International Journal of Electrical, Electronic and Communication Sciences
ISSN: 2517-9438
Vol:9, No:8, 2015

FPGA Based Implementation of Simplified Space Vector PWM Algorithm for Multilevel Inverter Fed Induction Motor Drives

Tapan Trivedi, Pramod Agarwal, Rajendrasinh Jadeja, Pragnesh Bhatt

***Abstract*—**Space Vector Pulse Width Modulation is popular for variable frequency drives. The method has several advantages over carried based PWM and is computation intensive. The implementation of SVPWM for multilevel inverter requires special attention and at the same time consumes considerable resources. Due to faster processing power and reduced over all computational burden, FPGAs are being investigated as an alternative for other controllers. In this paper, a space vector PWM algorithm is implemented using FPGA which requires less computational area and is modular in structure. The algorithm is verified experimentally for Neutral Point Clamped inverter using FPGA development board xc3s5000-4fg900.

***Keywords*—**Modular structure, Multilevel inverter, Space Vector PWM, Switching States.

I. Introduction

IN recent years, the demand of induction motor drives in medium and high power has increased. Due to limited rating of switches and problems due to series and parallel combination of them, multilevel inverters have become popular. Various topologies like diode clamped [1], flying capacitor and H-Bridge [3] inverter have been proposed out of which neutral point clamped inverter has gained wide attention in medium voltage high power induction motor drives [2]. At the same time, Space vector Pulse Width Modulation is found to be the most suitable for variable frequency drives because of better DC link utilization [4] and optimal harmonic reduction. Since the method is computational intensive, its realization has attracted many researchers especially in the multilevel inverter domain. Many methods have been proposed [5]-[7] for simplification of SVPWM algorithm, most of them using present day DSP or microcontrollers. At the same time FPGAs have also been considered due to their ability to accommodate computational intensive and signal processing circuits. FPGA basically serves two purposes: (1) introduction of dedicated chip for generation of PWM signals can relieve processor from making complex calculation and hence can be used for other calculations such as speed and flux estimation. (2) Since no. of PWM pins/channels are limited in present day DSP controller, introduction of FPGA can avoid necessity of more PWM units for higher level of inverters in single chip of microcontroller. In [8], authors use non-orthogonal reference frame method for generation of signals. To address m no. of phases and n levels, two papers [9], [10] implemented simplified FPGA module for generic multiphase multilevel inverters. However, they require another processing unit for generation of no. of references used resulting in extra circuitry.

Tapan Trivedi and Rajendrasinh Jadeja are with Marwadi Education Foundation Group of Institutions, Rajkot, India (e-mail: tapankumar.trivedi@marwadieducation.edu.in).

Pramod Agarwal is with Indian Institute of Technology, Roorkee, Uttarakhand India (e-mail: pramgfee@iitr.ernet.in).

Pragnesh Bhatt is with the M & V Patel Electrical Engineering Department, Charotar University of Science and Technology, Changa, India, (e-mail: pragneshbhatt.ee@charusat.ac.in).

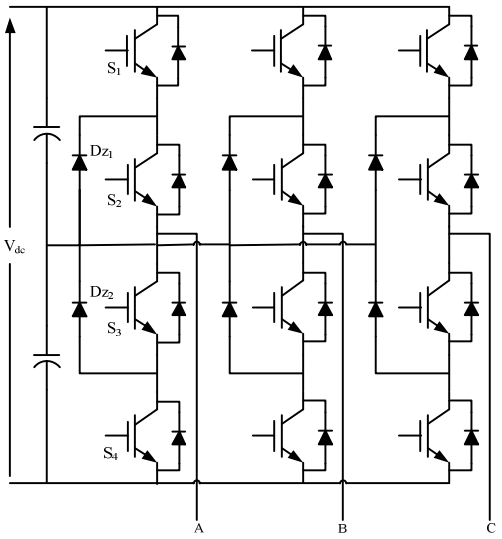


Fig. 1 Power Circuit Diagram of Diode Clamped Multilevel Inverter

In present work, a SVPWM algorithm which can be used for control of open loop variable frequency drive is implemented on FPGA. As acceleration or deceleration of drive leads to unbalanced capacitor voltages, redundant vectors are used to mitigate the problem. A single command in terms of frequency is provided to the modular structure which can be optimized individually to suit the application.

This paper is organized into mainly four sections. In second section, inverter topology and SVPWM algorithm are discussed. In third section, simplified algorithm for FPGA implementation is presented which is followed by experimental results and conclusion.

II. Inverter and Control Strategy

A. Diode Clamped Multilevel Inverter

A very popular topology of multilevel inverter which is widely used for induction motor drives is shown in Fig. 1. This topology is also known as Neutral Point Clamped Inverter [1]. In Diode clamped multilevel inverter, no. of switch pairs are given by (m-1) whereas each pair is clamped to V_{dc}/(m-1) voltage by clamping diode. The DC link voltage is shared by (m-1) no. of capacitors connected in series and they share equal amount of voltages. When inner switches (such as S_2 and S_3) are turned on, the output current is provided by the capacitors of finite value which results in charging and discharging of capacitors resulting in unbalanced voltage sharing across capacitor [2]. These deviations are quite large during fluctuating load or acceleration and deceleration. Hence, voltage balancing mechanism has to be adopted to avoid excessive stress across either device or premature failure of the device. The device switching states to be used in this paper are mentioned in Table I

TABLE I
Device Switching States Assignment [2]

Switching states	Device switching states				Inverter terminal voltage Vaz
	S_1	S_2	S_3	S_4	
P	1	1	0	0	$V_{dc}/2$
O	0	1	1	0	0
N	0	0	1	1	$-V_{dc}/2$

B. SVPWM Algorithm for Three Level Inverter Topology

The Space Vector and their switching states for three level inverter are shown in Fig. 2 with region identifications. The diagram is divided into six identical sectors. Each sector has 4 regions which are also knows as triangles. The dwell times for the respective states for adjacent voltage vectors are calculated using "NTV" approach.

As shown in Fig. 2, there are $3^3 = 27$ switching states out of which $(3\text{-}1)^3=8$ are redundant and hence switching sequence considers only 19 switching states; which are divided into small, medium, large and zero vectors. Various methods have been proposed for simplification of on-time calculation in region. However, due to simplicity and ease of implementation, method described in [7] is considered for dwell time calculation.

Using this method, the on times for three vectors in respective region are given by:

$$T_a = T_s\left[v_{\alpha 0} - \frac{v_{\beta 0}}{2h}\right] \quad (1)$$

$$T_b = Ts\left[\frac{v_{\beta 0}}{h}\right] \quad (2)$$

$$T_c = T_s - T_a - T_b \quad (3)$$

where $v_{\alpha 0}$ and $v_{\beta 0}$ are values of v* along α_0- β_0 axis.

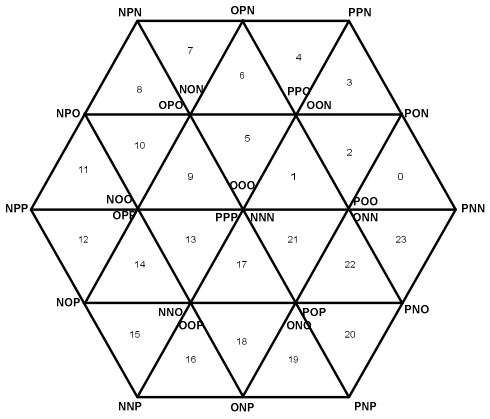
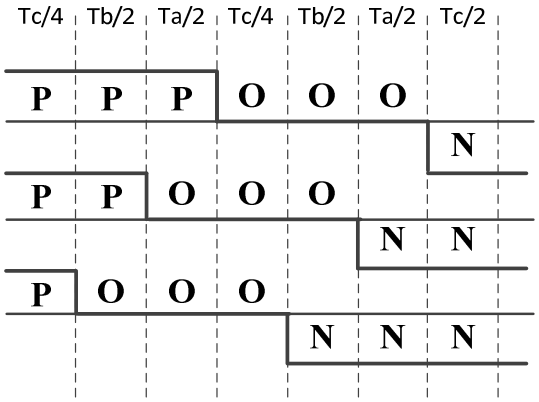
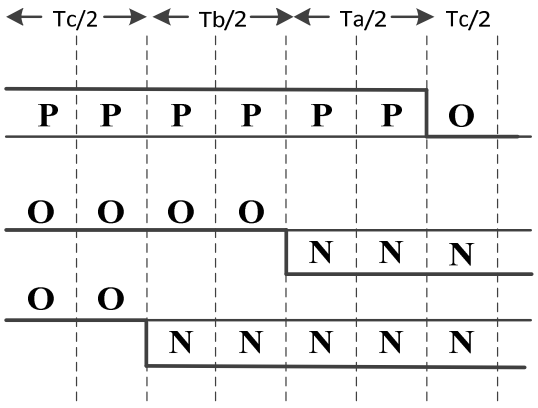


Fig. 2 SVPWM switching states for three phase three level inverter

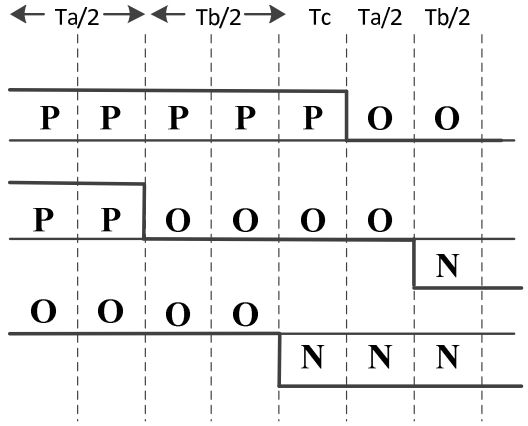
Device switching sequence is prepared using 13-segments over one sample time T_s (where $T_s = 2*T_c$) instead of 7-segment as per the switching sequence criteria shown in Fig. 3. Due to this, uniformity is maintained in the algorithm for FPGA and single unit can be formed to select segment and dwell time instead separate units.



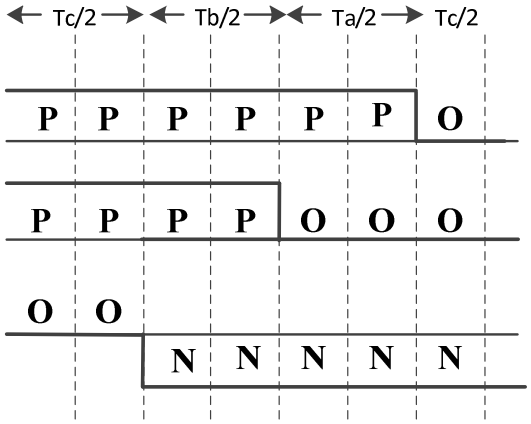
(a)



(b)



(c)



(d)

Fig. 3 Single side Switching Sequence for three level NPC inverter in different regions (a) region-I (b) region-II (c) region-III (d) region-IV

As it can be seen from Figs. 3 (a) and (c) that redundant vectors are used and hence it balances the charging and discharging of capacitors.

III. FPGA Based Implementation of Algorithm

The Space Vector Pulse Width Modulation algorithm is a computation intensive algorithm which involves sine-cosine values retrieval, region and sector calculation, dwell time calculation, and switching state assignment. Since software Dead band is more appropriate and utilizes fewer resources; its generation should be an integral part of PWM unit. Hence, a modular structure which treats these tasks as separate units is more suitable.

Fig. 4 shows implementation of this multiple modules and their communication. Module1 is having combinational and/or sequential logic and calculates necessary parameters. The output of module1 is shared to module2 which takes it as the input parameter and processes it. The shared memory acts as buffer until next enable signal is received from the controller. For example, in Fig. 5 discrete integrator acts as module 1, it calculates *theta* which is shared to sector identification module. In the current work, a pair of *To* and *From Register* is used as shared memory since it requires minimum space.

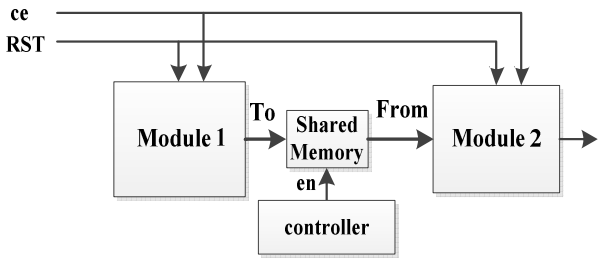


Fig. 4 Basic Description of modular structure

The overall system is divided into various sections namely; discrete integrator, sector and region identification, v_α and v_β calculation, dwell time calculation, ON period allocation and segment calculation, switching state assignment. Most of the calculations are done using 16-bit numbers with appropriate decimal places. These sections are explained as follows:

A. Discrete Integrator

An input of f^* is given in per unit form through input pins of FPGA. This input is in 8-bit and hence any 8-bit ADC or controller can be used for input to the FPGA. The integrator is implemented using Backward Euler method which is given by:

$$y(k) = y(k-1) + T_s * u(k) \tag{4}$$

where u(k) is the frequency input, and y(k) is in the form of angle. From the equation (4), it is evident that only an adder and a delay component are used. Adder saturates after its maximum defined value which can be further scaled to get $\theta = \omega_e t$. The value of m* can be implemented using a lookup table. For variable frequency drive, the value of modulation index should be set such that it compensates for the stator impedance drop at lower frequency.

B. Sector & Angle Calculation

There are two approaches to implement this module. In first method, a lookup table can be used to find sector from the angle directly using nearest output value or division by 60˚ to obtain sector identity from which sector can be calculated. However both methods occupies more resources and hence a simplified approach is presented here. The sector can be presented as

$$S^* = \text{int}\left(\frac{\theta}{\left(\pi/3\right)}\right) \tag{5}$$

Hence, it can be rewritten as

$$S^* = \text{int}\left(0.95493 \times \theta\right) \tag{6}$$

where, S^* denotes that it is a number with zero based indexing which is useful for zero based selector/MUX. This uses constant multiplication and converts blocks. Also, angle within sector is given by,

$$\gamma = \theta - \left(\pi/3\right) S^* \tag{7}$$

Equation (7) utilizes a subtractor and a constant multiplication block.

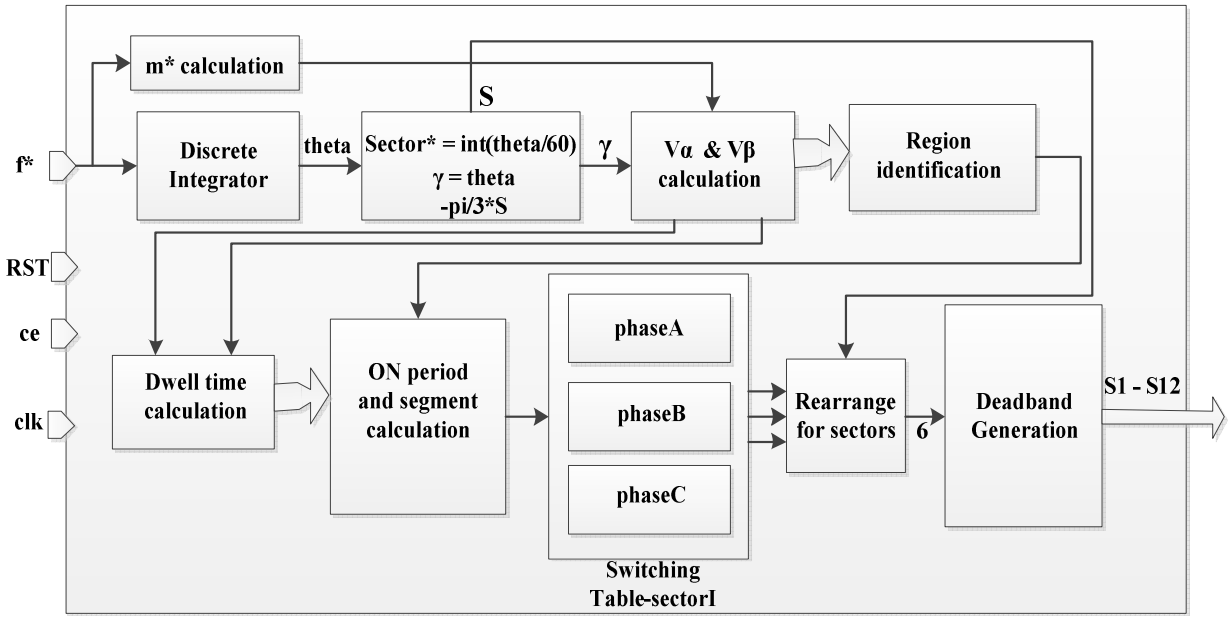


Fig. 5 Schematic Diagram for Implementation of SVPWM algorithm

C. Orthogonal Component Calculation and Region Identification

In this section, the equations used in [7] are implemented for dwell time calculation. However, region identification and component calculation are done separately using the method explained above.

D. Segment Calculation

Dwell times calculated from (1)-(3) are rearranged as per the 13-segment method shown above. These, segments are selected from the region selector and compared with single 16-bit up-down counter which yield segment index. The segment index is given by

$$I_{seg} = \text{cumsum}\left(boolean\left(Dwell < Count\right)\right) \tag{8}$$

where, *Dwell* is current dwell time which is a matrix of 1x(n-1)/2 and n is no. of segments required. *Count* is present count of the counter and *boolean* is Boolean result of their compare operation.

E. Switching Table

A single ROM based switching table is used for the switching sequence in each phase. In contrast to DSP controller or other processor which uses a whole byte/word for storing switching state, each switching state is a signed number of (n+1)/2 bits where n is level of inverter which minimizes the resources. Let us denote, switching sequence of phase A, phase B and phase C as R_1, R_2 and R_3 respectively. The relation between switching sequence of one sector is shown in Table II. Table II suggests that there is no need for storing ROM table other than Sector*-0. (Sector* has zero based indexing.)

TABLE II
SELECTION OF ROM VALUES FOR SWITCHING STATE ASSIGNMENT IN SIX SECTORS

Sector*	Phase A	Phase B	Phase C
0	R_1	R_2	R_3
1	$-R_2$	$-R_3$	$-R_1$
2	R_3	R_1	R_2
3	$-R_1$	$-R_2$	$-R_3$
4	R_2	R_3	R_1
5	$-R_3$	$-R_1$	$-R_2$

F. Deadband Generation

The equations which govern the deadband for complimentary output pairs are given as:

$$S_u = S_u \bullet \left(z^{-lat}\,\overline{S_l}\right) \tag{9}$$

$$S_l = S_l \bullet \left(z^{-lat}\,\overline{S_u}\right) \tag{10}$$

here, S_u and S_l are switching signals for upper device and lower device respectively whereas z^{-lat} is delay with latency *lat*. Although, the unit delay is simplest and popular way to implement delay with latency less than 100; for latency greater

than 100 it consumes more resources. A dual port RAM which writes the contents and later reads it after some cycles is an alternative way to implement it. Fig. shows implementation of the dead band generation scheme.

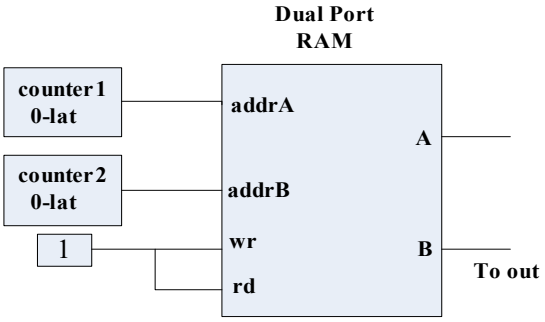
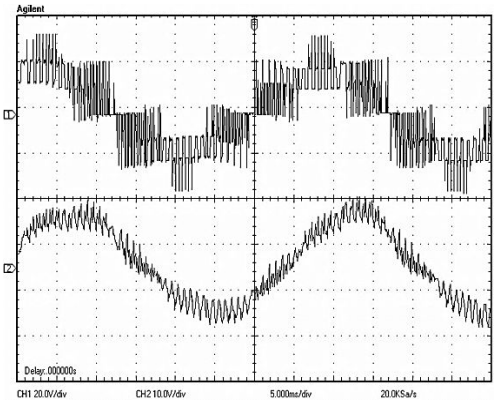


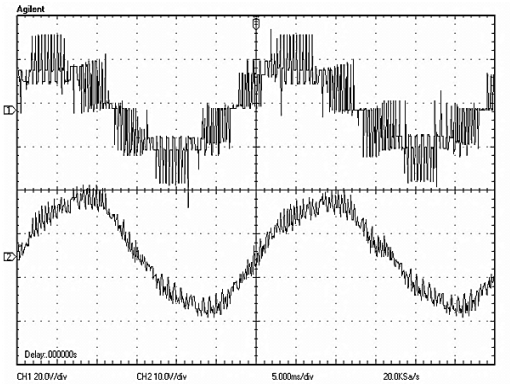
Fig. 6 Dual Port RAM based scheme for delay in Dead Band Generation

IV. Experimental Results

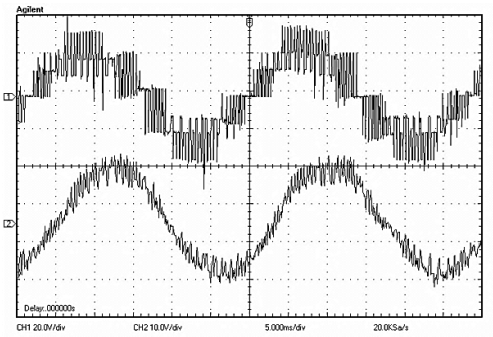
The proposed work is implemented using FPGA for Neutral Point Clamped Inverter topology with switching frequency 2 kHz. The FPGA used for implementation is xc3s5000-4fg900 operating at 40MHz clock frequency. Behavior of the said system supplying R-L load is analyzed at different frequency of operation as shown in Fig. 7 which is analogous to the said system supplying Induction motor with speed varying from stand-still to normal operating base speed. The R-L load under consideration is R=12 Ω, L=3 mH and DC link voltage (V_{dc}) is 50 V. The satisfactory behavior of the system is evident from the presented experimental results. Since orthogonal component calculation, dwell time calculations and switching state assignment is common to all number of level of inverter, the present control scheme can be used for higher level of inverter without much increase in resources utilized. The implementation is carried out using Xilinx ISE Design Suite® 10.1.03 system generator tool. The present scheme can also be utilized for the control of Cascaded H-Bridge topology.



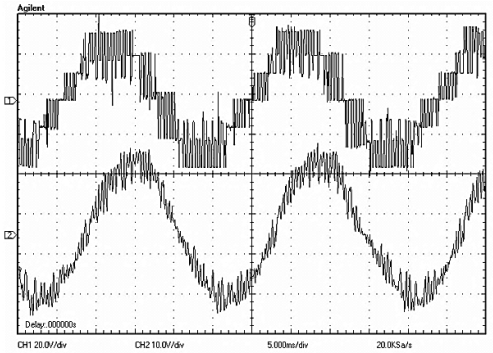
(a)



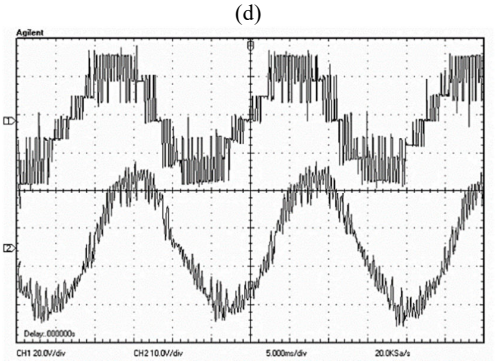
(b)



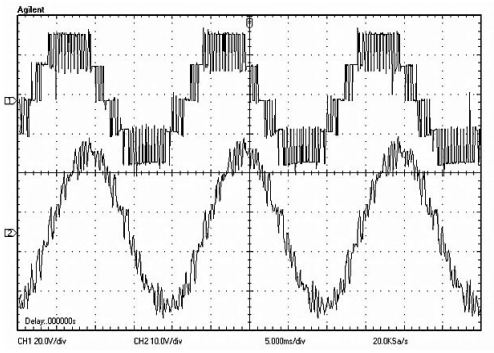
(c)



(d)



(e)



(f)

Fig. 7 Behavior of the system for different operating frequencies: V_{ab} (channel-1) and I_a (channel-2) (a) 27 Hz (b) 32 Hz (c) 35 Hz (d) 40 Hz (e) 42 Hz (f) 45 Hz [Y-axis: 50 V/div. (channel-1), 2A/div. (channel-2); X-axis: 5ms/div.]

V. Conclusion

The algorithm of SVPWM of multilevel inverter is developed with modular structure. FPGA is a good alternative for implementation of SVPWM algorithm for variable speed drives. For applications demanding higher level of inverter, the solution can reduce computation time and its incorporation is justifiable. Although, the authors have worked for open loop operation, the closed loop implementation is equally valid. There can be two alternatives for high performance closed drives,

- Interfacing FPGA with Microcontroller and feeding two orthogonal components v_α and v_β
- Development of closed loop algorithm on FPGA

The present implementation algorithm can be used for any level of inverter without increasing much resource for computation. A modular structure is used for implementation which provides flexibility to optimize the algorithm.

References

[1] Nabae, I. Takahashi, and H. Akagi, "A new neutral-point clamped PWM inverter", *IEEE Transaction .on Industrial Applications*, vol. IA-17, no. 5, pp. 518–523

[2] Wu, Bin. *High-Power Converters and ac Drives*, John Wiley & Sons (2006)

[3] Mariusz Malinowski, K. Gopakumar, Jose Rodriguez and Marcelo A. Pérez, "A survey on Cascaded Multilevel Inverters", *IEEE Transactions on Industrial Electronics*, vol. 57, no. 7, July 2010, pp 2197-2206.

[4] Bose B.K., *Modern Power Electronics and AC drives*, Prentice Hall, New Jersey, 2002.

[5] S. Wei, B. Wu, F. Li, and C. Liu, "A general space vector pwm control algorithm for multilevel inverters," in *Proc. 18th Annu. IEEE APEC*, Feb. 2003, vol. 1, pp. 562–568.

[6] N. Celanovic and D. Boroyevich, "A fast space vector modulation algorithm for multilevel three phase converters," *IEEE Transaction on Industrial Applications*, vol. 37, no. 2, pp. 637–641,Mar./Apr. 2001.

[7] A. K. Gupta, A. M. Khambadkone, "A Space Vector PWM Scheme for Multilevel Inverters based on Two level Space Vector PWM", *IEEE Transactions on Industrial Electr*onics, Vol. 53, No. 5, October- 2006 pp 1631-1639.

[8] E. F. F. Lima, N. P. Filho, J. O. P. Pinto, "FPGA Implementation of Space Vector PWM Algorithm for Multilevel Inverters Using Non-Orthogonal Moving Reference Frame" *Proceedings of IEEE International conference* IEMDS-09,pp-709-716

[9] Óscar López, Jacobo Álvarez, Jesús Doval-Gandoy, and Francisco D. Freijedo, "Multilevel Multiphase Space Vector PWM Algorithm", *IEEE Transaction on Industrial Electronics*, VOL. 55, NO. 5, MAY 2008,pp 1933-1942.

[10] O´scar Lo´pez, Jacobo A´ lvarez, Jesu´s Doval-Gandoy, Francisco Freijedo, Andre´s Nogueiras and Carlos M. Penalver, "Multilevel Multiphase Space Vector PWM Algorithm Applied to Three-Phase Converters", *Proceeding of 34th IEEE annual conference on Industrial Electronics* 2008, pp 3290-3295.