Analysis and FPGA Realization of a Pulse Width Modulator based on Voltage Space Vectors

R.Rajendran
Karpagam College of Engineering
Coimbatore
Tamil Nadu, India

Dr.N.Devarajan
Government College of Technology
Coimbatore
Tamil Nadu, India

ABSTRACT

Pulse width modulation based on space vectors (SVPWM) technique has become the most popular and important PWM techniques for three phase voltage source inverters (VSI). In this paper, we present the analysis and realization of SVPWM for variable speed control of AC motor drives, employing Xilinx Spartan 3E FPGA device. The selection of zero vectors and space vector sequence are given and the possibilities of realization on FPGA are analyzed in this paper. A voltage source inverter with the proposed SVPWM was simulated in MATLAB/SIMULINK. The simulation and experimental results proved the proposed scheme.

Keywords

Space Vector PWM, FPGA, Matlab/Simulink, VSI.

1. INTRODUCTION

One of the most preferred pulse width modulation (PWM) strategies today is space vector modulation (SVPWM). This kind of scheme in voltage source inverter (VSI) drives offers improved bus voltage utilization and less commutation losses [1-3]. Three-phase inverter voltage control by space vector modulation includes switching between the two active and zero voltage vectors so that the time interval times the voltage in the chosen sectors equals the command voltage times the time period within each switching cycle. During the switching cycle the reference voltage is assumed to be constant as the time period would be low. By simple digital calculation of the time one can easily implement the SVPWM scheme. However, the switching sequence may not be unique [4-5].

Digital Hardware Implementation of motor controller has been attractive recently because it can reduce system component ad reduce software investment [6]. With the progress of VLSI technology, the complex programmable logic device (CPLD) and field programmable gate array (FPGA) have drawn much attention due to their programmable hard-wired feature, shorter design cycle, and higher density for the implementation of digital system than other digital logic hardware[7-8]. Also it is very convenient for laboratory implementation of a project due to its unique hardware reconfigurable feature. Therefore the implementation and circuit realization FPGA for SVPWM scheme design has been reported in many papers [9-10].

The rest of the paper is organized as follows. Section 2 briefly introduces principle of space vector PWM method. Section 3 proposes FPGA Implementation of SVPWM method. Secion 4 explains simulation and experimental results and section 5 is the conclusion.

2. SPACE VECTOR PULSE WIDTH MODULATION

2.1 Principle of SVPWM

For a three-phase voltage source inverter as depicted in Fig.1, each pole voltage may assume one of the two values depending upon whether the upper switch or the lower switch is on. Therefore, only eight combinations of switches are possible; there are shown in Fig.2. Of these, two of them have zero states. Zero states occur when either the upper three or the lower three switches are conducting simultaneously.

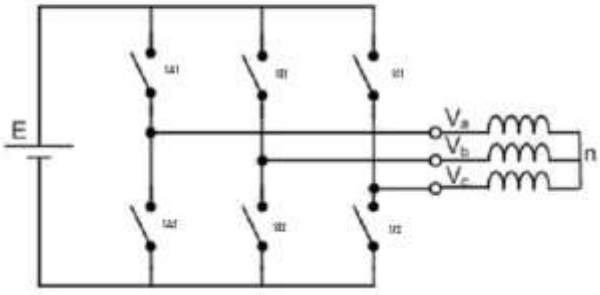


Figure 1 Six-switch voltage source inverter

The switches are termed as SA1, SA2 for pole A, SB1 and SB2 for pole B, and SC1 and SC2 for pole C. Different states are defined as follows.

A = 0 if SA1 off and SA2 on
1 if SA1 on and SA2 off
B = 0 if SB1 off and SB2 on
1 if SB1 on and SB2 off
C = 0 if SC1 off and SC2 on
1 if SC1 on and SC2 on

The instantaneous values of the line-to-line voltages of the inverter can be obtained from the above logic relations given by

$$V_{AB} = V_{dc}(A-B)$$

$$V_{BC} = V_{dc}(B-C)$$

$$V_{CA} = V_{dc}(C-A) \qquad (2.1)$$

Where V_{dc} is the DC bus voltage and V_{AB}, V_{BC}, VCA are the line-to-line voltages. The line-to-line voltages are given by

$$V_A = \frac{1}{3}(V_{AB} - V_{CA})$$

$$V_B = \frac{1}{3}(V_{BC} - V_{AB})$$

$$V_C = \frac{1}{3}(V_{CA} - V_{BC}) \qquad (2.2)$$

Replacing the values of line-to-line voltages in the previous set of equations yields the line-to-neutral voltages of the inverter.

$$V_A = \frac{Vdc}{3}(2A - B - C)$$

$$V_B = \frac{Vdc}{3}(2B - C - A)$$

$$V_C = \frac{Vdc}{3}(2C - A - B) \qquad (2.3)$$

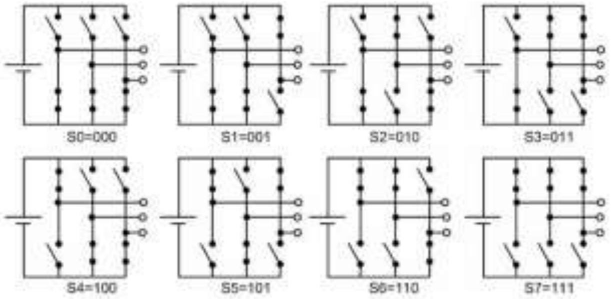


Figure 2 Possible switching pattern of the inverter

For state 4(S4), the values are

$$V_A = \frac{2}{3}V_{dc}$$

$$V_B = -\frac{1}{3}V_{dc}$$

$$V_C = -\frac{1}{3}V_{dc} \qquad (2.4)$$

By using the following transformation for a balanced three - phase system, the space vector for sector 1 with α and β components is given by

$$\begin{bmatrix} V\alpha \\ V\beta \end{bmatrix} = \frac{2}{3}\begin{bmatrix} 1 & -0.5 & -0.5 \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix} \cdot \begin{bmatrix} V_a \\ V_b \\ V_c \end{bmatrix} \qquad (2.5)$$

Therefore, for the first sector vector V_{s1} will be given by

$$\overline{Vs1} = \sqrt{2/3}\, V_{dc}e^{j\Theta} \qquad (2.6)$$

All other six nonzero states are given by

$$\overline{V_{sk}} = \sqrt{\frac{2}{3}}\; V_{dc}e^{j(k-1)60^\circ} \qquad (2.7)$$

Where the value of k varies from 1 to 6.This divides the plane into six equal regions within a regular hexagon. These vectors are of equal magnitude and mutually phase-displaced by 60^0. Fig.3 shows the realizable voltage space vectors for a three-phase VSI. Whenever the reference vector is in a sector, the switches work according to the time interval T_m and T_{m+1} set by the projection of the vector on the adjacent sides as shown in Fig.4. The [111] and the [000] states are defined as they lie on the origin. Suppose it is necessary to generate a space vector modulator for the following voltage system.

$$V_{an} = V_1\cos(\omega_m t - \gamma)$$

$$V_{bn} = V_1\cos(\omega_m t + 120 - \gamma)$$

$$V_{cn} = V_1\cos(\omega_m t - 120 - \gamma) \qquad (2.8)$$

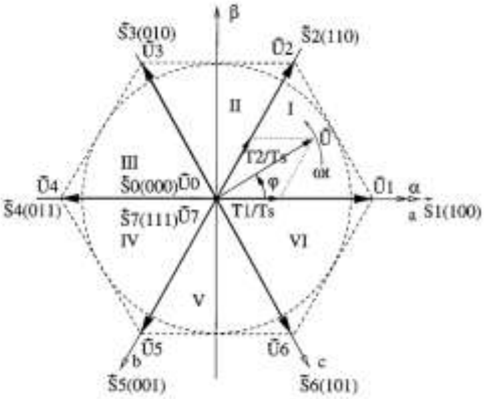


Figure 3 Possible space vectors

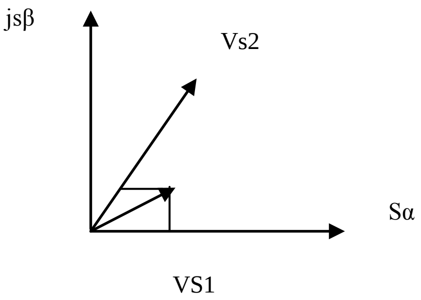


Figure 4 Time interval calculation for the space vectors

The system of voltages can be resolved into two components. These components in the stationary α and β axes are given as below:

$$V_{s\alpha} = V_1\sin(\omega_m t - \gamma)$$
$$V_{s\beta} = -V_1\cos(\omega_m t - \gamma) \qquad (2.9)$$

These two voltages can be combined as

$$\overline{Vs} = -jV_1e^{j(\omega mt - \gamma)} = V_1e^{j(\omega_m t - 90)}$$

$$= \sqrt{2/3}\, MV_{dc}e^{j(\omega_m t - 90)}$$ assuming γ to be zero.
(2.10)

Where M is the modulation index. The voltage vector refers to a circular trajectory within the hexagon with an angular frequency of ω_m. The maximum voltage that can be achieved is proportional to the radius of the largest circle inscribed within the hexagon.

2.2 Implementation of SVPWM

First the position of the rotating vector is computed by taking the arctangent of the ratio between its two different quadrature axes components. The switching time interval calculation is the tricky part for this scheme. If the voltage vector lies in the first sector, the time intervals can be expressed as depicted in Fig.5.

To synthesize a reference voltage vector $\overline{Vref}$ to its adjacent states and to obtain a minimum switching frequency for that the total cycle T_{cycle} should be divided into three segments T_m, T_{m+1}, T_0. Using simple geometry in Fig.4 yields

$$T_m + T_{m+1} + T_0 = T_{cycle}$$

$$\overline{Vref} \text{ X } T_{cycle} = T_m \text{ x } \overline{Vsm} + T_{m+1} \text{ x } \overline{Vs(m+1)} \qquad (2.11)$$

For a voltage vector residing in the first sector the equation can be expressed as

$$\overline{Vref} \text{ X } T_{cycle} = T_1 \text{ x } \overline{Vs1} + T_2 \text{ X } \overline{Vs2} \qquad (2.12)$$

By taking the components of the reference voltages in the quadrature axis,

$$\left|\overline{Vref}\right| \text{Sin}\Theta = \frac{T2}{Tcycle} \text{ x } \left|\overline{Vs2}\right| \text{Sin}60^0 \qquad (2.13)$$

The rest of the cycle can be divided between and T_0 and T_8. This can be expressed as

$$T_{cycle} - T_1 - T_2 = T_0 \qquad (2.14)$$

2.3 Switching Signals

As has been mentioned previously, the switching sequence is not unique. Of all these, the most prominent uses minimum inverter switching frequency, which is obtained by transitioning from one inverter state to another only by switching one inverter pole. The total zero time is divided between the two states.Fig.6. clearly demonstrated the switching in Sector I. Here, the cycle begins in state 0, i.e., [000], with each inverter pole being successfully toggled until state 8, [111], is obtained. The pattern is then reversed to complete the modulation cycle. Fig.6. shows the times from the start of each modulation cycle at which the inverter poles are toggled, $T_{Aon,}$ $T_{Bon,}$ T_{Con}, respectively. Taking the variations from one sector to another into consideration, it is possible to tabulate these times as functions of both the active and zero state times. It can be easily seen that from one state to the other only one inverter pole is toggled.

The other type of switching is a bus-clamped one. Under this scheme, the switching is done in one sector using two inverter poles only. One of the poles remains clamped to the higher node. This makes switching easier. This is also called a 60^0 bus-clamped scheme because the pole is clamped for 60^0. Now, one can alternate between [111] and [000] for the zero states in adjacent sectors. It is termed odd 60^0 bus-clamped switching if state 8 or [111] is used in odd sectors and vice versa.

For example, in case of a space vector in the first sector, the switching will be done according to the sequence [111], [110], [100], [110], [111]. So the upper switch of A is clamped to the positive terminal. The next sector would have a switching pattern that would like this: [000], [010], [110], [010], [000]. Here the lower switch of C will be clamped to the negative terminal and all of the inverter legs will toggle when there is a change in sector. It is worth nothing that the even sector only uses the [000] states.

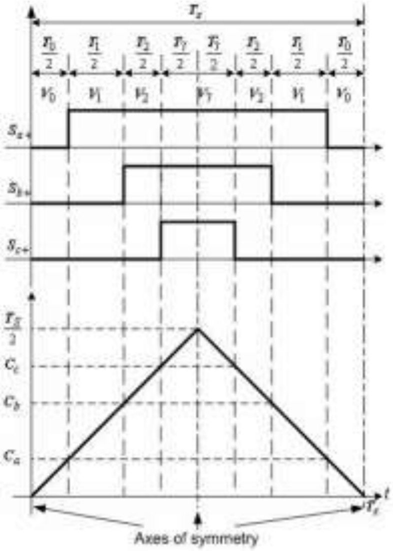


Figure 6 Switching signals for the SVPWM

3. REALIZATION OF SVPWM USING FPGA

With the advantage of high-frequency switching power devices, complex modulation schemes can no longer be realized, even using the advanced DSPs, because of the high speed switching requirement. Employing FPGA to realize PWM strategies meets the challenges of high switching frequency and reconfigurable structure for different applications [11-13]. A Field Programmable Gate Array (FPGA) is a reconfigurable digital integrated circuit that can be programmed to do any digital function. There are two main advantages of an FPGA over a microcontroller chip [14].

- FPGA has the ability to operate faster.
- FPGA supports hardware that is upwards of one million gates

FPGAs are programmed using support software and a download cable connected to a host computer. Once they are programmed, they can be disconnected from the computer and will retain their functionality until the power is removed from the chip. The FPGAs can be programmed while they run, because they can be reprogrammable in the order of milliseconds. The FPGA consists of three major configurable elements

- Configurable Logic Blocks (CLBs) arranged in an array that provides the functional elements and implements most of the logic in a FPGA.
- Input-Output Blocks (IOBs) that provide the interface between the package pins internal signal lines.

Programmable interconnects that provide routing path to connect inputs and outputs of CLBs and IOBs to the appropriate network.

3.1 FPGA Design Considerations

FPGA demonstrates good performance and logic capacity by exploiting parallelism [15]. At present single FPGA platform can play multi-functions, including control, filter and system.

FPGA design flow is a three-step process consisting of design entry, implementation, and verification stages, as shown in Fig.7.The full design flow is an iterative process of entering, implementing, and verifying the design until it is correct and complete [16-17]. The key advantage of VHDL when used for systems design is that it allows the behavior of the required system to be described (modeled) and verified (simulated) before synthesis tools translate the design into real hardware (gates and wires). HDL describes hardware behavior. There are main differences between traditional programming languages and HDL.

- Traditional languages are a sequential process whereas HDL is a parallel process.
- HDL runs forever whereas traditional programming language will only run if directed.

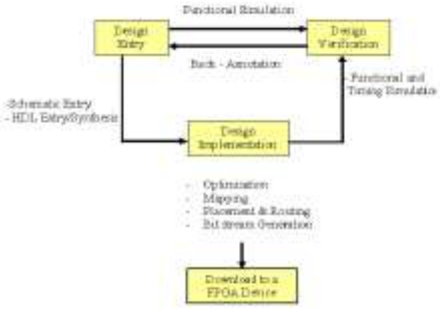


Figure 7 FPGA Design Flow

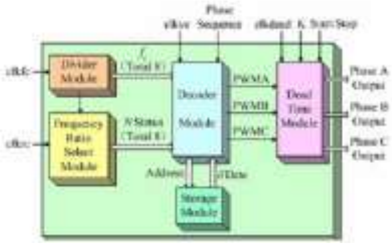


Figure 8 Functional Block Diagram

Fig.8 shows the circuit functional block diagram of the FPGA.The SVPWM scheme discussed previously is implemented on a low cost SPARTAN 3E FPGA.

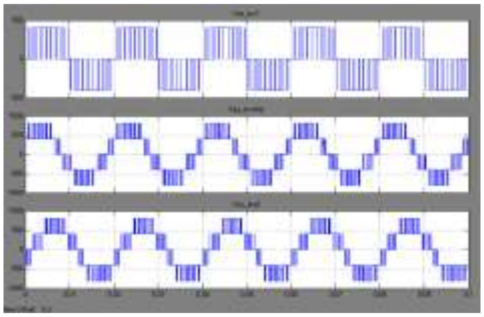
4. SIMULATION AND EXPERIMENTAL RESULTS

4.1 Simulation Results

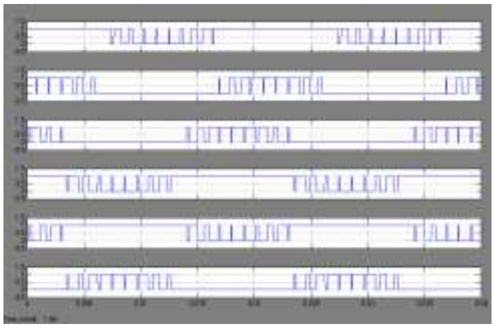
The Simulink model for Space Vector PWM has been brought in Fig.9. The Space Vector PWM output is generated from this Simulink module [18].

- Determine the position of reference vector according to fundamental frequency and time.
- According to sector wherein the reference vector is, determine the switching sequence.
- Calculate the time for different switching states.

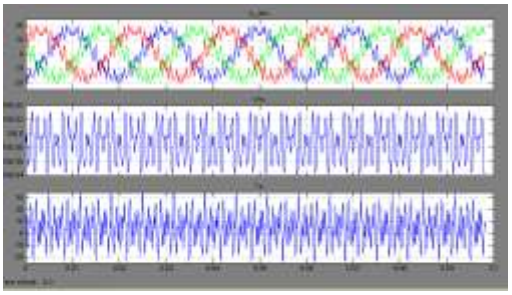
Fig.9 shows the simulation results of SVPWM.



(a) Simulation results of V_{an}, V_{ab} (Inverter) and V_{ac} (Load)



(b) Simulation results of pulse patterns

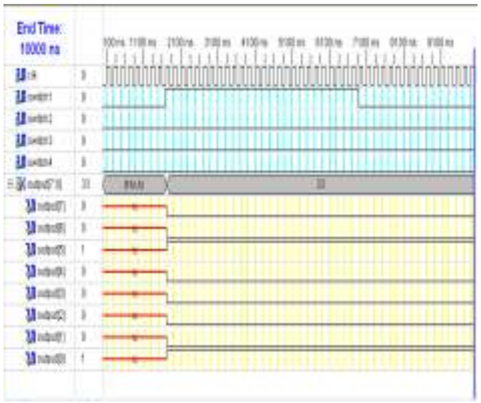


(c) Phase currents, angular speed, and electro magnetic Torque.

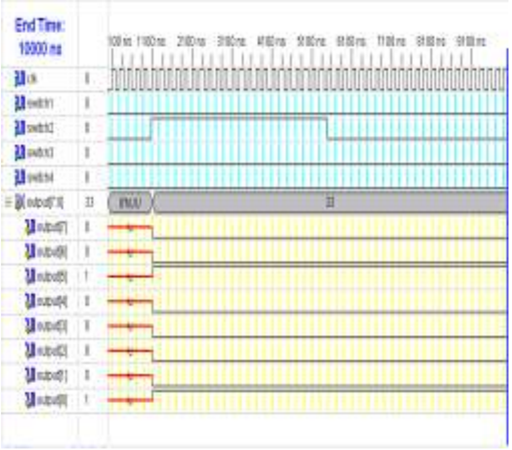
Figure 9 Simulation Results

4.2 VHDL Simulation Results

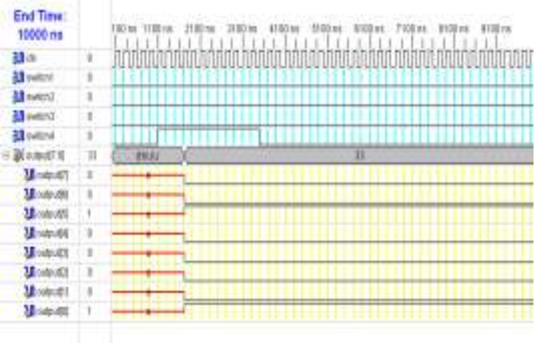
VHDL code was developed to examine the three switching patterns of SVPWM method as shown in Fig.10. This code is synthesized using Xilinx ISE®. The switching delays and the forward drop of the power switches and the dead time of the inverter are all neglected in the models. It is assumed that the model load has a fixed $R_L = 100\Omega$, L=1mH and the voltage source of the inverter is 150 V_{dc}, inverter switching frequency f_s=20000Hz.



(a)



(b)



(c)

Figure 10. Simulation Results Pulse Patterns

4.3 Experimental Setup and Results

The basic structure of experimental setup is depicted in Fig.11. This motor setup consists of induction motor and this motor is fed by the SVPWM inverter controlled directly by Spartan 3E FPGA.

In order to illustrate the efficiency of the proposed FPGA-based space vector modulator, experiments were carried out on a Spartan 3E FPGA-based prototyping platform.

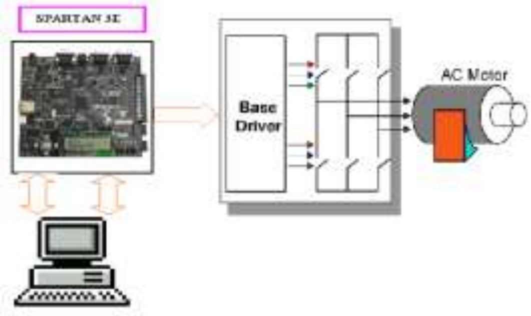
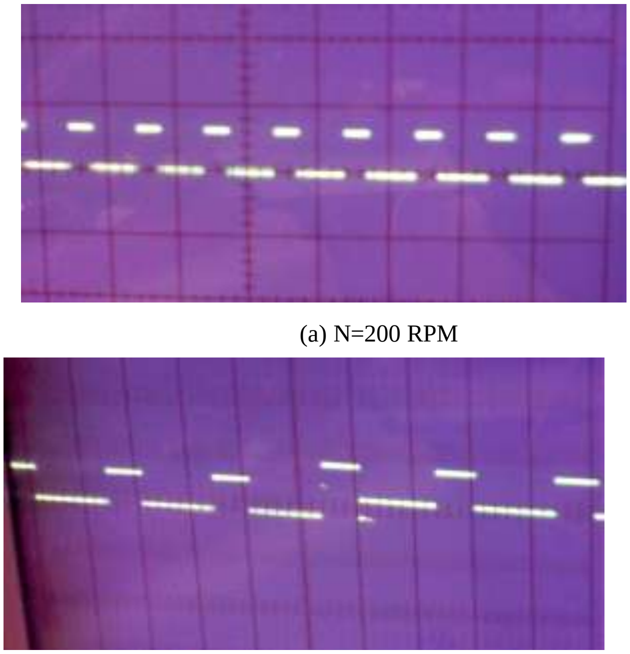


Figure 11 Structure of the experimental setup

Figure 12 Experimental setup

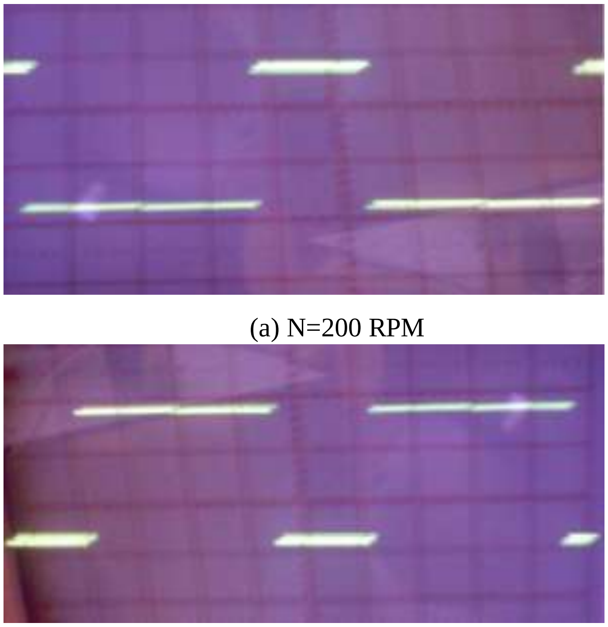
The whole execution time of the space vector modulated direct torque controller is only equal to 3.5µs as mentioned previously. As a result, the feedback control is performed quasi instantaneously and the delays in the feedback loop are very small compared to the process time scale. Thus, by implementing this controller in an FPGA, very good performance is reached.



(a) N=200 RPM

(b) N=1000 RPM

Figure 13 Pulse Pattern Generated by an FPGA



(a) N=200 RPM

(b) N=1000 RPM

Figure 14 Phase Voltages generated by SVPWM

Fig.13 shows the pulse pattern generated by an FPGA at 200 rpm and 1000 rpm for phase A and Fig.14 shows the voltage generated by SVPWM inverter at different speeds and it can be applied as input for induction motor.

5. CONCLUSIONS

We presented the analysis and realization of SVPWM for variable speed control of AC motor drives, implemented on Xilinx Spartan 3E FPGA device. The selection of zero vector and space vectors are analysed and possible implementation on FPGA is presented.

6. REFERENCES

[1] B.K.Bose, 1986, Power Electronics and AC Drives, Prentice-Hall,

[2] B.K.Bose, 1992, Modern Power Electronics: Evolution, Technology, and Applications, IEEE PRESS.

[3] J.Holtz, 1994,"Pulse Width Modulation for Electronic Power Conversion", IEEE Proceedings, Vol.82.

[4] S.Ogasawara, H.Akagi, and A.Nabae, 1989,"A novel PWM scheme of voltage source inverter based on space vector theory", European Power Electronics Conference Record, Germany, Pp.1197-1202.

[5] H.Van Der Broeck, H.Skudelny and G.Stanke, 1986,"Analysis and Implementation of a pulse width modulator based on voltage space vectors", IEEE-IAS Conference Records, pp.244-251.

[6] S.J.Henriksen, et al, 1999,"Digital hardware implementation of a current controller for M variable speed drive", IEEE Transaction on Industrial Applications, vol.35.

[7] S.L.Jung, M.Y.Chang, and Y.Y. Tzou, 1997,"FPGA-based control IC for a single phase PWM inverter used for UPS", IEEE PEDS Conf. Rec., pp.344-349, May 26-29.

[8] S.L.Jung, et al, May 1999,"Design and Implementation of an FPGA-based control IC for AC-voltage regulation", IEEE Transaction on Power Electronics, vol.14, No.3.

[9] Y.Y.Tzou, H.J.Hsu, Nov.1997"FPGA based SVPWM control IC for PWM inverters", IEEE Transaction on Power Electronics, vol.12, No.6, pp.953-963.

[10] A.V.Peterchev, S.R.Sanders, Jan.2003,"Architecture and IC Implementation of a Digital VRM Controller", IEEE Transaction on Power Electronics, vol.18.

[11] W.P.Hew, et al, Dec.2005,"Realization of space vector modulation technique in a single FPGA chip for induction motor drive PWM," IEEE International Conference on Electron Devices and Solid State Circuits, Hong Kong, pp.817-820.

[12] Z.Zhou, et al, "Realization of an FPGA- based space vector PWM controller," IEEE International Conference on Power Electronics and Motion Control (IPEMC'2006).

[13] Bob Zeidman and Robert Zeidman, 2002, Design with FPGA and CPLD, CMP Books, Bekerly.

[14] Xilinx Application Notes

[15] C.Dick, Spring 1999"A system level design approach for FPGA based DSP implementations", in DSP World.

[16] S.Sjoholm, L.Lindth, 1997,"VHDL for Designers", Prentice Hall, pp.30-162.

[17] A.Rushton, 1998"VHDL for logic synthesis", John Wiley & Sons Ltd., pp.169-226.

[18] www.mathworks.com.