International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Impact Factor (2012): 3.358

Implementation of Space Vector Pulse Width Modulation using Arduino

Asma[1], Naik R. L.[2] and Jangamshetti Suresh H.[3]

[1]Student Member IEEE, Department of Electrical and Electronics Engineering, Basaveshwar Engineering College, Bagalkot

[2]Member IEEE Department of Electrical and Electronics Engineering, Basaveshwar Engineering College, Bagalkot

[3]Senior Member IEEE Department of Electrical and Electronics Engineering, Basaveshwar Engineering College, Bagalkot

Abstract: ***Space Vector Pulse Width Modulation (SVPWM) method is an advanced computational intensive PWM algorithm for voltage source converter. Among various PWM techniques, SVPWM is most preferred technique because of higher DC bus utilization and ease in implementation. Traditionally processors like Microprocessors, Microcontrollers, DSP and dsPICs are used to implement SVPWM. However these processors are having limitations like moderate processing, complex implementation and higher switching losses. To overcome these aspects ATmega microcontroller is proposed for implementing SVPWM. The paper presents a review of different processors like DSP, dsPIC and ATmega further implementation of SVPWM using ATmega328P has been presented. The results shows that the space vector PWM waveforms generated are symmetric with respect to the middle of each PWM period. It has been concluded that ATmega microcontroller is the suitable processor for implementing SVPWM technique.***

Keywords: Arduino, ATmega, DSP, dsPIC, SPWM, SVPWM.

1. Introduction

In recent years, the SVPWM is widely used for three-phase PWM inverters since it gives higher output voltage and lower harmonic distortion than the conventional Sinusoidal Pulse Width Modulation (SPWM) method [8]. SVPWM is an advanced real time control mechanism that can be used to generate balanced three phase ac voltages of the desired magnitude and frequency at the output of an inverter. To implement space vector modulation a reference signal V_{ref} is sampled with a frequency f_s ($T_s = 1/f_s$). The reference vector is then synthesized using a combination of the two adjacent active switching vectors and one or both of the zero vectors. SVPWM technique can be implemented using different processors like DSP [1], dsPIC [2] and ATmega [4]. It is observed from the literature survey that implementation of SVPWM in DSP using look up table, increases the execution time and overload the processor during closed loop operation [1]. Further DSPs do not have built-in PWM units to control switches of inverter. These features increase circuit complication and computational burden [5]. While implementing SVPWM in dsPIC, unconventional mnemonic names are present, inefficient lookup tables, no add/subtract with carry, small stack, paged architecture becomes a real pain. The PIC18 architecture addresses many of the above said issues, however implementation of SVPWM at higher switching frequency adds extra limitation in program environment [6]. SVPWM program developed in DSP and dsPIC has to be executed in Code Composer and MPLAB respectively. To execute the program, library referencing has to be made for the created project. This library referencing is processor dependent, which creates in-convenience to program developers.

To overcome the above said aspects ATmega microcontroller is proposed for implementing SVPWM. ATmega is an 8-bit CPU and on the same clock it is 4 times faster than 8-bit PIC and 12 times faster than 8051. It has a GCC based IDE that is free for the whole range of their processors. Power consumption is much lower as compared to PIC. UART (TX and RX) can be enabled separately which is useful for multi-processor on a common bus.

Hence this paper presents the review of different processors like DSP, dsPIC and ATmega along with implementation of SVPWM using ATmega328P. Section II presents the basic principle of SVPWM, section III presents brief description regarding DSP and dsPIC processors and also algorithm for implementing SVPWM using ATmega328P. The results shows that the space vector PWM waveforms generated are symmetric with respect to the middle of each PWM period. Symmetric waveform generation will eliminate even harmonics and reduces the odd harmonics. It has been concluded that to overcome the limitations of DSP and dsPIC, ATmega is the suitable processor for implementing SVPWM technique used for the voltage source converter.

2. Principle of SVPWM

Space vector PWM refers to a switching scheme of the six power switches of a 3-phase VSI. It generates minimum harmonic distortion and also provides more efficient use of DC supply voltage in comparison with the sinusoidal modulation method. The topology of conventional two level voltage source inverter is shown in Fig.1

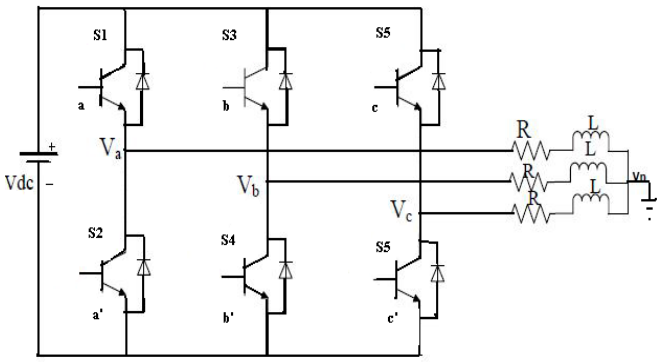

Figure 1: Structure of typical 3-phase VSI

Modeling of two level voltage source inverter in terms of switching function is given by (1)

$$\begin{bmatrix} V_a \\ V_b \\ V_c \end{bmatrix} = \frac{V_{dc}}{3} \begin{bmatrix} 2 & -1 & 2 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix} \begin{bmatrix} S_1 \\ S_2 \\ S_3 \end{bmatrix} \quad (1)$$

Where S_1, S_3, S_5 are the switching functions.
The space phasor can be defined by using two voltages V_α and V_β and are given by (2) & (3), [1]

$$V_\alpha = \frac{3}{2} * V_{an} \quad (2)$$

$$V_\beta = \frac{\sqrt{3}}{2} * [V_{bn} - V_{cn}] \quad (3)$$

The space phasor voltage can be obtained from following equations (4) and (5),

$$V_s = V_\alpha + j * V_\beta \quad (4)$$

$$\alpha = \tan^{-1} \frac{V_\beta}{V_\alpha} \quad (5)$$

SVPWM treats the inverter as a single unit. Specifically the inverter can be driven into eight unique states. Modulation is accomplished by switching the state of inverter. Space vector pulse width modulation treats the sinusoidal voltage as a constant amplitude vector rotating at constant frequency. This PWM technique approximates the reference voltage V_{ref} by a combination of the eight switching patterns (V_0 to V_7) [1]. The vectors (V_1 to V_6) divide the plane into six sectors, V_{ref} is generated by two adjacent non-zero vectors and two zero vectors.

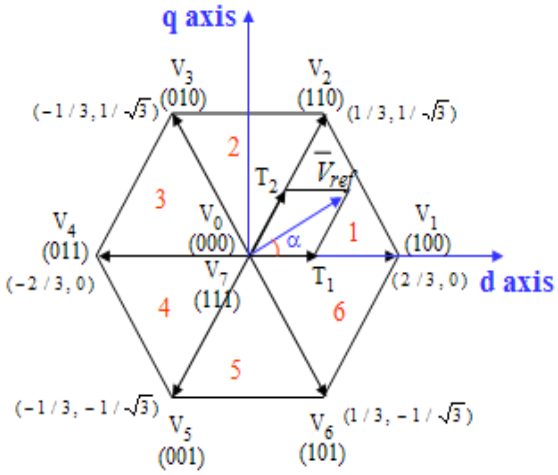

Figure 2: Voltage space vectors

3. Processors for Implementing SVPWM

Space vector pulse width modulation is an optimum pulse width modulation technique for a two level inverter which can be implemented using different processors as follows.

A. DSP

Texas Instrument Microcontroller TMS320F2812 is a new generation of controller with a main frequency of 150 MHz. TMS320F2812 consists of two event manager modules (A & B), inbuilt Analog to Digital converter and consists of 12 PWM output pins, 6 pins for each Event Manager Module[1]. The ADC module include 12 bit core with built in dual sample and hold circuits with fast conversion at 25 MHz. Module takes the input between 0 to 3V. The start of conversion process can be triggered through the event manager. The module has 16 channels, configurable as two independent 8 channel modules to serve event manager modules. Two independent modules can be cascaded to form a 16 channel module. The two 8-channel modules have capability to auto sequence a series of conversions. Each module has the choice of selecting any one of the respective eight channels available through an analog MUX. In cascaded mode auto sequencer functions as a single 16-channel sequencer. Once conversion is complete the selected channel value is stored in its respective ADCRESULT register. Auto sequencing allows the system to convert the same channel multiple times, allowing the user to perform oversampling algorithms. Code Composer software is used to interface the computer to the DSP kit of Texas Instruments [1]. The setup process for PWM generation includes the following steps [1]

- Setup and load ACTRx
- Initialize CMPRx
- Setup and load COMCONx
- Setup and load T1CON (for EVA) or T3CON (for EVB) to start the operation
- Rewrite CMPRx with newly determined values

B. dsPIC

dsPIC is suited for the application that requires DSP processing, as normal PIC is not able to do DSP fast enough. PIC is a family of modified Harvard architecture microcontroller made by Microchip Technology. The dsPIC includes 48 (Kbytes) on-chip flash program space, 2 (Kbytes) of on chip data RAM, 1 (Kbyte) of nonvolatile data EEPROM, and processor speed up to 30 (MIPS). Peripheral features consist of PWM and QEI modules for adjustable motor speed control applications [2].
The microcontroller supports not only 9 channels, each has a 10-bit resolution A/D converter with fast response time but also UART, CAN, and SPI communication. The microcontroller is set to operate at 20 (MIPS), supplied by a 5 (V) source, to control the SVPWM inverter using PWM module [2]. Required software for obtaining PWM signals is developed with C programming language in MPLAB IDE environment. dsPIC is way better at floating math work, as it has accumulators that make heavy math work easier. But it is of course more expensive and complex to use. The step of programming for PWM generation logic is as following [7]

- Disable the PWM pin (PxTCON) output drivers as an input by setting the associated TRIS bit.
- Set the PWM period by loading the TxTPER register.
- Configure the CCP module for the PWM mode by loading the PWMxCONy register with the appropriate values.
- Set the PWM duty cycle by loading the PxDCy register.
- Set the dead time by loading the PxDTCONy register.
- Configure and start Timer2.
- Enable the PWMx pin output driver by clearing the associated TRIS bit.

C. SVPWM USING ATmega328P

The Arduino Uno is a microcontroller board based on the ATmega328. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button as shown in Fig.3. Simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Arduino can do standard mathematical operations. While floating point numbers are allowed if declared as floats.

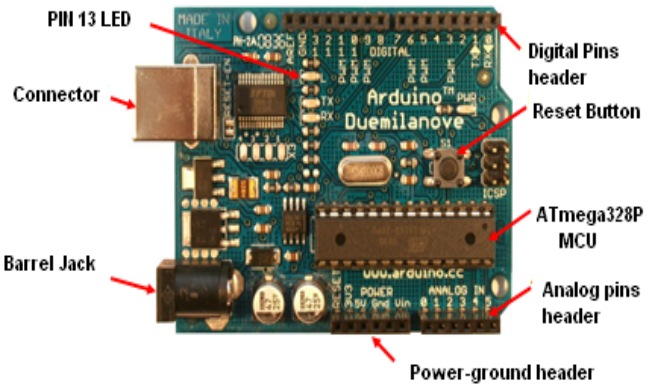

Figure 3: Arduino Board

D. Specifications

Microcontroller: ATmega328
Operating Voltage: 5V
Input Voltage (recommended): 7-12V
Input Voltage (limits): 6-20V
Digital I/O Pins: 14 (of which 6 provide PWM output)
Analog Input Pins: 6
DC Current per I/O Pin: 40 mA
DC Current for 3.3V Pin: 50 mA
Flash Memory: 32 KB (ATmega328) of which 0.5 KB used by boot loader
SRAM: 2 KB (ATmega328)
EEPROM: 1 KB (ATmega328)
Clock Speed: 16MHz

Arduino also created software which is compatible with all Arduino microcontrollers. The software, also called "Arduino", can be used to program any of the Arduino microcontrollers by selecting them from a drop-down menu. Being open source, and based around C.

The Arduino development environment contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions, and a series of menus. It connects to the Arduino hardware to upload programs and communicate with them. Software written using Arduino is called sketches. These sketches are written in the text editor. It has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino environment including complete error messages and other information. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor.

E. PWM Module

The phase correct PWM mode (WGM2:0=1 or 5) provides a high resolution phase correct PWM waveform generation option. Phase correct PWM is based on a dual slope operation. The counter counts repeatedly from BOTTOM to TOP and then from TOP to BOTTOM. TOP is defined as 0xFF when WGM2:0=1, and OCR0A when WGM2:0=5[9].

In phase correct mode the counter is incremented until the counter value matches TOP. When the counter reached TOP, it changes the count direction. TCNT0 value will be equal to TOP for one timer clock cycle. The compare unit allows generation of PWM waveforms on the OC0xpins setting. COM0x1:0=2 will produce a non-inverting PWM. An inverting PWM output can be generated by setting the COM0x1:0=3. Setting the COM0A0=1allows the OC0A pin to toggle on compare matches if the WGM02 bit is set. The actual OC0x value will only be visible on the port pin if the direction for the port pin is set as output. The PWM waveform is generated by setting the OC0x register at the compare match between OCRx and TCNT0 when the counter increments and setting the OCRx register at compare match between OCRx and TCNT0 when the counter decrements[9].

The generations of Space vector pulse Width Modulation Waveforms are shown in below Fig.5.

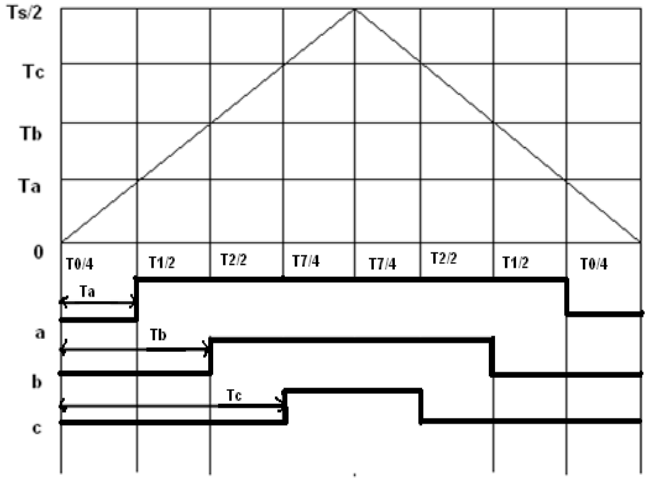

Figure 5: Generation of SVPWM waveforms

The algorithm used for implementing SVPWM inverter using ATmega is as follows

- Generate sine wave using look up table.

- Initialize Time base control registers where center aligned PWM operation takes place by selecting continuous up/down count mode.
- Initialize register values where duty cycles are loaded.
- Calculate Vs and φ based on the requirement of V_α and V_β.
- Given a period T of carrier wave, calculate T_1, T_2, and T_0.
- Calculate duty cycles T_a, T_b & T_c in TIM1, TIM2& TIM3 registers. At the beginning of a new cycle (PWM interrupt occurs) T_1, T_2, and T_0 are computed. And then, the active time of S_1, S_2,. . . , S_6 switches are consequently written in the TIM1, TIM2, and TIM3 registers of the ATmega328 microcontroller.

F. Flow Chart

The flow chart for implementing SVPWM using ATmega328p is shown in Fig.6.

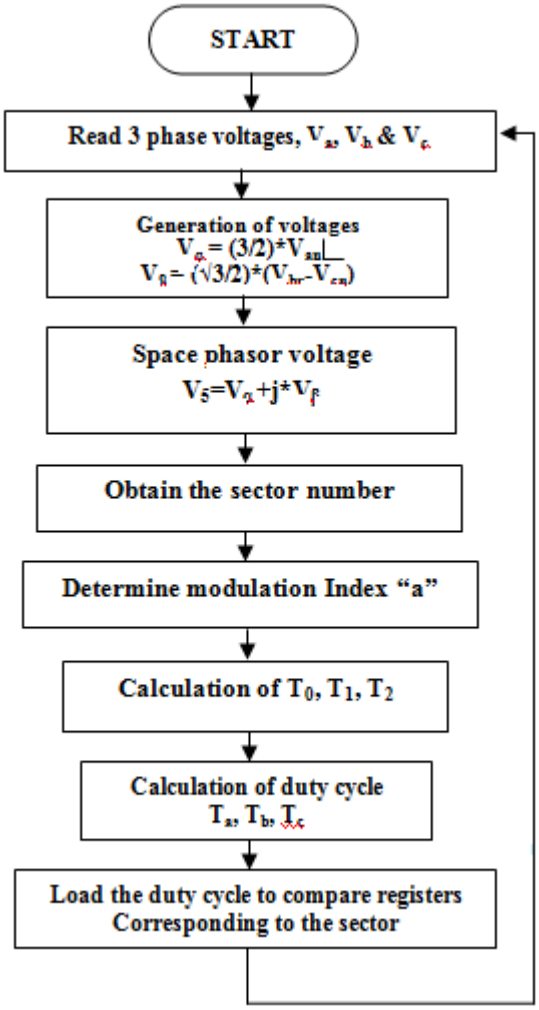

Figure 6: Flow chart for SVPWM Implementation

4. Results and Discussions

The space vector PWM waveforms generated are symmetric with respect to the middle of each PWM period. Symmetric waveform generation will eliminate even harmonics and reduces the odd harmonics. Symmetric space vector PWM generation is shown in Fig.7.

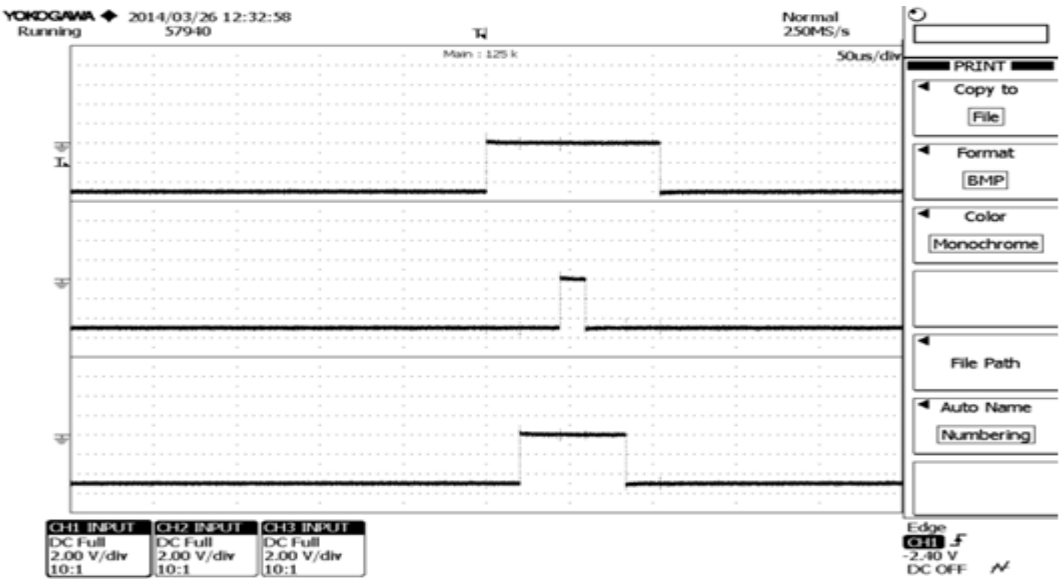

Figure 7: Symmetrical waveform generation

5. Conclusion

The basic principle of SVPWM, brief description about different processors like DSP, dsPIC and ATmega and the algorithm of SVPWM using ATmega has been discussed in this paper. It has been reviewed that ATmega with its advanced features and the advantages over other processors is the suitable processor for implementing Space Vector Pulse Width Modulation technique for the inverter switching control.

References

[1] Ronad B. F, Naik R. L, Jangamshetti Suresh.H. "A Novel Method to Eliminate Negative Time Period of SVPWM using DSP TMS320F2812",International Conferences on Renewable Energies and Power Quality(ICREPQ'11) Las Palmas de Gran Canaria (Spain),13th to 15th April,2011.
[2] Duc-Cuong Quach, Quan Yin, Yu-Feng Shi and Chun-Jie Zhou," Design and Implementation of Three-phase SVPWM Inverter with 16-bit dsPIC", 2012 12th International Conference on Control, Automation, Robotics & Vision Guangzhou, China, 5-7th December 2012 (ICARCV 2012).
[3] Zhenya Yu, Arefeen Mohammed, Issa Panahi,"Review of Three PWM techniques",Texas Instruments, DSP Automotive/Industrial Applications, Houston, TX 77477.
[4] Slamet, "Generation of Space Vector PWM Using Microcontroller ATmega 16", International Journal of Scientific & Engineering Research, Volume 4, Issue 3, March-2013 ISSN 2229-5518.
[5] Oscar Lopez Sanchez, "SVPWM for multilevel multiphase voltage source converters",Vigo,9th Jan 2009.
[6] Marwan A.A. Badran, Ahmad M. Tahir, Waleed F. Faris, "Digital Implementation of Space Vector Pulse Width Modulation Technique Using 8-bit Microcontroller", World Applied Sciences Journal 21 (Mathematical Applications in Engineering): 21-28, 2013 ISSN 1818-4952 © IDOSI Publications, 2013.
[7] Mr.N.D.Patel,H.D.Patel,R.R.Soni,T.H.Panchal, "Development and implementation of spwm Logic using dspic33fj16gs402 for three Phase inverter", Journal of information, knowledge and research in Electrical engineering.
[8] Susovan Mukhopadhyay, Sujit K. Biswas, Nirmal K. Deb, "A Simple Sector Independent Space Vector Modulation using DSP Processor", International Journal of Power Electronics and Drive System (IJPEDS) Vol.2, No.3, September 2012, pp. 297~304 ISSN: 2088-8694.
[9] ATMEL, Atmega328P datasheet
[10] dsPIC30F datasheet.

Author Profile

Asma was born in Mudhol, Karnataka, India on 7th Dec. 1987. She obtained B.E (Electrical and Electronics) from Kuvempu University, Karnataka, India in 2009. She is currently pursuing M.Tech. Degree in Power and Energy Systems in Electrical and Electronics Engineering, Basaveshwar Engineering College, Bagalkot, India. Her areas of interest include Wind-Solar Energy Systems, Transmission and Distribution systems.

Naik. R. L Was born in Herigulbal, Karnataka, India on 1st July 1974. He obtained B.E (Electrical and Electronics) from Karnataka University Dharwad, Karnataka, India in 1996 and M.Tech (Power and Energy System) from NITK Surathkal Karnataka, India in 2005. His areas of interest include Power Electronics, Drives and Renewable Energy Sources. He has attended National and International Conferences. Presently he is working as faculty in the Department of Electrical & Electronics Engineering at Basaveshwar Engineering College, Bagalkot, India.

Dr. Suresh. H. Jangamshetti: (S'88, M'90, SM'97) was born in Bijapur, Karnataka, India on May 28, 1963. He obtained his B.E (Electrical) degree from Karnataka University Dharwad in 1985 and M.Tech. (Power Systems) & Ph.D (Wind Energy Systems) from IIT Kharagpur in 1989 & 2000 respectively. His areas of interest include Wind-Solar Energy Systems, Energy Conservation, Computer Applications to Power System and FACTS He won the "Outstanding IEEE Student Branch Counsellor" award for the year 1996(R10) and 2010 (IEEE Bangalore Section) at Basaveshwar Engineering College, Bagalkot, Karnataka, India. He was Fulbright-Nehru Visiting Lecture Fellow at Michigan Technological University, Houghton MI USA during Fall 2011. He is working as Professor in the department of E&E at Basaveshwar Engineering College, Bagalkot.