A Flexible SVPWM implemented in FPGA

K.I.P.M. Queiroz, F.R. de Sousa, R.L.A. Ribeiro, E.C. Braz

DEE/CT- Universidade Federal do Rio Grande do Norte
Campus Universitario, Lagoa Nova, CEP 59072-970, Natal-RN, Brazil
Email: kuriosiuri@ieee.org, rangel@ct.ufrn.br, rlucio@ct.ufrn.br, cardineli@ufrnet.br

***Abstract*— This paper presents an FPGA implementation of a flexible SVPWM (F-SVPWM) control strategy for three-phase motor drive systems. Instead of traditional strategies, it uses a flexible-vector sequence scheme which allows to apply voltage vectors in any desired order. In addition, the proposed scheme can reduce the load common mode voltage and can minimize the harmonic distortion of the load phase currents. To achieve these improvements, we developed an optimized dedicated single processor. The proposed F-SVPWM was simulated in the Altera Quartus II environment and then verified experimentally using an Altera Cyclone II EP2C35 prototype board.**

I. Introduction

The Space Vector Pulse Width Modulation (SVPWM) is one of the most popular technologies in the AC servo drive systems [1]. Its implementation is normally based on Microcontrollers (MCUs) or Digital Signal Processor (DSP) [2]. The former is widely employed due to easy of use and flexibility, nevertheless the speed is very limited. The later is often used to increase the switching frequency. However, both of them suffer some disadvantages, such as long development period and large use of CPU. The realization of SVPWM in a FPGA provides many benefits, e.g. shorter design cycle and fair cost. The reconfigurable hardware and the fast circuit response (due to the simultaneous instead of sequential execution) are additional advantages of FPGA based designs.

Although there are others SVPWM realizations based on FPGA, this new design offer more flexibility when compared to the standard techniques. Dead time and overmodulation is considered in [3], but all the parameters necessary for its operation are stored in ROM memory. This is not good solution for real applications, since reprogramming the device is necessary. The SVPWM designs presented in [3], [4], [5] and [6], it is impossible to produce the vectors sequence in a desired order. They always apply the null vectors as the same way, according to asymmetrical or symmetrical mode.

Despite the fact that standard PWM techniques are very used in many applications in industry, they generate common-mode voltages, which disturbs the electrical machines connected to the power lines, causing bearing and seal deterioration [7]. The common-mode currents also cause activation of ground fault detection circuits, electromagnetic interference and heat the conduits it through. In the literature, there are many ways to eliminate or at least reduce the common-mode voltage ([8], [9] and [10]) . In [11], the authors have proposed the utilization of a new machine drive configuration (with four-phase) that permits to eliminate the common-mode voltage. In this case, the SVPWM employed to control the power switches of the four-phase inverter is a slightly different from the usual realizations. It should to allow changes in the sequence of output vectors, periodically. This technique is also applied in six-phase inverters and six-phase induction motor.

In this paper, we propose a flexible SVPWM (F-SVPWM) based on FPGA for a three-phase motor drive system. It has an input parameter which defines the output sequence of vectors application. It is important to notice that in three-phase systems, it is not possible to eliminate the common-mode voltage, though we can minimize its magnitude by changing the order of output vectors. Owing to the reconfigurable hardware, the system designed can be adapted in the future to control four-phase or six-phase induction motors. There are others advantages in the use of this strategy, e.g., the minimization of harmonic distortion and switching losses in the converters.

The digital system for the SVPWM was modeled as the interconnection of the two modules: a datapath, which contains the data register, the operator, the multiplexers and the busses; a controller wich generates the control signals for the datapath, and the sequence of steps to perform the overall computation [12]. This model is very interesting in complex systems, because it reduces the difficulty observed when only one finite state machine (FSM) is used. This approach can be so recursive as we wish.

This paper is organized as follows. The next section describes the basic space-vector theory used. Section III explains some definitions required in the proposed scheme and the all modules involved in the digital system. The simulation and experimental results are illustrated in section IV and finally, some conclusions are drawn in section V.

II. Standard Space-Vector

Fig. 1 shows the a simplified schematic diagram of the system considered in this work which is composed of a three-phase inverter, an induction motor and a power source. The latter is obtained by filtering and rectifying the three-phase power supply (380V, 60 Hz) and the induction motor has windings connected in a wye configuration. The inverter has 6 switches (q_1, q_2, q_3, q_4, q_5 and q_6) and their respective diodes. However, only three of them (q_1, q_2 and q_3) will be analyzed because q_1q_4, q_2q_5 and q_3q_6 are complementary. As a result, there are eight different combinations of them, and thus, eight voltage vectors in which, two are null and six are active.

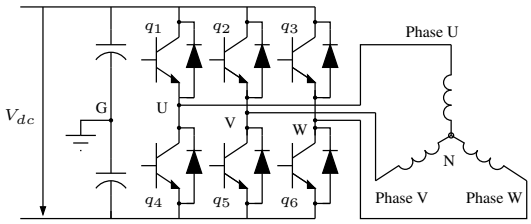

Fig. 1. Simplified schematic diagram of the three-phase system.

The space-vector technique allows to synthesizing a desired vector $\vec{V}_{ref}$ from two adjacent active vectors, $\vec{V}_a$ and $\vec{V}_b$ (among $\vec{V}_1$-$\vec{V}_6$, see Fig. 2), and the null vectors ($\vec{V}_7$ and $\vec{V}_8$). In the proposed design, two null vectors are inserted in a PWM period (T), one for each half ($T/2$), in any desired order. The vector's magnitude can be expressed as follows

$$\vec{V}_n = \frac{2}{3} V_{dc} e^{j\frac{(n-1)}{3}}, n = 1, 2, \ldots, 6$$

$$\vec{V}_7 = \vec{V}_8 = 0$$

where V_{dc} is the dc-link voltage.

The vector $\vec{V}_{ref}$ has two components V_α and V_β as shown in Fig. 2 and it can be synthesized from the vectors $\vec{V}_x$ and $\vec{V}_y$ that corresponds to applying the active vectors ($\vec{V}_a$ and $\vec{V}_b$) during time intervals, t_a and t_b. The null vectors are also applied to reduce the inverter switching frequency.

$$\vec{V}_{ref} = V_\alpha + jV_\beta = \vec{V}_x + \vec{V}_y = \frac{2t_a}{T}\vec{V}_a + \frac{2t_b}{T}\vec{V}_b$$

$$\frac{T}{2} = t_a + t_b + t_0$$

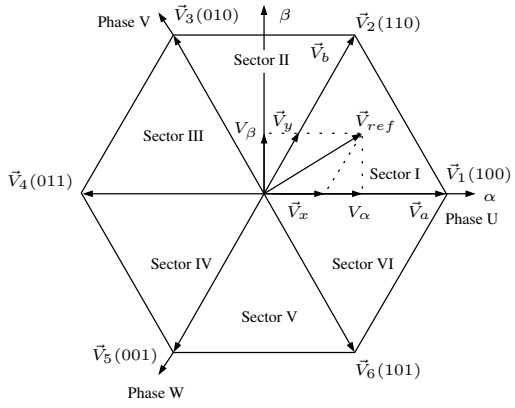

Fig. 2. Voltage space-vector diagram.

Hence, the half PWM period T is composed by the switching times t_a, t_b and t_0. The total time of the null vectors can be expressed as

$$t_0 = \frac{T}{2} - t_a - t_b \quad (1)$$

and the switching time of the active vectors for each sector can be calculated as shown in table I.

TABLE I
SWITCHING TIMES FOR SECTOR

Sector	$\vec{V}_a$	t_a	$\vec{V}_b$	t_b
I	$\vec{V}_1$	$\frac{3T}{4}(\frac{V_\alpha}{V_{dc}} - \frac{V_\beta}{\sqrt{3}V_{dc}})$	$\vec{V}_2$	$\frac{3T}{4}\frac{2V_\beta}{\sqrt{3}V_{dc}}$
II	$\vec{V}_3$	$\frac{3T}{4}(-\frac{V_\alpha}{V_{dc}} + \frac{V_\beta}{\sqrt{3}V_{dc}})$	$\vec{V}_2$	$\frac{3T}{4}(\frac{V_\alpha}{V_{dc}} + \frac{V_\beta}{\sqrt{3}V_{dc}})$
III	$\vec{V}_3$	$\frac{3T}{4}\frac{2V_\beta}{\sqrt{3}V_{dc}}$	$\vec{V}_4$	$\frac{3T}{4}(-\frac{V_\alpha}{V_{dc}} - \frac{V_\beta}{\sqrt{3}V_{dc}})$
IV	$\vec{V}_5$	$-\frac{3T}{4}\frac{2V_\beta}{\sqrt{3}V_{dc}}$	$\vec{V}_4$	$\frac{3T}{4}(-\frac{V_\alpha}{V_{dc}} + \frac{V_\beta}{\sqrt{3}V_{dc}})$
V	$\vec{V}_5$	$\frac{3T}{4}(-\frac{V_\alpha}{V_{dc}} - \frac{V_\beta}{\sqrt{3}V_{dc}})$	$\vec{V}_6$	$\frac{3T}{4}(\frac{V_\alpha}{V_{dc}} - \frac{V_\beta}{\sqrt{3}V_{dc}})$
VI	$\vec{V}_1$	$\frac{3T}{4}(\frac{V_\alpha}{V_{dc}} + \frac{V_\beta}{\sqrt{3}V_{dc}})$	$\vec{V}_6$	$-\frac{3T}{4}\frac{2V_\beta}{\sqrt{3}V_{dc}}$

III. Digital Hardware Implementation

In order to present the SVPWM design, some preliminary definitions based on the previous section equations are necessary, such as *modulation index* (m_α and m_β), *scale factor* (k), *frequency register* (*freqreg*) and *synchronous pulse* (*syncpulse*). The two first ones are used to calculate the switching times and can be expressed by the equations 2, 3 and 4. Both are input parameters, but m_α and m_β are considered as fixed-point fractions and the k as an integer. Another input parameter (an integer) is the $freqreg$ which is employed to adjust the PWM period conveniently (see equation 5). The last definition, *syncpulse*, is a pulse signal generated at the beginning of each PWM half period. It is used to synchronize the internal system modules as well as the external blocks connected to the SVPWM.

$$m_\alpha = \frac{V_\alpha}{V_{dc}} \quad (2)$$

$$m_\beta = \frac{V_\beta}{V_{dc}} \quad (3)$$

$$k = \frac{3T}{4} = \frac{3}{2}(freqreg + 1) \quad (4)$$

$$T = 2(freqreg + 1)T_{clk} \quad (5)$$

The digital system (DS) was developed using the Finite State Machine with Datapath (FSMD) model [12] which divides the system in two blocks, a datapath and a controller, as described in Fig. 3. Other input parameters are the system

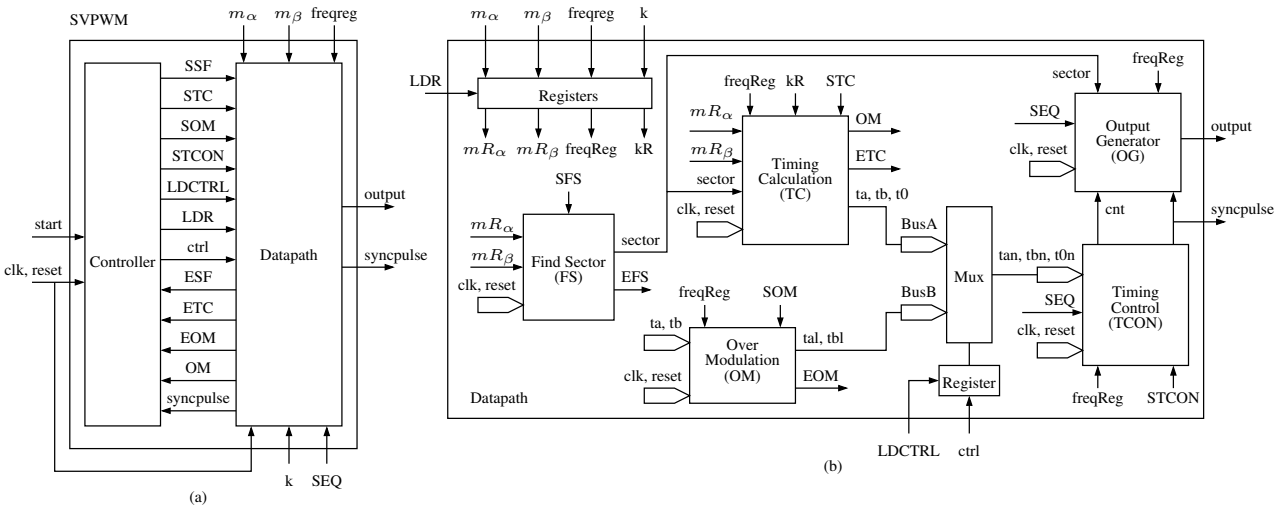

Fig. 3. (a) Block diagram of the proposed F-SVPWM. (b) Block diagram of the Datapath.

clock (*clk*), the start signal (*start*), the reset signal (*reset*) and the sequence of output vectors (*SEQ*). The controller is represented by a finite state machine (*Moore* machine) and the datapath is composed of five modules which were also modeled using the FSMD model. The functions performed by the DS are:

1) Identify the reference vector ($\vec{V}_{ref}$) sector.
2) Calculate the switching times from table I.
3) Recalculate the switching times, if overmodulation exists.
4) Generate the output vectors.
5) Control the time vectors application.

Initially (see Fig. 4), the controller waits for a *start* signal (state *Start*) and then loads the datapath registers (state *Load*). After that, the SF module is started and works for some clock periods (state *Sector*). The controller keeps inactive waiting for SF to finish (ESF=1), since the other modules need $\vec{V}_{ref}$ sector information. When the sector is found, the controller starts the TC module (state *Times*) and waits until it finishes (ETC=1). This module also informs (through the signal OM) if overmodulation is observed, allowing the controller to activate or not the OM module[1]. Finally, the TCON module is started (state *TCON*), and by its turn, activates the OG module using the *syncpulse* signal (state *Output*) [2].

When the datapath registers are loaded, the signals mR_α, mR_β, $freqReg$ and kR are available for all internal datapath modules and remain the same until a new cycle starts. The load action is realized by the controller through the *LDR* signal. Each datapath module starts only when a specific signal arrives with a high logical level. Except for the OG module, the others begin their activities with signals from the controller: SSF (*start SF module*), STC (*start TC module*), SOM (*start OM module*) and STCON (*start TCON module*). As mentioned before, the OG module is started by the TCON module using the *syncpulse* signal. Both (OG and TCON) never stop working, except when the user pushes the reset button and consequently generates the *reset* signal. This action reset all the system's blocks, including the controller. If the user wishes to initiate the system again, a new *start* signal is necessary.

The signals *tan*, *tbn* and *t0n* in Fig. 3 receive values from TC module (*ta*, *tb* and *t0*) or from OM module (*tal* and *tbl*) according to the overmodulation detection executed by the TC module. When it is positive (that is, overmodulantion exists) the controller changes the input TCON values using the *ctrl* signal. This choosing have to be the same during some clocks and consequently a register is necessary [3].

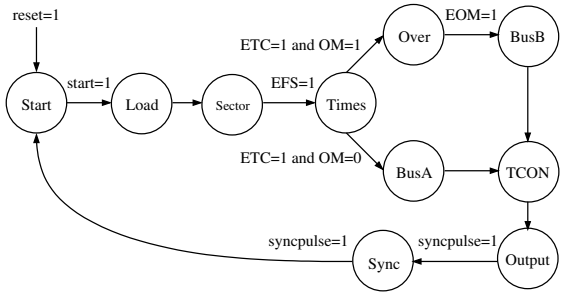

Fig. 4. State Diagram of Controller.

A. Sector Finder (SF) Module

Due to existing different switching time equations, as shown in table I, the reference voltage sector knowledge is necessary. The SF module is responsible for finding the reference vector sector and does it using the following conditions:

If $0 \leq V_\beta < \sqrt{3}V_\alpha$, then $\vec{V}_{ref}$ locates in the sector I;

[1]The states *BusA* and *BusB* configure the internal buses in the datapath and the OM module is started by the state *Over*. The buses are chosen using the *ctrl* signal.

[2]The state *Sync* is only for synchronizing the controller.

[3]The LDCTRL signal is used by the controller to load the register.

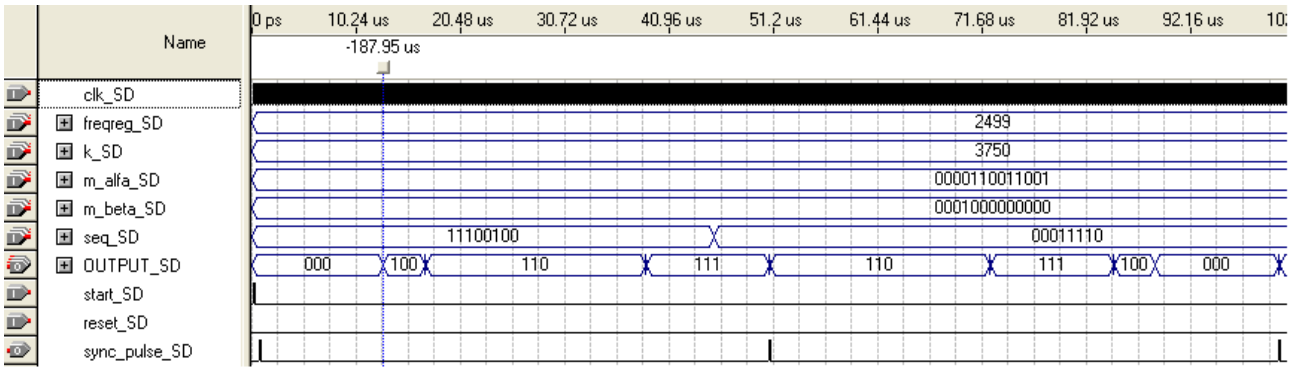

Fig. 5. Simulation results using constant modulation indexes in sector I ($m_\alpha = 0.2$, $m_\beta = 0.25$).

If $V_\beta \geq \sqrt{3}V_\alpha$ and $V_\alpha \geq 0$, or $V_\beta > -\sqrt{3}V_\alpha$ and $V_\alpha < 0$, then $\vec{V}_{ref}$ locates in sector II;

If $0 < V_\beta \leq -\sqrt{3}V_\alpha$, then $\vec{V}_{ref}$ locates in sector III;

If $\sqrt{3}V_\alpha < V_\beta \leq 0$, then $\vec{V}_{ref}$ locates in sector IV;

If $V_\beta \leq \sqrt{3}V_\alpha$ and $V_\alpha \leq 0$, or $V_\beta < -\sqrt{3}V_\alpha$ and $V_\alpha > 0$, then $\vec{V}_{ref}$ locates in sector V;

If $-\sqrt{3}V_\alpha \leq V_\beta < 0$, then $\vec{V}_{ref}$ locates in sector VI;

Instead of V_α and V_β in above statements, we can use the *modulation index* components to select the appropriate sectors.

B. The Timing Calculation (TC) Module

This module calculates the switching times which compose the PWM period. The t_a and t_b times are shown in table I and t_0 in equation 1, as mentioned before. However, t_0 is still divided in two parts, one for each null vector, assigned as follows

$$t_7 = t_8 = \frac{T/2 - t_a - t_b}{2}$$

The input parameters m_α, m_β and k allow this module to obtain the switching times using only multipliers and adders. The addition operation is executed firstly and its result is multiplied to the *scale factor* (k).

C. The Over Modulantion (OM) Module

When $t_a + t_b$ is greater than $T/2$, t_0 is negative and the overmodulation occurs. Therefore, the switching times are recalculated by the OM module using the old time values. The new values ($\bar{t}_a$ and $\bar{t}_b$) are obtained from (6) (if $t_a > t_b$) and (7) (if $t_b > t_a$) . To avoid the overmodulation, the input parameters m_α and m_β have to be lower than $1/\sqrt{3}$.

$$\bar{t}_a = \frac{t_a}{t_a + t_b}\frac{T}{2}, \bar{t}_b = \frac{T}{2} - \bar{t}_a \quad (6)$$

$$\bar{t}_b = \frac{t_b}{t_a + t_b}\frac{T}{2}, \bar{t}_a = \frac{T}{2} - \bar{t}_b \quad (7)$$

D. Output Generator (OG) and Timing Control (TCON) Modules

These modules generate the PWM outputs based on the information from the other modules, e.g. switching time values, $\vec{V}_{ref}$ sector and overmodulation detected. The OG chooses the correct combination for the outputs signals (*output*) and keeps it until its duration time finishes. On the other hand, the TCON informs to OG when outputs should be changed.

IV. Experimental Results

The experimental results were obtained using the *Signal Tap II Logic Analyzer* and the Altera Cyclone II EP2C35 prototype board. Previously, the F-SVPWM was simulated in the Altera Quartus II environment with constant *modulation indexes*[4] in sector I ($m_\alpha = 0.2$, $m_\beta = 0.25$). The order of output vectors were changed in the half PWM period, through the input parameter SEQ. Each output vector has a binary code in the SEQ byte ($\vec{V}_7 = 00$, $\vec{V}_8 = 11$, $\vec{V}_a = 01$ and $\vec{V}_b = 10$), where the output sequence is defined according to SEQ bits grouped two by two (from right to left). The two first bits correspond to the first output vector code, the two next bits correspond to the second output vector code and thus successively until the last bits. The output vectors in the simulation are $\vec{V}_7 = 000$, $\vec{V}_a = 100$, $\vec{V}_b = 110$ and $\vec{V}_8 = 111$ for the first half period and $\vec{V}_b = 110$, $\vec{V}_8 = 111$, $\vec{V}_a = 100$ and $\vec{V}_7 = 000$ for the second. The results can be viewed in Fig. 5, where clk=50 MHz, $freqreg = 2499$ and $k = 3750$. The PWM frequency was 10 KHz.

The same simulation input parameters were used in the experimental procedure, except *modulation indexes* and SEQ value (the output sequence was $\vec{V}_7$, $\vec{V}_a$, $\vec{V}_b$ and $\vec{V}_8$). The parameters m_α and m_β were changed for each PWM period, according to a sinusoidal signal. The experimental data for q_1 were acquired from the FPGA (sampling frequency=50 MHz) using *Signal Tap* tool, stored in a text file and then plotted in a *Gnuplot* graphic (see Fig. 6). The sinusoidal reference was also plotted in the same graphic, but its values were changed

[4]The modulation indexes correspond to fixed-point fractions with 13 bits, where the first bit (from left to right) represents the signal, the second bit represents the integer part and the others represent the fraction part.

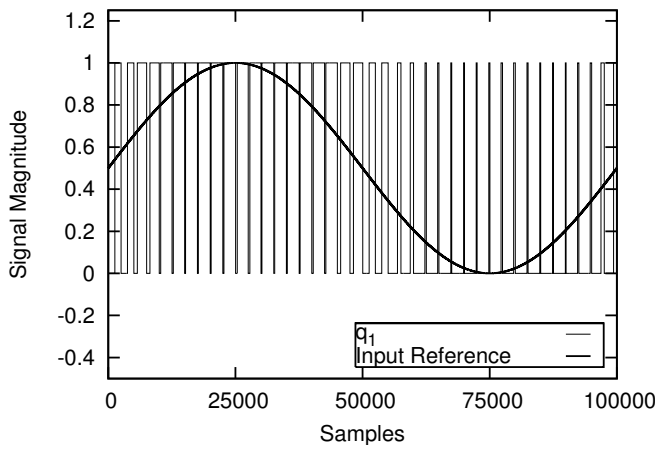

Fig. 6. Experimental results (q_1 output signal) using *modulation indexes* as a sinusoidal signal.

to fit in the [0,1] magnitude interval. We can verify that the output width pulses agree with the signal input reference.

V. Conclusion

This paper presents a flexible implementation of space vector pulse width modulation (F-SVPWM) technique on FPGA. Its main feature corresponds to reduce the load common mode voltage and minimize the harmonic distortion of the load phase currents. These results can be obtained changing the order of output vectors for each half PWM period. Therefore, SVPWM proposed uses as an input parameter, the desired order for these vectors. Experimental results and simulations were achieved using *Signal Tap Logic Analyzer* and *Quartus II* from Altera. Sinusoidal signals as well as constant values were tested as *modulation indexes*.

The Finite State Machine with Datapath (FSMD) model was employed in all system's modules design, reducing its implementation complexity. Moreover, due to the reconfigurable hardware FPGA feature and the modularity presented by the FSMD model, the system can be adapted to control four-phase or six-phase induction motors. In the future, the SVPWM will be used in a closed loop system in order to obtain experimental results and validation tests with three-phase inverter and induction motors.

Acknowledgment

The authors wish to thank the INPE (National Institute for Space Research) and Federal University of Rio Grande do Norte (UFRN) for technical support, and the National Council of Technological and Scientific Development (CNPq) for the financial support. The authors also wish to thank Dr. Alexandre Nowasad for the assistance in the preparing of this paper.

References

[1] Y. Y. Tzou and H. J. Hsu, "FPGA realization of space-vector PWM control IC for three-phase PWM inverters," *IEEE Trans. Power Electron.*, vol. 12, pp. 953–963, 1997.

[2] Y. Gui-jie, S. Li, C. Nai-zheng, and L. Yong-ping, "Study on method of the space vector PWM," in *Proc. of the CSEE*, vol. 21, China, 2001, pp. 79–83.

[3] W. P. Hew, C. P. Ooi, and N. A. Rahim, "Realization of space vector modulation technique in a single FPGA chip for induction motor drive PWM," in *IEEE International Conference on Electron Devices and Solid-State Circuits (EDSSC'2005)*, Hong Kong, Dec. 2005, pp. 817–820.

[4] Z. Zhou, T. Li, T. Takahashi, and E. Ho, "Design of a universal space vector PWM controller based on FPGA," in *IEEE Applied Power Electronics Conference and Exposition (APEC '2004)*, vol. 3, Hong Kong, Dec. 2004, pp. 1698–1702.

[5] T. Wiangtong and P. Dechsuwan, "Unified motor controller based on space vector modulation technique," in *IEEE International Symposium on Circuits and Systems (ISCAS'2006)*, 2006, pp. 5635–5638.

[6] Z. Zhou, T. Li, T. Takahashi, and E. Ho, "Realization of an FPGA-based space-vector PWM controller," in *IEEE International Power Electronics and Motion Control (IPEMC'2006)*, 2006.

[7] M. J. Costello, "Shaft voltages and rotating machinery," *IEEE Trans. Ind. Applicat.*, pp. 419–426, 1993.

[8] Y. Murai and Y. Kawase, "Leakage current reduction for a high-frequency carrier inverter feeding and induction machine," *IEEE Trans. Ind. Applicat.*, pp. 858–863, 1992.

[9] S. Ogasawara and H. Akagi, "Modelling and damping of high-frequency leakage currents in PWM inverter-fed AC motor drive systems," *IEEE Trans. Ind. Applicat.*, pp. 1105–1114, 1996.

[10] A. L. Julian, T. A. Lipo, and G. Oriti, "Elimination of the common-mode voltage in three-phase sinusoidal power converter," in *In Conf. Rec. PESC IEEE*, 1996, pp. 1968–1972.

[11] C. C. Azevedo, R. L. A. Ribeiro, A. M. N. Lima, E. R. B. Filho, and T. M. Oliveira, "Eliminating the common-mode voltage in AC drive systems using a four-phase machine," in *Proc IEEE Industry Applications Society Conference (IAS'2001)*, Chicago, 2001, pp. 859 – 866.

[12] D. Borrione, J. Dushina, and L. Pierre, "Formalization of finite state machines with data path for the verification of high-level synthesis," in *Proc. XI Brazilian Symposium on Integrated Circuit Design*, 1998, pp. 99–102.