation using the asynchronous configuration. At 1000 rpm, two simulation results have similar values of speed, current, and voltage. But the HIL simulation has ripples due to sample rejection. These sample rejections are caused by the tiny difference in the sampling time between the MCU and HIL simulator. The low cost MCU shows limited resolution of the timer module and experiences difficulty to produce precise sampling time. With increasing motor speed, the negative influences become larger and finally the speed regulator loses control at a speed of 8000 rpm, as shown in Figs. 11. (b) and (c).

Fig. 12 shows results of the HIL simulation with the synchronous configuration. The results show that the motor speed is controlled effectively up to 8000 rpm without ripple. The figure also shows that the voltage references of the HIL simulation deviate from the offline model with an increase of the motor speed. The time delay is represented in the time domain as $x(t-T_D)$ and can be transformed in the frequency domain as $e^{-sT_D}X(s)$ or $e^{-j\omega T_D}X(j\omega)$. Therefore, there is relationship between the digital controller side voltage, u^*_{dq} and HIL simulator side voltage, u_{dq} as $U_{dq}=e^{-j\omega T_D}\cdot U^*_{dq}$. Because the amount of time delay, T_D of the HILS is different from the offline simulation or experiments, there exists distortions of the HIL simulation results compared with the offline simulation or actual experiments. Due to the speed and current controller, the values of the MCU and HIL simulator are the same although there exists a time delay between the MCU and HIL simulator. The time delay effects results in voltage deviation compared with the offline simulation.

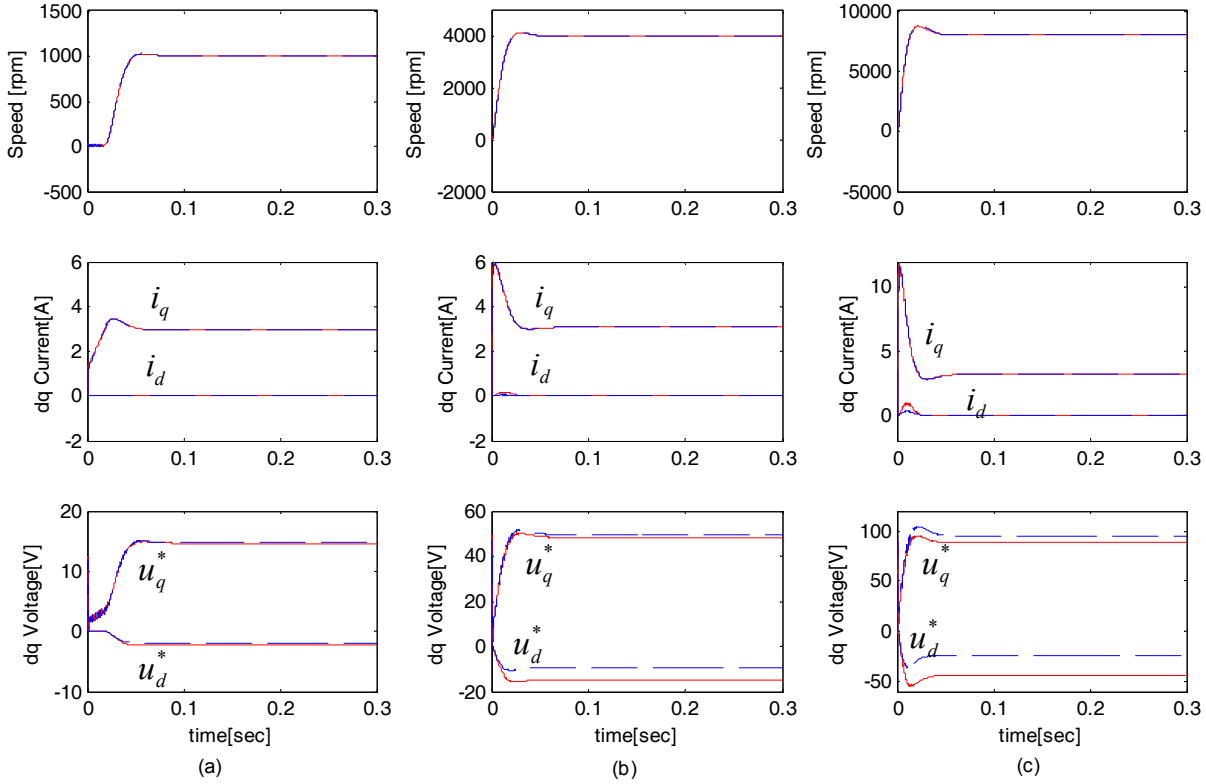

Fig. 12. Comparison between offline and HIL simulation with the synchronous configuration (solid line : HIL simulation, dashed line : offline simulation): (a) 1000rpm; (b) 4000rpm; (c) 8000 rpm

5. Conclusions

The system configuration of the signal and average value level HIL simulation environment is introduced. Appropriate design approaches of the main components are also proposed. From the results of a temporal analysis, the synchronized system configuration minimizes the time delay equivalent to as the amount of 3-step. By comparing the experimental and offline simulation results, the HIL simulation results are validated. The results show that the synchronous configuration is an appropriate solution for obtaining more accurate simulation results than an asynchronous configuration. For the application of high speed drives, a proper compensation method for the time delay effects is needed.

Nomenclature

T_{pwm} : The PWM period
T_{on} : On time of PWM signal
T_{off} : Off time of PWM signal
T_{dead} : Dead time of PWM signal
T_m : Sampling time of the real-time model
T_n : Sampling time of the encoder pulses generator
θ_m : Motor position information of the real-time model
θ_n : Generating motor position information
θ_n^{int} : Interpolated position information of motor
θ_n^{ext} : Extrapolated position information of motor
t_k^s : k-th sensing of MCU
t_k^o : k-th output of MCU
t_k^c : Capture complete of k-th output duty of MCU
t_m^a : m-th acquisition of HILS for t_{c_k}
t_m^e : m-th execution complete of HILS
t_m^u : m-th update of HILS
T_{so} : The required time to output the executed result using sensing data
T_{oc} : The required time for capture the duty ratio
T_{ca} : The propagation delay to invoke the execution of HILS
T_{ae} : The execution time for HILS
T_{eu} : The delay time to update the executed result of HILS
T_{u_delay} : The delay time for interpolation of updated data
T_{us} : The propagation delay to invoke the execution of MCU
T_r : Total response time
i_q : q-axis current
i_{d*} : d-axis current
u_q : reference voltage of q-axis
u_q^* : reference voltage of q-axis
u_d^* : reference voltage of d-axis

Acknowledgements

The authors of this paper were partly supported by the Second Stage of Brain Korea21 Projects.

References

[1] A. Emadi, L. Young Joo, and K. Rajashekara, "Power Electronics and Motor Drives in Electric, Hybrid Electric, and Plug-In Hybrid Electric Vehicles," *IEEE Trans. Industrial Electronics*, vol. 55, pp. 2237-2245, 2008.

[2] J.G. Lee, "A Lookup Table Based Loss Minimizing Control for FCEV Permanent Magnet Synchronous Motors", *Journal of Electrical Engineering & Technology*, vol. 4, No. 2, 2009.

[3] Y.D. Son, "Drive System Design for a Permanent Magnet Motor with Independent Excitation Winding for an Electric Bicycle", *Journal of Electrical Engineering & Technology*, vol. 5, No. 4, 2010.

[4] Y. K. Chin and J. Soulard, "Modeling of Iron Losses in Permanent Magnet Synchronous Motors with Field-Weakening Capability for Electric Vehicles," *International Journal of Automotive Technology*, vol. 4, pp. 87-94, 2003.

[5] R. Champagne, L. A. Dessaint, H. Fortin-Blanchette, and G. Sybille, "Analysis and validation of a real-time AC drive simulator," *IEEE Trans. Power Electronics*, vol. 19, pp. 336-345, 2004.

[6] L. Jianhui, K. R. Pattipati, Q. Liu, and S. Chigusa, "An Integrated Diagnostic Development Process for Automotive Engine Control Systems," *IEEE Trans. Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 37, pp. 1163-1173, 2007.

[7] A. Saleem, R. Issa, and T. Tutunji, "Hardware-In-the-Loop for on-line identification and control of three-phase squirrel cage induction motors," *Simulation Modelling Practice and Theory*, vol. 18, pp. 277-290, 2010.

[8] A. Wagener, T. Schulte, P. Waeltermann, and H. Schuette, "Hardware-in-the-Loop Test Systems for Electric Motors in Advanced Powertrain Applications," *in 2007 SAE World Congress*, pp. 2007-01-0498,2007.

[9] A. Dhaliwal and J. Allen, "Hardware-in-Loop Simulation of Electric Drives -Desciption of a Typical Simulation Platform," *in 2007 SAE World Congress*, pp. 2009-01-2839, 2009.

[10] B. Lu, X. Wu, H. Figueroa, and A. Monti, "A Low-Cost Real-Time Hardware-in-the-Loop Testing Approach of Power Electronics Controls," *IEEE Trans. Industrial Electronics*, vol. 54, pp. 919-931, 2007.

[11] V. R. Dinavahi, M. Reza Iravani, and R. Bonert, "Real-time digital simulation of power electronic apparatus interfaced with digital controllers," *IEEE Trans. Power Delivery*, vol. 16, pp. 775-781, 2001.

[12] S. Ayasun, R. Fischl, S. Vallieu, J. Braun, and D. dIrlI, "Modeling and stability analysis of a simulation-stimulation interface for hardware-in-the-loop applications," *Simulation Modelling Practice and Theory*, vol. 15, pp. 734-746, 2007.

[13] W. Ren, M. Steurer, and T. L. Baldwin, "An Effective Method for Evaluating the Accuracy of Power Hardware-in-the-Loop Simulations," IEEE Trans. Industry Applications, vol. 45, pp. 1484-1490, 2009.

[14] J. Musil, "3-Phase PMSM Vector Control," *Reference Manuaul of freescale semiconductor,* Inc., DRM0632004.

[15] M. Platnic, "Implementation of Vector Control for PMSM Using the TMS302F240 DSP," *Application Report of Texas Instruments*, SPRA4941998.

[16] A. Dhaliwal, S. C. Nagaraj, and S. Ali, "Hardware-in-the-Loop Simulation for Hybrid Electic Vehicles - An Overview, Lessons Learned and Solutions Implemented," *in 2009 SAE World Congress*, pp. 2009-01-0735, 2009.

[17] C. M. Ong, *Dynamic Simulation of Electric Machinary*: Prentice Hall PTR, 1998.

[18] W. Hong, C. Choi, W. Lee, J.-P. Hong, and D. KUM, "Advanced Permanent Magent Motor Drive Modeling For Automotive Application Under MATLAB/Simulink Environment," *International Journal of Automotive Technology*, vol. 10, pp. 489-495, 2009.

[19] N. C. Beaulieu, E. Biglieri, and M. Lai, "Sample rejection and importance sampling in the simulation of multidimensional signalling systems," *IEE Proceedings of Communications, Speech and Vision,* vol. 140, pp. 445-452, 1993.

[20] P. Loskot and N. C. Beaulieu, "Sample rejection for efficient simulation of binary coding schemes over quantized additive white Gaussian noise channels," *IEEE Trans. Communications*, vol. 53, pp. 1145-1154, 2005.

[21] M. Torngren, "Fundamentals of implementing Real-time Control Applications in Distributed Computer Systems," *Journal of Real-time systems*, vol. 14, pp. 219-250, 1998.

Chinchul Choi received the B.S., M.S. degrees in control and instrumentation engineering from Changwon National University, Changwon, Korea, in 2006 and 2008, respectively. He is currently working toward the Ph.D. degree at Changwon National University. His research interests are the control, modeling, and fault diagnosis of electric drive systems.

Kangseok Lee received the B.S., M.S. degrees in control and instrumentation engineering from Changwon National University, Changwon, Korea, in 2008 and 2011, respectively. He is currently working toward the Ph.D. degree at Changwon National University. His research interests are software programming and testing of embedded control systems.

Wootaik Lee received the B.S., M.S. degrees in electrical engineering and Ph.D. degree in automotive engineering from Hanyang University, Seoul, Korea, in 1996, 1998, and 2002, respectively. From 2001 to 2003, he was with Automotive Control and Electronics Research Center, Seoul, Korea, where he was a senior engineer for developing automotive electronic control unit, and designing in-vehicle control network. Since 2003, he has been with the Department of Control and Instrumentation Engineering, Changwon National University, Changwon, Korea, where he is currently an Associate Professor. His research interests include model-based development process for embedded control system, and motor drive for automotive application.