FPGA IMPLEMENTATION OF SAMPLED SPACE VECTOR PULSE WIDTH MODULATION TECHNIQUE FOR TWO LEVEL INVERTER

S. NAGESWARI[1] Dr.V.SURESH KUMAR[2]

[1]Department of Electrical & Electronics Engg., A.C College of Engineering and Technology, Karaikudi, Tamilnadu, India. E- mail : mahabashyam@rediffmail.com

[2]Department of Electrical & Electronics Engineering, Thiagarajar College of Engineering , Madurai, Tamilnadu, India. E-mail: vskeee@tce.edu

Abstract *– This paper proposes a simple and fully digitized hardware design scheme of a space vector pulse width modulation (SVPWM) which is verified and implemented on a single chip field programmable gate array (FPGA). The advantage of this design is that, it determines the commutation pattern without computing the angles of each sextant but using the samples of modulating signal. The algorithm can be implemented without digital signal processor. The switching times are calculated "off-line" and they are stored in memory. The algorithm is implemented using single FPGA chip. It involves the design, VHDL coding, functional and timing simulation, and synthesis using xilinx spartan-3E. Experimental results obtained from a three phase two level inverter prototype are presented to validate the theoretical observations.*

Keywords – *SVPWM, FPGA, VHDL, inverter*

1. Introduction

PWM inverters are becoming more and more popular in today's motor drives. PWM inverters make it possible to control both the frequency and magnitude of the voltage and current applied to a motor. As a result, PWM inverter powered motor drives offer better efficiency and higher performance compared to fixed frequency motor drives. PWM techniques have been the subject of intensive research during the last few decades. A large variety of methods, different in concept and performance have been compared based on different merit factors such as; total harmonic distortion (THD) of the output voltage or current at the inverters, switching losses of the inverters, maximum inverter output voltage as a function of a given DC bus voltage[1].

Sinusoidal pulse width modulation (SPWM) is used to control the inverter output voltage and maintains good performance of the drive in the entire range of operation between zero and 78% of the value that would be reached by square operation. If the modulation index exceed this value, linear relationship between modulation index and output voltage is not maintained and the over modulation methods are required. And in recent years, an inherently digital modulation technique known as SVPWM which is based on space vector theory, has been developed [2]. It has been reported that SVPWM technique offers superior performance over other techniques in terms of lower THD, switching loss, torque ripple, better DC link utilization and easier to implement in digital system[3]. Space vector modulation techniques have been increased by using in last decade, because they allow reducing commutation losses and the harmonic content of output voltage, and to obtain higher amplitude modulation indexes if compared with conventional SPWM techniques [4]. Due to increasing reliability and performance of microprocessors, digital control techniques have been dominating over their analog counterparts in the last years [5]. Most space vector modulation schemes are carried out using analog circuits or digital circuits, such as DSPs and microcontrollers [6, 7]. A more efficient and faster solution is the use of Field Programmable Gate Arrays (FPGA). Since FPGA can carry out parallel processing by means of hardware mode which occupies nothing of CPU, the system can get a very high speed level [8]. With the FPGA's field-programmable capability, the flexible adjustment of dead time and switching frequency makes it suitable to drive various switching devices in practical applications [9].

The space vector pulse width modulation technique involves complex calculations. In earlier papers [10][11] digital signal processor was used to perform the arithmetic calculations. In this case, minimum two processors are needed to implement the technique. Where as in the proposed work, all the calculations are done off-line and stored in memory. Even though memory space is large compared to previous methods, it takes minimum accessing time. It does not affect the speed of operation. Memory is also programmed in FPGA itself and so no need of external memory. Furthermore, there are many advantages due to the rapid design process and reprogrammable functions for FPGA. FPGA enables to produce prototype logic designs right in a short period. It is possible to create, implement, and verify a new design.

2. Space vector PWM

The basic power circuit topology of a 3Ø voltage source inverter supplying a star connected three-phase load is given in Figure 1. The power circuit contains in general six semiconductor switches such as MOSFETs, IGBTs etc with anti-parallel diodes for protection.

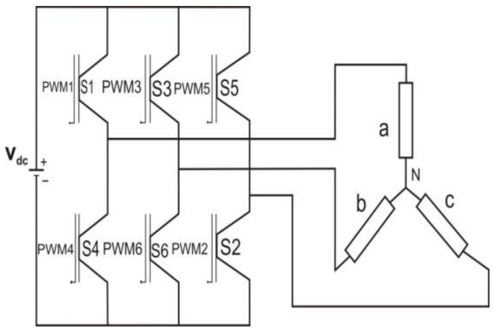


Fig. 1. Power circuit topology of a three phase VSI.

The two power switch of one leg is complimentary in operation with a small dead band between the switching of two devices. Switching operation of the inverter yield in total eight output vectors with six being active or non zero and two zero vectors. The six active vectors are labeled as V_1, V_2, V_3, V_4, V_5,V_6 and the two zero vectors are labeled as V_0, and V_7.

If these eight voltage vectors are converted to two axes, it can be plotted as shown in Figure 2. The tips of the six non zero vectors, when cornered, form a regular hexagon with the two zero vectors lying at the origin.

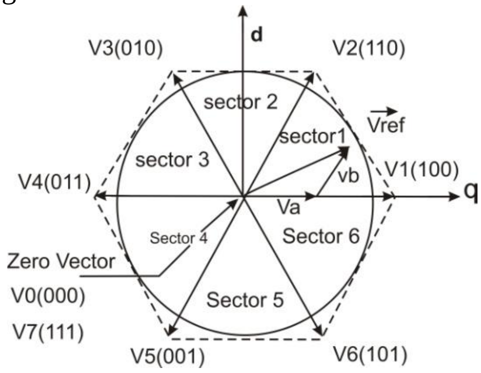


Fig. 2. Voltage vectors in (d-q) plane.

Space vector theory represents the three phase voltages as voltage space vector rotates in (d-q) plane. The magnitude of the vector is related to the magnitude of the phase voltages and the time that the vector take to complete one revolution is the period time of fundamental phase voltages. The inverter power switches are driven to obtain eight voltage vectors according to eight switching positions. These voltage vectors represent six active vectors (V_1-V_6) that subdivide the plane into six equal sectors, and two non-active vectors (V_0,V_7), sometime call zero vector at origin. The desired three-phase sinusoidal output voltages correspond to a circle in the (d-q) plane as shown in Figure 2. The output line and phase voltages for each space vector are shown in Table 1. Two phase components can be calculated by using the equation.1.

$$\begin{bmatrix} V_d \\ V_q \end{bmatrix} = \frac{2}{3}\begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix}\begin{bmatrix} V_{an} \\ V_{bn} \\ V_{cn} \end{bmatrix} \tag{1}$$

Table 1 Output voltages of three phase inverter

Voltage vectors	Switching vectors			Line to neutral voltage			Line to Line Voltage		
V_0	0	0	0	0	0	0	0	0	0
V_1	1	0	0	2/3	-1/3	-1/3	1	0	-1
V_2	1	1	0	1/3	1/3	-2/3	0	1	-1
V_3	0	1	0	-1/3	2/3	-1/3	-1	1	0
V_4	0	1	1	-2/3	1/3	1/3	-1	0	1
V_5	0	0	1	-1/3	-1/3	2/3	0	-1	1
V_6	1	0	1	1/3	-2/3	1/3	1	-1	0
V_7	1	1	1	0	0	0	0	0	0

In each sampling period, the required reference voltage vector V_{ref}, is approximated by an average time of three main voltage vectors. There are two active vectors adjacent to each sector and a zero vector. The duty ratios are determined in such way that the realized output voltage vector, when averaged over one switching cycle, coincides with the reference voltage vector, V_{ref}. So the reference voltage vector is given in equation.2.

$$V_{ref} = \overrightarrow{V_a}\frac{t_a}{T_s} + \overrightarrow{V_b}\frac{t_b}{T_s} + \overrightarrow{V_0}\frac{t_0}{T_s} \tag{2}$$

Where, t_a, t_b and t_o are the respective on-time to generate voltage vectors V_a , V_b and V_0 and T_s is the sampling time.

2.1. To determine T_a, T_b & T_0 (0.5< m < 0.906)

The dwell times t_a, t_b,t_0 are calculated using equation.3 to equation.5.

$$t_a = \frac{\sqrt{3}}{\pi} mT_s \left[\sin\frac{k\pi}{3}\cos\theta - \cos\frac{k\pi}{3}\sin\theta\right] \tag{3}$$

$$t_b = \frac{\sqrt{3}}{\pi} mT_s \left[\cos\frac{(k-1)\pi}{3}\sin\theta - \sin\frac{(k-1)\pi}{3}\sin\theta\right] \tag{4}$$

$$t_0 = T_s - (t_a + t_b) \tag{5}$$

Where, m is the modulation index and θ is the angle of reference vector in each sector.

2.2. To determine the line and phase voltage

The line and phase voltage of the inverter can be found equation.6 and equation.7.

$$\begin{bmatrix} V_{ab} \\ V_{bc} \\ V_{ca} \end{bmatrix} = V_{dc} \begin{bmatrix} 1 & -1 & 0 \\ 0 & 1 & -1 \\ -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} \qquad (6)$$

$$\begin{bmatrix} V_{a} \\ V_{b} \\ V_{c} \end{bmatrix} = \frac{1}{3} V_{dc} \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} \qquad (7)$$

3. Xilinx FPGA design flow

FPGA design methodologies include VHDL or Verilog languages, synthesis, map, and place and route scheme. The considerations are pin-pin delays, pipelining and logic levels, and floor planning, etc. FPGAs are now posses' sufficient performance and logic capacity to implement a number of digital signal processing algorithms effectively. This is accomplished by exploiting parallelism as well as mapping techniques such as distributed arithmetic. At present, a single FPGA platform can replace for multiple applications, including controllers, filters, and interfaces. High performance FPGA circuits are often achieved using design techniques generally unfamiliar to the DSP design community, which traditionally relies on the processor hardware together with smart compiler technology. The block diagram for FPGA design flow is as shown in Figure 3. The FPGA design flow is a three-step process that consists of the following stages.

(i) Design entry - In this stage of the design flow, a hardware description language (HDL) for text-based entry, a schematic editor, or both is used to create the design.

(ii) Design implementation - By implementing to a specific Xilinx architecture, the logic design file format, such as EDIF, that is created in the design entry stage is converted into a physical file format. The physical information is contained in the native circuit description (NCD) file for FPGAs. Then a bit stream file is created.

(iii) Design verification - Using a gate level simulator, the Xilinx XChecker cable, or JTAG cable, to ensure that the design meets the timing requirements and functions properly.

4. FPGA realization of SVPWM

To realize space vector modulation in a FPGA chip, following steps are to be carried out

Step-1. Creating a look-up Table of sinθ & cosθ.

Step-2. Sector identification.

Step-3. Determine time duration T_a, T_b, T_0.

Step-4. Determine the switching time of each switch (S1 to S6).

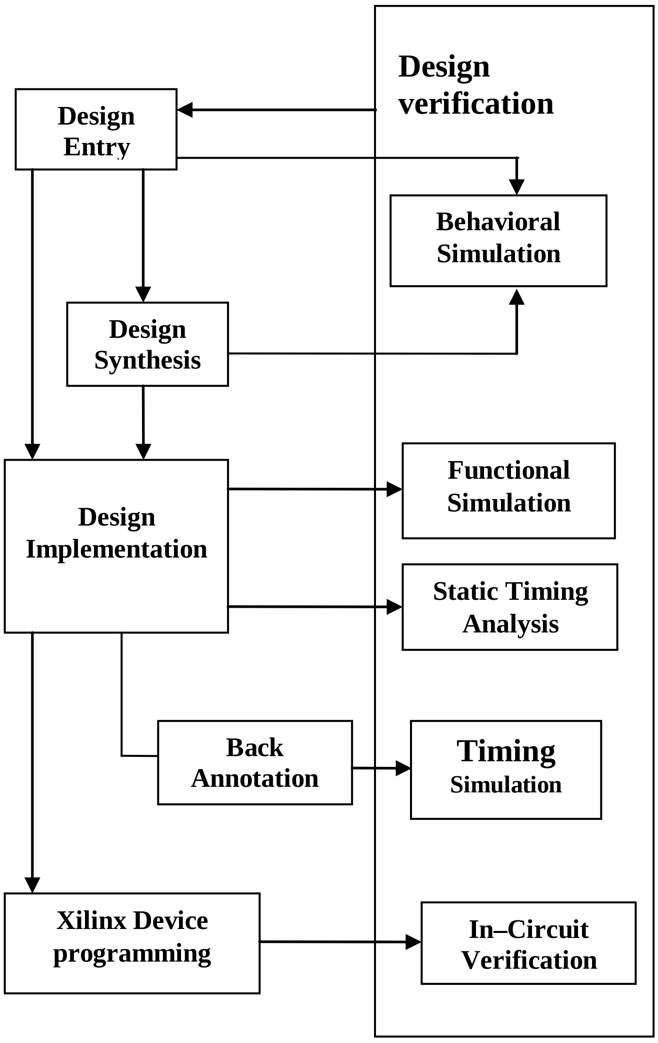


Fig. 3. Xilinx FPGA design flow block diagram.

4.1. RTL schematic of Space Vector Modulation

RTL schematic comprises of pulse generation, frequency measurement, PI controller, transmit/receive and LCD display block. It clearly describes the logic of the very high speed integrated circuit hardware description language (VHDL) coding. It also gives idea about input and output parameters.

4.2. PWM Generation

The system block diagram of pulse generation logic is shown in Figure 4. The sine and the cosine values are stored in the look up Table. As the resolution chosen is 8 bit, 256 sine and cosine values are stored in an array. Next to this block is sector identification block and this block is used to find the sector at which the reference vector presents. The number of counts required for one complete cycle will be 200 (for 50 Hz).

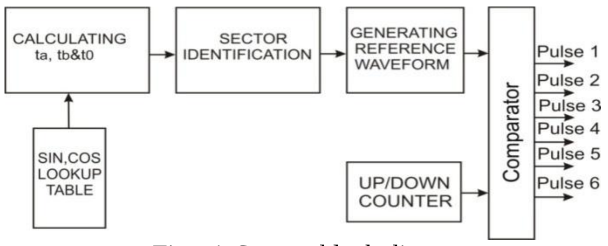


Fig. 4. System block diagram.

Table 2 Switching time of each transistor (S_1 to S_6)

Sectors	Upper Switches	Lower Switches
1	$S_1 = t_a + t_b + \frac{t_0}{2}$ $S_3 = t_b + \frac{t_0}{2}$ $S_5 = \frac{t_0}{2}$	$S_4 = \frac{t_0}{2}$ $S_6 = t_a + \frac{t_0}{2}$ $S_2 = t_a + t_b + \frac{t_0}{2}$
2	$S_1 = t_a + \frac{t_0}{2}$ $S_3 = t_a + t_b + \frac{t_0}{2}$ $S_5 = \frac{t_0}{2}$	$S_4 = t_b + \frac{t_0}{2}$ $S_6 = \frac{t_0}{2}$ $S_2 = t_a + t_b + \frac{t_0}{2}$
3	$S_1 = \frac{t_0}{2}$ $S_3 = t_a + t_b + \frac{t_0}{2}$ $S_5 = t_b + \frac{t_0}{2}$	$S_4 = t_a + t_b + \frac{t_0}{2}$ $S_6 = \frac{t_0}{2}$ $S_2 = t_a + \frac{t_0}{2}$
4	$S_1 = \frac{t_0}{2}$ $S_3 = t_a + \frac{t_0}{2}$ $S_5 = t_a + t_b + \frac{t_0}{2}$	$S_4 = t_a + t_b + \frac{t_0}{2}$ $S_6 = t_b + \frac{t_0}{2}$ $S_2 = \frac{t_0}{2}$
5	$S_1 = t_b + \frac{t_0}{2}$ $S_3 = \frac{t_0}{2}$ $S_5 = t_a + t_b + \frac{t_0}{2}$	$S_4 = t_a + \frac{t_0}{2}$ $S_6 = t_a + t_b + \frac{t_0}{2}$ $S_2 = \frac{t_0}{2}$
6	$S_1 = t_a + t_b + \frac{t_0}{2}$ $S_3 = \frac{t_0}{2}$ $S_3 = t_a + \frac{t_0}{2}$	$S_4 = \frac{t_0}{2}$ $S_6 = t_a + t_b + \frac{t_0}{2}$ $S_2 = t_b + \frac{t_0}{2}$

Therefore to complete one sector, it requires 34 counts. By the count number, the location of reference vector in the sector can be predicted. Once the count reaches 34, the sector will be incremented by one. For each sector, switching time will be varied as shown in Table 2. Along with the rollover of integrator (i.e.) the upper 8 bits of integrator points to the starting point of the lookup table. Similarly it takes place for the other sectors also. Followed by this, the block is the reference waveform generation block. In this block the reference waveform is generated. The corresponding duty ratio values taken from the look-up table are compared with the up-down counter values and the PWM pulses (pulse1-pulse6) are generated. These steps are explained in the program flow as shown in Figure 5.

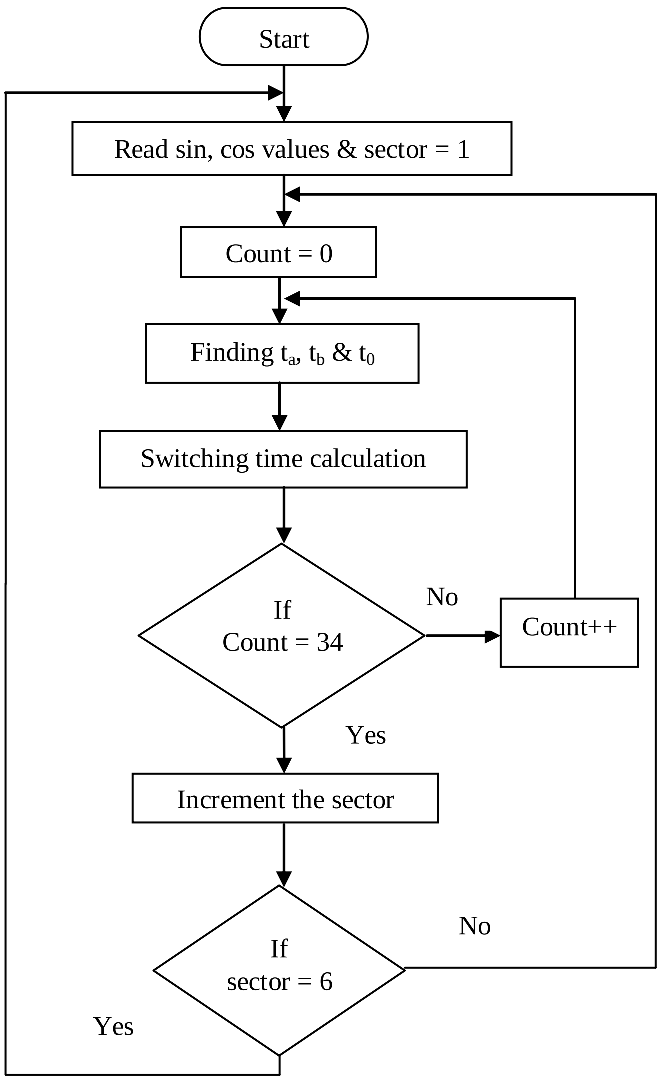

Start
Read sin, cos values & sector = 1
Count = 0
Finding t_a, t_b & t_0
Switching time calculation
If Count = 34
No
Count++
Yes
Increment the sector
If sector = 6
No
Yes

Fig. 5. Program Flow.

4.3. Measurement of Frequency Step

The RTL schematic of frequency step is shown in Figure 6. The clock (clk), and speed (Rpm) are its inputs and there by calculating the frequency and frequency step at its outputs. Step size can be calculated using equation.8.

$$STEP = \frac{f_r * 6 * 2^r}{f_s} \quad (8)$$

-Where r is the number of bits in the register. Here r =16. STEP is the step size, f_r is the frequency step and f_s is the line voltage frequency.

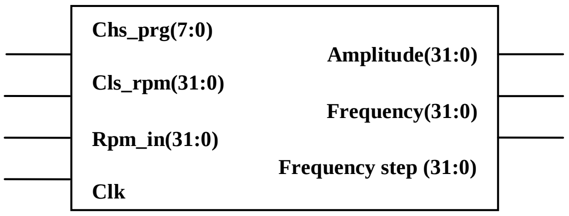


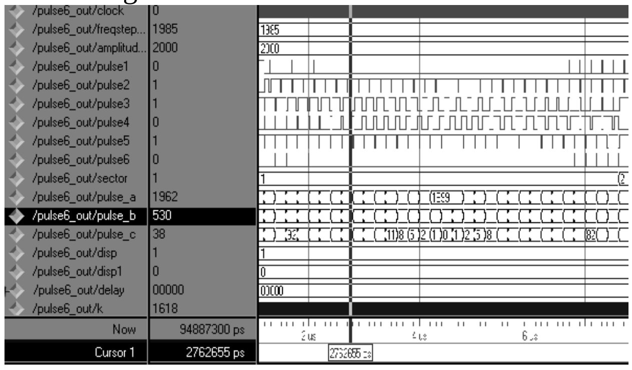
Fig. 6. RTL schematic of frequency step.

The generation of the sine wave with variable frequency is performed by using a look up table. Since in space vector modulation each sector is 0° to 60°, 0° to 59° will be sufficient to calculate d-q value. The 16 bit counter is used to determine the location of next value of the look up table. The frequency is varied by varying step size.

A 16 bit counter divides input clock frequency by 2 times. The upper byte of the 16 bit counter is used to address the look up table, where sine values of 0°-59° degree are stored. 256 points can be taken if no step value is added to the 16 bit counter. If the step size is 2, then the time taken to cycle the 256 points will be half of the previous one of the step size 1. If the step size is 4, then the 256 points will be cycled in the 1/4 time of step size 1. Hence the frequency is varied by changing the step size.

5. Simulation results

The simulation results of the PWM generator for the first sector are shown in Figure 7. The gate pulses are generated from the PWM generator. The simulation is carried out using ModelSim software**.** The test bench applies input stimuli signals during simulation, the simulator is created to represent reality; it verifies the functionality and performance of the design.



/pulse6_out/clock	0
/pulse6_out/freqstep...	1985
/pulse6_out/amplitud...	2000
/pulse6_out/pulse1	0
/pulse6_out/pulse2	1
/pulse6_out/pulse3	1
/pulse6_out/pulse4	0
/pulse6_out/pulse5	1
/pulse6_out/pulse6	0
/pulse6_out/sector	1
/pulse6_out/pulse_a	1962
/pulse6_out/pulse_b	530
/pulse6_out/pulse_c	38
/pulse6_out/disp	1
/pulse6_out/disp1	0
/pulse6_out/delay	00000
/pulse6_out/k	1618
Now	94887300 ps
Cursor 1	2762655 ps

Fig. 7. ModelSim Timing Waveform.

6. Experimental results

Figure 8 shows the experimental setup which consists of a 3Φ two level inverter power circuit and FPGA control circuit. The switches used in the power circuit are IGBT – CT60AM. The logic for generating different voltage levels is implemented in Spartan-3E FPGA chip which is composed of 250K logic. The PWM signals with the fundamental frequency of 50 Hz are produced. To visualize the actual output of the inverter block, filtering of the PWM ripple is required. The PWM voltage signal is filtered here using first order filter. Pole voltage waveforms for the three phases are obtained by filtering the PWM using a low-pass filter and are shown in Figure 9. The parameters of the system used in the hardware setup are shown in Table 3. The gating signals generated from the FPGA are given to IGBT switches in the inverter circuit. Figure 10 shows the phase voltage waveform across the load. Experimental results show the constructed SVPWM IC can generate the required gating signals for the switches in the inverter.

Table 3 parameters of the system used in the hardware setup

Inverter	3Φ two level VSI
Switching device	IGBT
Input voltage	Variable DC
Load	3Φ lamp load
Modulation index	0.906
Dead time	50μs
Filter	R=1500Ω, C=100nF

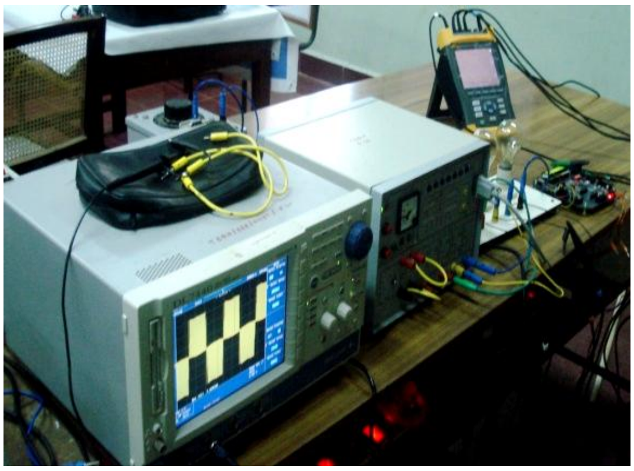


Fig. 8. Hardware Setup.

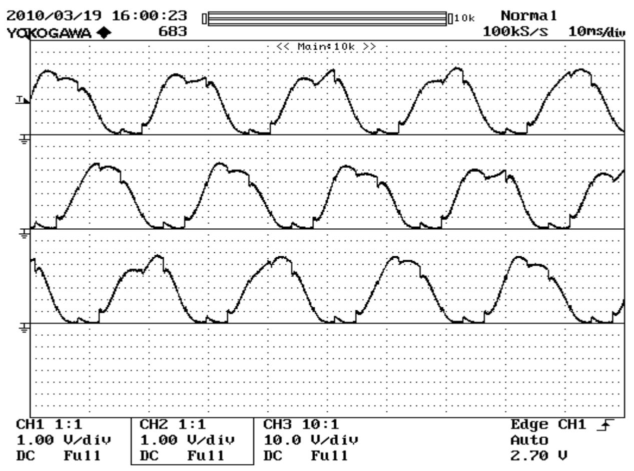


Fig. 9. Filtered PWM Signals.

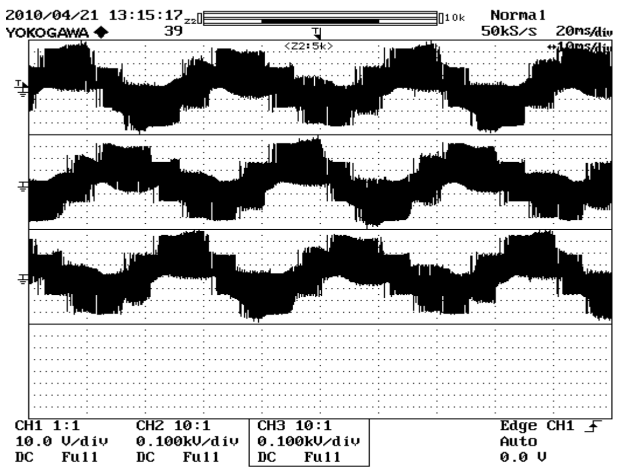


Fig. 10. Phase Voltage Waveforms.

7. Conclusion

This paper presents a simple algorithm for FPGA realization of space vector modulation technique for 3Φ two level inverters. The advantage of this design is that, it determines the commutation pattern without computing the angles of each sextant and the computational load can be increased without increasing the cost of the hardware. This method eliminates the need of DSP and so the SVPWM IC can be incorporated without DSP. Simulation results are achieved using ModelSim software. Experimental results have validated the effectiveness of the approach.

References

1. Van der Broeck, Skudelny, HC., and Stanke GV.: *Analysis and realisation of a pulsewidth modulator based on voltage space vectors*. In: IEEETransactions on Industrial Applications, April 1988, Vol.24, No.1, p. 142–150.
2. Ying-Yu Tzou and Hau-Jean Hsu : *FPGA realization of space-vector PWM control IC for three-phase PWM inverters*. In: IEEE Transactions on Power Electronics, Vol.12, No.6, 1997, pp. 953 – 963.
3. Mauricio Tonelli, Pedro Battaiotto and Maria I. Valla (2001), *FPGA Implementation of an universal space vector modulator*. In: Proceedings of the IEEE IECON'01 , 2001, p.1172-1177.
4. Woei-Luen, Chen, Chun-Hao Pien and Yung-Ping Feng (2008), *Design of an FPGA-based space vector PWM Generator for three-phase voltage-sourced Inverters*. In: Proceedings of the conference PECon'08, 2008, p.584-588.
5. Zhou Yuan, Xu Fei-peng, Zhou and Zhao-yong (2006), *Realization of an FPGA-based space-vector PWM Controller*. In: Proceedings of the Power Electronics and Motion Control Conference, 2006, p.1-5.
6. Wajiha Shireen, Mohammed S.Arefeen and David Figoli (2003), *Controlling multiple motors utilizing a single DSP controller*. In: IEEE Transactions on Power Electronics, Jan 2003, Vol.18, No.1, p. 124 –130.
7. Celanovic, N, and Boroyevich, D (2001), *A fast space-vector modulation algorithm for multilevel three-phase converters*. In: IEEE Transactions on Industry Applications, Vol.37, No. 2, Mar.-Apr. 2001, p.637 - 641.
8. Hew, WP, Ooi, CP and Rahim, NA (2005), Realization *of space vector modulation technique in a single FPGA chip for induction motor drive*. In: Proceedings of the IEEE Conference on Electron Devices and Solid-State Circuits, 2005, p. 817 – 820.
9. Victor M. Mora, Ciro A. Nunez, Victor M. Cairdenas and Homero Miranda (2006), *Simple and practical FPGA implementation of space vector modulation based on geometrical considerations*. In: Proceedings of the IEEE International Power Electronics Congress CIEP 2006, p.1-6.
10. Jiang, S, Liang, J, Liu, Y, Yamazaki, K and Fujishima, M (2005), *Modeling and co simulation of FPGA-based SVPWM control for PMSM*. In: Proceedings of the IEEE International Conference IECON 2005, p.1538-1543.
11. Zhaoyong Zhou, Tiecai Li, Takahashi.T and Ho.E (2004), *Design of a universal space vector PWM controller based on FPGA*. In: Proceedings of the International Conference APEC'04 , 2004, p.1698 – 1702.