ISSN 1330-3651 (Print), ISSN 1848-6339 (Online)
UDC/UDK 621.313.13.072.1:004.421

A HIGH SPEED CURRENT dq PI CONTROLLER FOR PMSM DRIVE

Mohammad Marufuzzaman, Mamun Bin Ibne Reaz, Labonnah Farzana Rahman, Tae Gyu Chang

Original scientific paper

A Field Programmable Gate Array (FPGA) based solution of current dq PI controller is proposed in this research, which is usually implemented in digital signal processor (DSP) based computer. The main problem in DSP based solution is the execution time, which is usually in microseconds range as well as reaching its physical limits. Therefore, completing the execution within nanoseconds becomes a major challenge to all researchers, which can be done by reducing the clock cycles. Implementing the overall controlling algorithm into FPGA will certainly reduce the execution time dramatically to pledge the steadiness of the motor. The result shows that the proposed FPGA performance requires only 68 ns of execution time for operating frequency of 30 MHz and accuracy of 99,9 %, which is the lowest computational cycle for the era.

Keywords: *current dq, FOC PMSM (Field Oriented Control of Permanent Magnet Synchronous Motor), FPGA (Field Programmable Gate Array), PI controller*

PI regulator dq struje velike brzine za pogon PMSM

Izvorni znanstveni članak

Rješenje PI regulatora struje dq utemeljeno na Field Programmable Gate Array (FPGA) predlaže se u ovom istraživanju, a obično se provodi na računalu s procesorom digitalnog signala (DSP). Glavni problem kod DSP temeljenog rješenja je vrijeme izvršenja, koje je obično u rasponu od mikrosekundi, kao i dostizanju njegovih fizičkih granica. Stoga, dovršavanje izvršenja unutar nanosekundi postaje veliki izazov za sve istraživača, što može biti učinjeno smanjenjem ciklusa sata. Uvođenje ukupnog kontrolnog algoritma u FPGA sigurno će dramatično smanjiti vrijeme izvršenja kao zalog za postojanost motora. Rezultat pokazuje da predložena FPGA izvedba treba samo 68 ns korištenog vremena za operativnu frekvenciju od 30 MHz i točnost od 99,9 %, što je najniži računalni ciklus ovoga doba.

Ključne riječi: *FOC PMSM (vektorski upravljani sinkroni motor s permanentnim magnetom), FPGA (integrirani sklop programilan od korisnika), PI regulator, struja dq*

1 Introduction

In field oriented control of permanent magnet synchronous motor (FOC PMSM) drives, the *dq*-axis current control plays an important role in determining the overall system performance. The proportional and integral (PI) controller is most widely used for controlling dq currents of PMSM. The performance of the system mainly depends on the quality of the applied current control strategy. The implementation of FOC requires current controllers with fast response and high accuracy in order to provide the optimal efficiency of the servo drives.

The implementation of FOC requires current controllers with fast response and high accuracy in order to provide the optimal efficiency of the servo drives. The controller requires a much faster computation update rate to sustain the closed-loop bandwidth. The processing speed needed to deal with these hardware-rich tasks reaches the order of microseconds to nanoseconds nowadays. Several researchers implemented digital signal processor (DSP) based current controllers in FOC PMSM drives [1, 2]. However, in DSP environment, it is difficult to reach the processing speed in order of nanoseconds [3]. On the other hand, specific hardware technology such as field programmable gate array (FPGA) can be considered as an appropriate solution to boost performances of PI controllers and to reduce their execution time delay. All the sophisticated application, which is now deploying FOC PMSM motors, required robust and fast controllers. In order to reduce the execution time it is necessary to perform the tasks in plain and simple way rather than using the complex circuitry. Better accuracy with minimal execution time is a major concern to realize current controllers in FPGA. Many researchers proposed the hardware implementation of current controller of FOC PMSM drives. Naouar et al. implemented an FPGA based predictive current controller for synchronous machine (SM) speed drive [4]. The limited switching frequency predictive current controller was considered in this research. The implementation was needed 106 latency times. This means, the overall computation time was 2,12 μs for 50 MHz clock, which is still in microseconds range. Reducing the execution time into nanoseconds range along with good accuracy will certainly improve the current dq PI controller performances.

This research shows the FPGA implementation of current dq PI controller in Quartus II Altera environment. The FPGA implementation of this current dq PI controller is executed in very short time with a good accuracy.

2 FPGA implementation of the proposed current dq PI controller

A high-end Altera Stratix IV EP4SGX230KF40C2 FPGA family based on Taiwan semiconductor manufacturing company (TSMC) 40 nm process has been used as target component for the implementation of the proposed controller. The overall block diagram of current dq PI controller module is shown in Fig. 1. The stator voltage (V_{sd}, V_{sq}) signals are calculated from the multiplication of error signal with system depended proportional and integral gain constants. There is also a saturation limit (V_{max}, V_{min}) that existed in the proposed PI controller to control the V_{out}. This protects the system from overshoot and undershoots. According to the block diagram of Fig. 1, the architecture of the overall system can be divided into two mirror components. Each component is partitioned into elementary modules. From

a functional point of view, this partitioning makes the development process simpler.

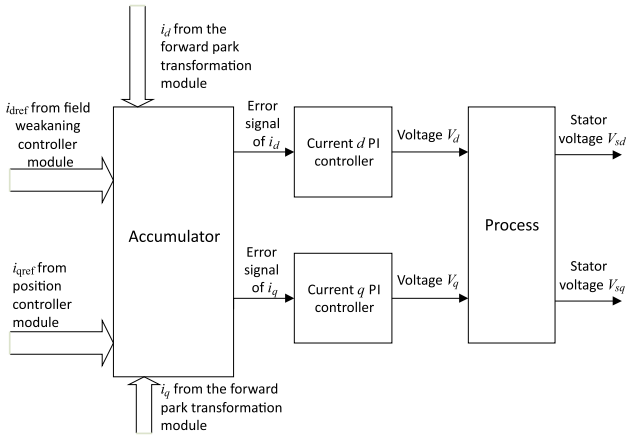

Figure 1 Block diagram of current dq PI controller

The overall process flow of current dq PI controller implementation in FPGA starts by generating the *error*. This *error* is converted to fixed point format for further calculations. As the proportional and integral constants of the system are less than "1" so it can be represented in both floating point and fixed point format. Floating-point numbers can be represented in FPGA by IEEE 754 double precision floating-point number formats. However, it requires 64 bits. In addition, any arithmetic operation can be done only with another IEEE754 standard numbers. All the operands need to be 64 bits. Thus, it needs more pins in FPGA that make the circuits slower and more complex [5]. On the other hand, fixed point format of decimal numbers is used to avoid the complex floating point calculations which do not only reduce the process time but also occupy less logic elements [6]. Fixed point format *Q5.10* is used for FPGA implementation of current dq PI controller as it will assure more than 99,9 % of accuracy. The fixed point format presents all the numbers in [−31,999; 31,999] range. The current dq PI controller has an integral part that is a known mathematical model. However, it is difficult or impossible to find an antiderivative, which is an elementary function. The complexity of integral made it intricate to implement in FPGA. The trapezium method is used to solve the integral calculation. Thus, for quick calculation as well as accurate result, integral portion of the proposed PI controller can be written as

$$Serror_n = Serror_{(n-1)} + \frac{1}{2}\left(error_n + error_{(n-1)}\right) \times clock, \quad (1)$$

where, $Serror_n$ is the integral result and *clock* is the number of clock cycles needed for one iteration.

After calculating the *error* and *Serror*, both of these values are converted to *Q5.10* fixed point format to multiply with K_p and K_i. Thus, the regenerated Eq. (1) can be written as,

$$V_{\text{out}(n)} = \left(K_p \times error\_f_n\right) + \left(K_i \times Serror\_f_n\right), \quad (2)$$

where, *error_f* and *Serror_f* is the fixed point formatted value of *error* and *Serror*, respectively.

Finally, the initial output voltage is checked with the maximum and minimum threshold voltages to control the saturation. The overall process is controlled by a global *reset* and an internal *ready* signal, which is synchronized with the clock. A total of 66 pins are required for input and 32 pins are required for output signals. Several registers are needed for the FPGA implementation. Some system dependant parameters are also used for flexibility of the design. These parameters are defined as constants at the beginning of the design for ease of implementation.

3 Results and Discussion

The FPGA implementation of current dq PI controller has been developed in Quartus II Altera environment. In Quartus II Altera environment, it is possible to simulate the design even with the gate level delays. ModelSim SE 6.3a is a widely used simulator for viewing and analyzing the simulation results of FPGA. Thus, it is used in this research to show the simulation output of this system. Fig. 2 shows the gate level simulation (GLS) result of the proposed current dq PI controller of FOC PMSM drives. The design runs in 30 MHz clock frequency as shown in Fig. 2. So each clock period requires only 34 ns. In Fig. 2 it is shown that if the reset signal is high i.e. "1", the controller will not show any output. It is also clearly shown that the calculated output needs only 2 clock cycles for its execution which is only 68 ns in 30 MHz clock speed.

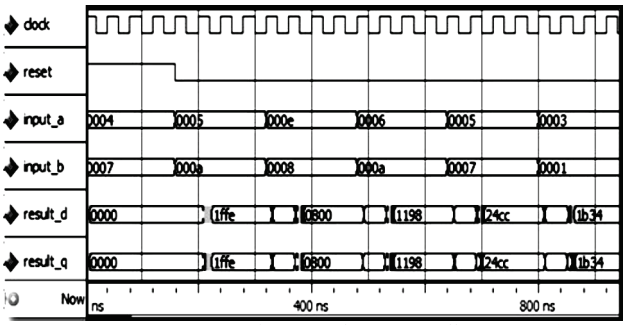

Figure 2 GLS output of Current dq PI Controller (30 MHz)

The design summary of the proposed current dq PI controller is shown in Tab. 1. The hardware implementation of current dq PI controller supports up to 30 MHz clock speed without using any Phase Lock Loop (PLL). It is also shown that the controller needs less than 1 % of total logic elements, total combinational functions as well as total registers of this device. Hence, a small chunk of FPGA resources is used by the module.

Table 1 Design Summary of Current dq PI Controller

Family	Stratix IV
Device	EP4SGX230KF40C2
Total Logic Elements	1182/ 182,400 (<1 %)
Total Combinational functions	1185/182,400 (<1 %)
6 input functions 5 input functions 4 input functions <=3 input functions	59 155 328 643
Total registers	254/ 182,400 (<1 %)
Total block memory bits	0/14,625,792 (0 %)
Clock	Up to 30 MHz
Total Pins	98

To validate the result, test parameters are defined for testing and are compared with the mathematical calculated results. Test parameters defined for testing the design are $K_p = 0{,}6$, $K_i = 0{,}5$, $V_{max} = 20$ and $V_{min} = 2$.

These parameters are converted to *Q5.10* fixed point format and defined at the beginning of the program. Different sets of 16 bits random data are tested for verification. The hand calculation is done according to Eq. (1). For example if $i_d = 5$ and $i_{dref} = 10$ then the hand calculated result (*NR*) is,

$$\begin{aligned} error &= i_{dref} - i_d = 10 - 5 = 5, \\ Serror &= 0 + (1/2)\cdot 5 \cdot 4 = 10, \\ NR &= error \cdot K_p + Serror \cdot K_i = 5 \cdot 0{,}6 + 10 \cdot 0{,}5 = 8. \end{aligned} \quad (3)$$

In simulation results, shown in Fig. 2, all outputs of the proposed module are multiplied by 1024 or 2^{10} according to fixed point format. All of these values are represented in signed decimal number. Therefore, the output values need to be divided by 2^{10} to achieve the original results. Thus for same set of test data the simulated result (*SR*) is,

$$SR = 8190 / 2^{10} = 7{,}99804687. \quad (4)$$

Comparing the *NR* obtained from Eq. (3) with the simulated result obtained from Eq. (4) the accuracy of the module measured is 99,98 %.

Five different sets of random data iare tested for verification and the result is shown in Tab. 2. According to Tab. 2, it is shown that the results are similar though the module has some floating point calculations.

Table 2 Comparison between Simulation and Numerical values

	$K_p = 0{,}6$; $K_i = 0{,}5$; $V_{max} = 20$ and $V_{min} = 2$		$K_p = 0{,}625$; $K_i = 0{,}5$; $V_{max} = 20$ and $V_{min} = 2$	
Input	Simulated result (*SR*)	Numerical result (*NR*)	Simulated result (*SR*)	Numerical result (*NR*)
$i_d = 5$ $i_{dref} = 10$	Result $=(1FFE)_h=(8190)_d=$ $= 8190/2^{10} = 7{,}99804687$	$V_{out} = 0{,}6\cdot 5 + 0{,}5\cdot 10 = 8{,}0$	Result $= (2080)_h = (8320)_d =$ $= 8320/2^{10} = 8{,}125$	$V_{out} = 0{,}625\cdot 5 + 0{,}5\cdot 10 =$ $= 8{,}125$
$i_d = 14$ $i_{dref} = 8$	Result $=(0800)_h = (2048)_d=$ $= 2048/2^{10} = 2$	$V_{out} = 0{,}6\cdot(-6) + 0{,}5\cdot 8 = 0{,}4$ as $V_{out} < V_{min}$, $V_{out} = V_{min} = 2$	Result $= (0800)_h = (2048)_d =$ $= 2048/2^{10} = 2$	$V_{out} = 0{,}625\cdot(-6) + 0{,}5\cdot 8 =$ $= 0{,}25$ as $V_{out} < V_{min}$, $V_{out} = V_{min} = 2$
$i_d = 6$ $i_{dref} = 10$	Result $=(1198)_h = (4504)_d=$ $= 4504/2^{10} = 4{,}3984375$	$V_{out} = 0{,}6\cdot 4 + 0{,}5\cdot 4 = 4{,}4$	Result $= (1200)_h = (4608)_d =$ $= 4608/2^{10} = 4{,}5$	$V_{out} = 0{,}625\cdot 4 + 0{,}5\cdot 4 = 4{,}5$
$i_d = 5$ $i_{dref} = 10$	Result $=(0FFE)_h=(4094)_d=$ $= 4094/2^{10} = 3{,}998046875$	$V_{out} = 0{,}6\cdot 5 + 0{,}5\cdot 2 = 4{,}0$	Result $= (1080)_h = (4224)_d =$ $= 4224/2^{10} = 4{,}125$	$V_{out} = 0{,}625\cdot 5 + 0{,}5\cdot 2 = 4{,}125$
$i_d = 2$ $i_{dref} = 12$	Result $=(5000)_h=(20480)_d=$ $= 20480/2^{10} = 20$	$V_{out} = 0{,}6\cdot 10 + 0{,}5\cdot 32 = 22{,}0$ as $V_{out} > V_{max}$, $V_{out} = V_{max} = 20$	Result $=(5000)_h = (20480)_d =$ $= 20480/2^{10} = 20$	$V_{out} = 0{,}625\cdot 10 + 0{,}5\cdot 32 =$ $= 22{,}25$ as $V_{out} > V_{max}$, $V_{out} = V_{max} = 20$

As the module used *Q5.10* fixed point formats the result is almost the same as decimal values. Again, the modular approach with each elementary module calculation made the results accurate. Instead of processing overall calculation, each of the iterations is performed at once. Negative number manipulation is another challenging task as it may produce wrong outcomes. In this design, negative fixed point numbers are handled separately so that the correctness is ensured without influencing the execution time.

FPGA realization of current dq PI controller for FOC PMSM drive is successfully completed and validated with other researches for comparison as shown in Tab. 3. Instead of using DSP based solution in [1] and [2], this research proposed the FPGA realization of current dq PI controller. Moreover, it shows that this work can accomplish the transformation within 2 clock cycles which means the execution time is as low as 68 ns in 30 MHz frequency. The execution time required in this proposed solution is much smaller than research from [4], [7] and [8]. The proposed design required no memory resources as required in [8].

Table 3 Comparison with previous works

	Implementation	Execution time	Clock cycle	Other information
Jung et al. [1]	DSP	Not Applicable	Not Applicable	Hybrid fuzzy PI controller for current control
Sousy [2]	DSP	Not applicable	Not Applicable	SMC for Current Control
Naouar et al. [4]	FPGA	2,12 μs	106	Predictive current controller is used for SM drive
Rachid et al. [7]	FPGA	>1 μs	-	PI controller with decoupling method is used for current control
Kung et al [8]	FPGA	240 ns	6	PI controller is used for current control
This work	FPGA	68 ns	2	PI controller with fixed point

FPGA realization of the proposed controller performs faster than other research work. Although the proposed controller needs some floating point calculations it provides good accuracy. 30 MHz clock frequency is more than enough for FOC PMSM drive system. Thus, from the comparison study it is observed that this proposed FPGA implementation of current dq PI controller is a faster and accurate solution for a real time current dq PI controller of FOC PMSM drive.

4 Conclusion

A precise, cost effective, simple and high-speed implementation of current dq PI controller is presented in this paper. This FPGA implementation of current *dq* PI controller is of the immense interest because it can offer fast computation and optimal efficiency with stability of the motor system. The gate level simulation result clearly shows that only "2" - clock cycle is required for completing the overall tasks. It means the execution time

required for this hardware implementation is as low as 68 ns in 30 MHz clock frequency. The implementation is tested in three different clock speeds and found that it requires the same clock cycle in all speeds. This FPGA implementation is a prominent contribution in this area of research as it provides the quickest solution among all other research works. Thus, this FPGA realization is an exact and rapid solution of current dq PI controller, which managed to prove that the overall motor system and performance of FOC PMSM have been improved in terms of its efficiency and stability.

Acknowledgements

This work was supported by the Ministry of Science, Technology and Innovation, Malaysia (Technofund: TF1008C130).

5 References

[1] Jung, J. W.; Choi, Y. S.; Leu, V. Q.; Choi, H. H. Fuzzy PI-type current controllers for permanent magnet synchronous motors. // IET Electric Power Applications. 5, 1(2011), pp. 143-152.

[2] El-Sousy, F. F. M. Robust wavelet-neural-network sliding-mode control system for permanent magnet synchronous motor drive. // IET Electric Power Applications. 5, 1(2011), pp. 113-132.

[3] Zhang, D.; Li, H. A Stochastic-Based FPGA Controller for an Induction Motor Drive with Integrated Neural Network Algorithms. // IEEE Transactions on Industrial Electronics. 55, 2(2008), pp. 551-561.

[4] Naouar, M. W.; Ammar Naassani, A.; Monmasson, E.; Slama Belkhodja, I. FPGA-Based Predictive Current Controller for Synchronous Machine Speed Drive. // IEEE Trans. on Power Electronics. 23, 4(2008), pp. 2115-2126.

[5] Marufuzzaman, M.; Reaz, M. B. I.; Ali, M. A. M.; Rahman, L. F. Hardware Approach of Two Way Conversion of Floating Point to Fixed Point for Current dq PI Controller of FOC PMSM Drive. // Electronics and Electrical Engineering. 7, 123(2012), pp.79-82.

[6] Dong, W.; Ercegovac, M. D.; Nanning, Z. Design of High-Throughput Fixed-Point Complex Reciprocal/Square-Root Unit. // IEEE Trans. Circuits and Syst. II: Express Briefs. 57, 8(2010), pp. 627-631.

[7] Rachid, B.; Jean-Gabriel, M.; Stephane, S.; Arnaud, T. Towards the System-on-Chip Realization of a Sensorless Vector Controller with Microsecond-order Computation Time. // Proceedings of Canadian Conference on Electrical and Computer Engineering (CCECE '06), Ottawa, Canada, May 2006, pp. 1073-1077.

[8] Rodriguez-Andina, J. J.; Moure, M. J.; Valdes, M. D. Features, Design Tools, and Application Domains of FPGAs. // IEEE Transactions on Industrial Electronics. 54(2007), pp. 1810-1823.

[9] Ying-Shieh, K.; Ming-Hung, T. FPGA-Based Speed Control IC for PMSM Drive with Adaptive Fuzzy Control. // IEEE Transactions on Power Electronics. 6, 22(2007), pp. 2476-2486.

Authors' addresses

Mohammad Marufuzzaman
Faculty of Engineering and Built Environment
Department of Electrical, Electronic and Systems Engineering
Universiti Kebangsaan Malaysia
43600 Bangi, Selangor, Malaysia
E-mail: marufsust@gmail.com

Mamun Bin Ibne Reaz, Professor
Faculty of Engineering and Built Environment
Department of Electrical, Electronic and Systems Engineering
Universiti Kebangsaan Malaysia
43600 Bangi, Selangor, Malaysia
E-mail: mamun.reaz@gmail.com

Labonnah Farzana Rahman
Faculty of Engineering and Built Environment
Department of Electrical, Electronic and Systems Engineering
Universiti Kebangsaan Malaysia
43600 Bangi, Selangor, Malaysia
E-mail: labonnah.deep@gmail.com

Tae Gyu Chang, Professor
School of Electrical and Electronics Engineering
Chung-Ang University
221, Heuksuk-dong, Dongjak-ku
Seoul 156-756, South Korea
E-mail: tgchang@cau.ac.kr