Marzo, 2019 Vol.3 No.8 17-23

Desarrollo y programación de señales de conmutación para inversor trifásico basadas en técnicas SPWM e implantadas en una FPGA

Development and programming of switching signals for three-phase inverters based on SPWM techniques and implemented in an FPGA

CHAVARRÍA-DOMÍNGUEZ, Benjamín†, CHAVARRÍA-DOMÍNGUEZ, Fernando*, JIMENEZ-SILVA, J. Isidro y ALVAREZ-MARTINEZ, Luis F.

Centro Nacional de Investigación y Desarrollo Tecnológico, Departamento de Ingeniería Electrónica

ID 1er Autor: *Benjamín, Chavarría-Domínguez* / **ORC ID**: 0000-0002-1037-2313, **arXiv Author ID**: BenjaminC, **CVU CONACYT ID**: 887916

ID 1er Coautor: *Fernando, Chavarría-Domínguez* / **ORC ID**: 0000-0002-0858-1431, **arXiv Author ID**: fchavarria, **CVU CONACYT ID**: 475091

ID 2do Coautor: *J. Isidro, Jimenez-Silva* / **ORC ID**: 0000-002-4366-6350, **CVU CONACYT ID**: 1014029

ID 3er Coautor: *Luis F., Alvarez-Martinez* / **ORC ID**: 0000-0001-5894-7932, **CVU CONACYT ID**: 1015202

DOI: 10.35429/JCT.2019.8.3.17.23 Recibido 10 de Enero, 2019, Aceptado, 30 de Marzo, 2019

Resumen

Los inversores nos permiten convertir la corriente directa en corriente alterna con una forma de onda senoidal. El objetivo de este trabajo es presentar una de las técnicas que rigen la operación de estos inversores para un caso de tres fases. Se emplea el programa Matlab-Simulink para desarrollar desde un entorno gráfico y de bloques una técnica de modulación SPWM (Sinusoidal Pulse Width Modulation, por sus siglas en inglés) que permite generar los pulsos de conmutación del inversor trifásico basado en puentes H. Se muestra un método de programación basado en el entorno Icestudio para integrar la modulación SPWM y poder generar de forma física los pulsos desde una FPGA (Field-programmable gate array, por sus siglas en inglés). El valor de este trabajo radica en la descripción a detalle de los procedimientos necesarios para desarrollar la programación de la modulación SPWM que genera los pulsos de conmutación y su integración en el FPGA.

Inversores, FPGA y SPWM

Abstract

The inverters allow us to convert direct current into alternating current with a sine waveform. This work uses the Matlab-Simulink program to develop from a graphic and block environment a SPWM (Sinusoidal Pulse Width Modulation) modulation technique that allows generating the switching pulses of a three-phase inverter based on bridges H, a simulation of the pulses applied to the inverter is also performed to record the voltage at the output of the inverter. Finally, a programming method based on the Icestudio environment is shown to integrate and be able to physically generate the pulses from an FPGA (Field-programmable gate array). The value of this work lies in the detailed description of the procedures necessary to develop the programming of the SPWM modulation that generates the switching pulses and their integration into the FPGA.

Inverters, FPGA and SPWM

Citación: CHAVARRÍA-DOMÍNGUEZ, Benjamín, CHAVARRÍA-DOMÍNGUEZ, Fernando, JIMENEZ-SILVA, J. Isidro y ALVAREZ-MARTINEZ, Luis F. Desarrollo y programación de señales de conmutación para inversor trifásico basadas en técnicas SPWM e implantadas en una FPGA. Revista de Tecnología Informática. 2019. 3-8: 17-23

† Investigador contribuyendo como primer autor.

Introducción

Dentro de la electrónica de potencia existe un grupo de topologías encargadas de la conversión de corriente eléctrica C.D. a C.A. (corriente directa a corriente alterna) o viceversa. Los inversores emplean buses de C.D. que se hacen conmutar y cambiar de polaridad mediante un grupo de dispositivos semiconductores que actúan como interruptores para convertir la corriente de C.D. en una onda lo más cercana a una forma senoidal (C.A.). La calidad y cercanía de la forma de onda entregada por un inversor con respecto a una forma de onda senoidal depende en gran medida de la manera en cómo se hacen conmutar los dispositivos semiconductores. Existen diversas técnicas de modulación que generan los pulsos de comando encargados de la conmutación de los interruptores.

La técnica SPWM (Sinusoidal Pulse Width Modulation) basada en la comparación de señales portadoras (normalmente triangulares) con respecto a señales moduladoras (formas senoidales) permite obtener buenos resultados en la generación de la forma de onda del inversor. Su implementación no demanda una gran cantidad de procesamiento y es posible emplearla en frecuencias que van del rango de 1 kHz a los 15 kHz, hacer conmutar los interruptores a estas frecuencias permite obtener buenos resultados en la conversión de corriente del inversor.

Los inversores disponen de una etapa de control donde se implementa la técnica de modulación encargada de la generación de los pulsos de comando que rigen las conmutaciones de sus interruptores, es conveniente que la etapa de control está basada en un sistema embebido que emple un dispositivo digital. Sin embargo, hay que tener en cuenta varias consideraciones al elegir el tipo de dispositivo digital ya que los pulsos de comando no pueden sufrir retardos o desfaces entre ellos o de lo contrario podrían generarse errores en los tiempos de conmutación ocasionando pequeños instantes de corto circuito. Sistemas embebidos que son basados en microcontroladores tienen la limitante que debido a su forma de ejecución secuencial al ejecutar dos o más instrucciones que definan cambios en el estado de pines de salida estas órdenes se cumplan una detrás de la otra y no de forma paralela lo que ocasiona los desfaces y retardos señalados.

Un microcontrolador no es capaz de generar un pulso de menor tiempo que el necesario para ejecutar su instrucción más sencilla, por lo que la frecuencia puede resultar en otra limitante de estos sistemas, dependiendo de las prestaciones del sistema embebido.

Una alternativa conveniente consiste en el uso de sistemas basados en una FPGA (Field-programmable gate array), estos dispositivos consisten en arreglos matriciales de compuertas que mediante un lenguaje de descripción de hardware como VHDL (acrónimo proveniente de VHSIC <<Very High Speed Integrated Circuit>> y HDL <<Hardware Description Language>>) o Verilog permiten la construcción de cada una de las etapas que integran la modulación, teniendo la libertada de definir procesos que se lleven a cabo de forma paralela (evitando los desfaces) otra de las ventajas es que los sistemas embebidos basados en FPGA por lo general integran osciladores que les permiten trabajar en frecuencias en el orden los Megahertz.

Existen tres grandes empresas encargadas del desarrollo de tecnología de las FPGA: Xilix, Altera (ahora propiedad de Intel) y Lattice. Para programar directamente una FPGA desde el lenguaje VHDL es necesario superar una curva de aprendizaje que implica el conocimiento de los paradigmas de programación propios de la estructura de los lenguajes basados en la descripción de hardware, el entendimiento de la tarjeta de desarrollo donde se encuentre montada el FPGA y del entorno de programación que brinda el software proporcionado por cada empresa para sus FPGA.

En la literatura se han reportado trabajos basados en una herramienta alternativa denominada “System Generator” que proporciona la empresa Xilinx y para desarrollar esquemas de programación basados en bloques de una Toolbox desde el entorno de Matlab/Simulink, estos esquemas son traducidos a lenguaje VHDL para terminar de ser compilados de manera normal con el programa ISE proporcionado por la empresa Xilinx para la escritura de código VHDL. Sin embargo, este tipo de alternativas son condicionadas por la compra de licencias para poder ser usadas.

ISSN 2531-2197

En este trabajo se presenta una alternativa de programación basada en el uso de una FPGA de la empresa Lattice que esta soportada por el entorno de programación Icestudio, este programa es de código abierto (por lo que no es necesario la compra de licencias) y brida un entorno de programación hibrido en donde es posible programar por secciones modulares empleando representaciones de compuertas lógicas flip flops entre otros elementos, o bloques con código Verilog.

Se definió la programación de una modulación SPWM para un inversor de puente trifásico como caso por desarrollar, ya que este tipo de inversores tienen una gran aplicación para diversas actividades como la alimentación de un motor de inducción de tres fases. La Figura 1 muestra la topología del inversor de puente trifásico y su modulación tipo SPWM.

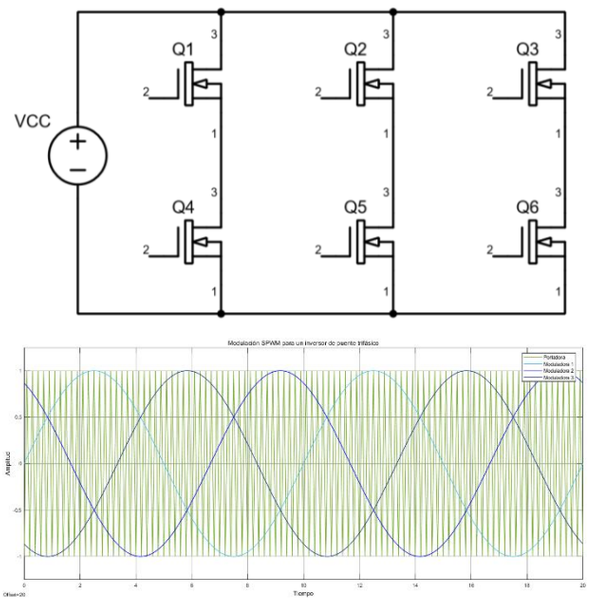

Figura 1 Inversor de puente trifásico con transistores y su respectiva modulación SPWM

El documento presentará la metodología en dos secciones, la primera mostrará la forma de generar de forma discretizada las señales de la modulación SPWM y la segunda describirá el esquema donde se programa la modulación desde el entorno del programa Icestudio. Los resultados corresponderán a las señales y pulsos de comando generadas por el FPGA una vez se implementada la modulación SPWM. Para finalizar se brindarán como conclusiones algunas consideraciones que se deben tener en cuenta al programar señales en dispositivos basados en una FPGA.

Metodología

El primer paso para la programación de la técnica de modulación SPWM en la FPGA es definir los parámetros de la señal portadora y señales moduladoras. Se propone una señal triangular portadora de 2.5 kHz comparada con tres señales senoidales moduladoras de 50 Hz, el índice de modulación en amplitud es 0.9 y el índice en frecuencia es de 50. La Ecuación 1 muestra la fórmula para la definición del índice de modulación en amplitud.

$$m_a = \frac{S_m}{S_p} = \frac{0.9\,V}{1\,V} = 1 \tag{1}$$

Donde "m_a" es el índice de modulación en amplitud, "S_m" corresponde a la amplitud de las señales moduladoras y "S_p" a la amplitud de la señal portadora.

La Ecuación 2 presenta la fórmula para el índice de modulación en frecuencia.

$$f_a = \frac{f_p}{f_m} = \frac{2500\,Hz}{50\,Hz} = 50\,Hz \tag{2}$$

Donde "f_a" es el índice de modulación en frecuencia, "f_p" corresponde a la frecuencia de la señal portadora y "f_m" a la frecuencia de las señales moduladoras.

Para programar las señales que conforman a la modulación es necesario discretizarlas en un grupo de vectores de datos. Para una correcta discretización se debe considerar que la división del número de muestras de los vectores de la señal portadora y moduladoras deben dar como resultado un valor entero. También es necesario tomar en cuenta la frecuencia del reloj base que integra el sistema embebido donde se monta la FPGA que se empleara para la implementación, nuevamente el número de muestras de los vectores que integran a las señales deben ser valores submúltiplos con respecto al valor de frecuencia del reloj.

Para este caso utilizaron 1,200 muestras para la señal portadora y para las señales moduladoras. Tomando en cuenta que la frecuencia del reloj integrado en el sistema embebido a utilizar es de 12 MHz, mediante la Ecuación 3 se corrobora que el número de frecuencia de los vectores sea un submúltiplo de la frecuencia que se tendrá de base y no tenga valores decimales.

ISSN 2531-2197

$$\frac{frecuencia\ del\ reloj}{número\ de\ muestras} = \frac{12\ MHz}{1200\ muestras} = 10{,}000 \qquad (3)$$

Desarrollo de la técnica SPWM en Simulink

La generación y discretización de las señales de la modulación SPWM se llevó a cabo en el entorno Simulink, la Figura 2 muestra el esquema empleado para realizar este proceso.

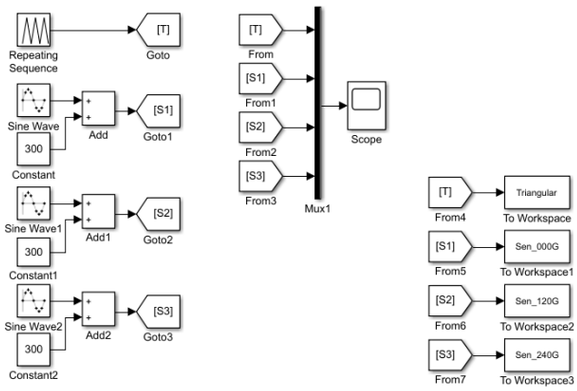

Figura 2 Esquema desarrollado en Simulink para la generación de la modulación SPWM

El esquema se ajustó a 12 segundos de tiempo de simulación con pasos 0.01 segundos para generar las 1200 muestras. Dentro del bloque "Repeating Secuence" se configura la señal triangular con 600 valores de amplitud, mientras que los bloques "Sine Wave" generan a las señales senoidales con 540 valores de amplitud, 10% menos que la señal triangular para conseguir el índice de en amplitud de 0.9.

Los bloques "Constant" se emplean como señal de offset para elevar las señales senoidales y evitar que estas crucen por cero, para la señal triangular no es necesario realizar este arreglo. Los bloques "To Workspace" con las etiquetas "Triangular", "Sen_000G", "Sen_120G" y "Sen_240G" son los encargados de adquirir y vectorizar la información de las cuatro señales que conforman a la modulación, el bloque Scope presenta la modulación SPWM como se muestra en la Figura 3.

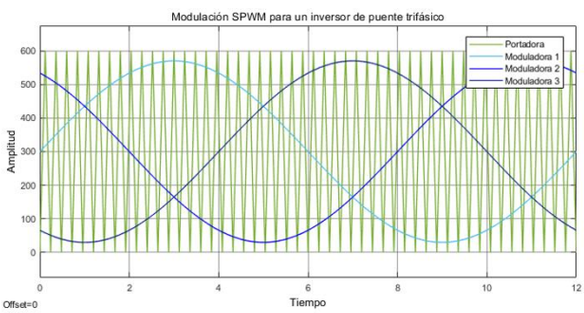

Figura 3 Modulación SPWM resultante del esquema desorrallo en Simulink

Los vectores almacenados por los bloques "To Workspace" son trasladados al programa Excel para redondear los valores de las señales discretizadas a números enteros ya que Verilog no trabaja con valores decimales, estos valores también son completados con instrucciones propias del lenguaje Verilog para crear las listas de palabras binarias que describen las señales de la modulación SPWM dentro del FPGA.

Programación de la técnica de modulación SPWM discretizada en el entorno Icestudio

En este apartado se describen las tareas que realizan los bloques del código desarrollado en el programa Icestudio. Este código genera la modulación desde el FPGA Lattice ice40hx8k Breakout Board. La Figura 4 presenta una vista general del código.

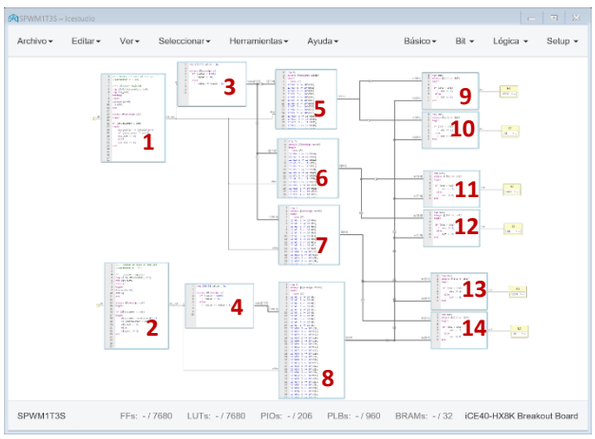

Figura 4 Esquema de programación de la modulación SPWM desarrollada en el entorno Icestudio

El bloque "1" consiste en el divisor empleado para ajustar las ondas senoidales a una frecuencia de 50 Hz, considerando el número de muestras empleadas (1,200) y la frecuencia base (12 MHz) es necesario dividir al reloj base entre el número 200. Como se muestra en las Ecuaciones 4 y 5.

$$R_{RM} = \frac{f_R}{N_m} = \frac{12\ MHz}{1{,}200} = 10{,}000 \qquad (4)$$

Donde "R_{RM}" es una relación entre la frecuencia del reloj base "f_R" y el número de muestras "N_m" de la señal discretizada.

$$f_d = \frac{R_{RM}}{f_s} = \frac{10{,}000}{50\ Hz} = 200 \qquad (5)$$

Donde "f_d" es la frecuencia que debe tener el divisor para que la señal senoidal discretizada se ejecute a una frecuencia de 50 Hz "f_S". El bloque "2" contiene al divisor que ajusta la señal triangular a una frecuencia de 2.5 kHz. Considerando las 1,200 muestras empleadas para discretizar y la frecuencia base (12 MHz), es necesario dividir al reloj base en 4 para obtener los 2.5 kHz. Las ecuaciones 6 y 7 muestran estas operaciones de modo similar que en las Ecuaciones 4 y 5.

$$R_{RM} = \frac{f_R}{N_m} = \frac{12\ MHz}{1{,}200} = 10{,}000 \quad (6)$$

Donde "R_{RM}" es una relación entre la frecuencia del reloj base "f_R" y el número de muestras "N_m" de la señal discretizada.

$$f_d = \frac{R_{RM}}{f_T} = \frac{10{,}000}{2.5\ kHz} = 4 \quad (7)$$

Donde "f_d" es la frecuencia que debe tener el divisor para que la señal triangular discretizada se ejecute a una frecuencia de 2.5 kHz "f_T".

Los bloques "3" y "4" corresponden a contadores de 11 bits necesarios para asignar cada espacio a las 1,200 muestras de cada una de las señales (senoidales y triangulares), estos contadores se desbordan al llegar al número 1,199 en binario, ya que el conteo inicia desde el número cero.

Los bloques "5", "6" y "7" encierran en un ciclo "case" las palabras binarias que representan a las señales senoidales, cada palabra binaria contiene la información de una muestra, por lo que la señal senoidal se compone de 1,200 palabras binarias. El bloque "8" contiene dentro de un ciclo case las 1,200 palabras binarias que representan a la señal triangular.

Los bloques "9", "10", "11", "12", "13" y "14" comparan la magnitud de las señales senoidales y triangular, se emite un estado alto (1 binario) o bajo (0 binario) dependiendo del criterio de comparación designado por una sentencia "if" y "else".

Los bloques "9" y "10" reciben la señal senoidal del bloque "5" por "in1" y la señal triangular del bloque "8" por "in2". Dentro del bloque "9" se define la condición que cuando "in1 < in2" se emita un estado alto (1 binario), en caso de que "in1 > in2" se emita un estado bajo (0 binario). Para el bloque "10" se realiza la función contraria, es decir, cuando "in1 > in2" se emite un estado alto y en caso de que "in1 < in2" se emite un estado bajo.

Los bloques "11" y "12" tienen la configuración muy similar a la anterior, reciben la señal senoidal del bloque "6" por "in1" y la señal triangular del bloque "8" por "in2". El bloque "11" define la condición que cuando "in1 < in2" se emita un estado alto y en caso que "in1 > in2" se emita un estado bajo. El bloque "12" realiza la función contraria, cuando "in1 > in2" se emite un estado alto y cuando "in1 < in2" se emite un estado bajo.

Los bloques "13" y "14" vuelven a repetir la configuración vista previamente, reciben la señal senoidal del bloque "7" por "in1" y la señal triangular del bloque "8" por "in2". El bloque "13" define que cuando "in1 < in2" se emita un estado alto y cuando "in1 > in2" se emita un estado bajo. El bloque "14" efectúa la función contraria, cuando "in1 > in2" se emite un estado alto y cuando "in1 < in2" se emite un estado bajo.

Resultados

Al ejecutar la verificación y compilación del código se pudo observar la cantidad de recursos empleados (ver Figura 17) para la implantación en la FPGA Lattice ice40hx8k Breakout Board, siendo un total de 54 Flip Flops, 5,676 LUTs, 17 PIOs, 773 PLBs y 0 BRAMs.

Una vez cargado el código en el FPGA se censaron las formas de onda senoidal y triangular digitalizadas en palabras binarias, así como los canales de salida de los pulsos correspondientes a las modulaciones SPWM unipolares trifásicas.

La Figura 5 muestra el comportamiento de la señal senoidal digitalizada en palabras binarias de 10 bits y 1,200 muestras, puede corroborarse que su ciclo tiene una frecuencia de 50 Hz

ISSN 2531-2197

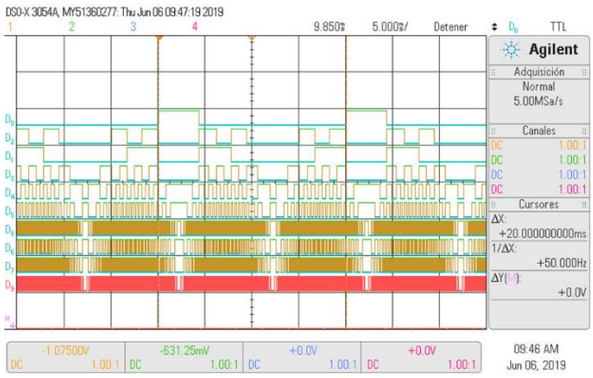

Figura 5 Lectura en el osciloscopio de la forma de onda senoidal discretizada e implementada en el FPGA

La Figura 6 muestra el comportamiento de la señal triangular digitalizada en palabras binarias de 10 bits y 1,200 muestras, puede corroborarse que su ciclo tiene una frecuencia de 2.5 kHz.

Figura 6 Lectura en el osciloscopio de la forma de onda triangular discretizada e implementada en el FPGA

La Figura 7 muestra las salidas correspondientes a los pulsos de la modulación SPWM unipolar trifásica, puede corroborarse que su frecuencia es de 50 Hz o 20 milisegundos.

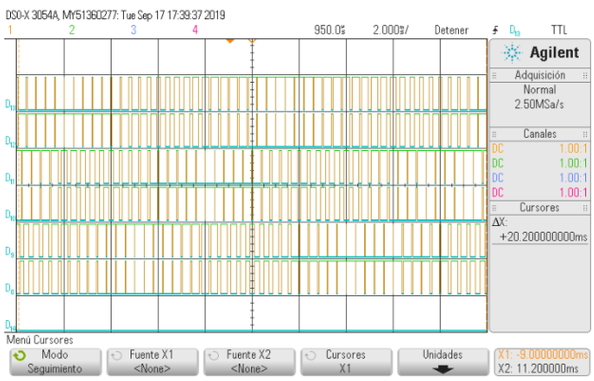

Figura 7 Lectura en el osciloscopio de los pulsos de comando de la modulación SPWM generados en el FPGA

Conclusiones

El uso de dispositivos embebidos basados en la descripción de hardware como las FPGA permite agilizar el desarrollo de la etapa de generación de pulsos de comando de un sistema de control de potencia, sin embargo, se deben considerar algunos puntos al desarrollar las etapas de programación:

– Para este trabajo se discretizaron dos tipos de señales (senoidal y triangular), este proceso requiere que los valores de muestreo representen lo más cercano posible la forma original de la señal, es necesario cuidar el redondeo de valores decimales al discretiza.

– Al momento de definir las frecuencias de las señales discretizadas dentro del FPGA se tiene que prever que los valores de división del reloj base para los subrelojes de las señales discretizadas sean submúltiplos del valor del reloj base, un valor de división arbitrario podría llevar al desarrollo de un subreloj inexacto que presente errores de acarreo.

– Es importante considerar la extensión de bits que será asignada a las palabras binarias donde se guardan las señales discretizadas ya que impactarán directamente en el número de componentes que serán empleados al implementar el código final dentro del FPGA.

Referencias

Chen, W., Xiao, F., Liu, J., & Wang, H. (2013, December). Study on the topology of three-phase inverter systems based on parallel-connected bridges. In Proceedings 2013 International Conference on Mechatronic Sciences, Electric Engineering and Computer (MEC) (pp. 3678-3682). IEEE.

Cisneros, M. A. P., Jiménez, E. V. C., & Navarro, D. Z. (2014). Fundamentos de Robótica y Mecatrónica con MATLAB© y Simulink©. RA-MA Editorial.

González J. (2016 - 2019). FPGAwars. España: FPGAwars/icestudio. Recuperado de http: http://fpgawars.github.io/

ISSN 2531-2197

García Franquelo, L., Rodriguez, J., Leon, J. I., Kouro, S., Martín Prats, M. D. L. Á., & Portillo Guisado, R. C. (2008). The age of multilevel converters arrives. IEEE Industrial Electronics Magazine, 2 (2), 28-39.

Hannan, S., Aslam, S., & Ghayur, M. (2018, February). Design and real-time implementation of SPWM based inverter. In 2018 International Conference on Engineering and Emerging Technologies (ICEET) (pp. 1-6). IEEE.

Haokai, L., & Jian, Y. (2013, August). Research on inverter control system based on bipolar and asymmetric regular sampled SPWM. In 2013 IEEE 11th International Conference on Electronic Measurement & Instruments (Vol. 2, pp. 595-598). IEEE.

Hirani, M., Gupta, S., & Deshpande, D. M. (2014, May). Comparison of performance of induction motor fed by sine pulse width modulated inverter and multi level inverter using XILINX. In *2014 IEEE International Conference on Advanced Communications, Control and Computing Technologies* (pp. 264-269). IEEE.

Pujari, S., Yeotkar, A., Shingare, V., Momin, S., & Kokare, B. (2015, May). Performance analysis of microcontroller and FPGA based Signal Processing a case study on FIR filter design and implementation. In 2015 International Conference on Industrial Instrumentation and Control (ICIC) (pp. 252-257). IEEE.

Ranganathan, S., Sriharsha, H. S., & Krishnan, R. K. (2015, December). Low cost FPGA implementation of SPWM using dynamically configurable switching frequency for three phase voltage source inverter. In *2015 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)* (pp. 1-5). IEEE.

Rodriguez, J., Lai, J. S., & Peng, F. Z. (2002). Multilevel inverters: a survey of topologies, controls, and applications. IEEE Transactions on industrial electronics, 49(4), 724-738.

Wunnava, S. V., & Sanchez, I. (1999, March). Correlating software modelling and hardware responses for VHDL and Verilog based designs. In Proceedings IEEE Southeastcon'99. Technology on the Brink of 2000 (Cat. No. 99CH36300) (pp. 122-125). IEEE.

ISSN 2531-2197