TecNM-Apizaco
Investigación, Desarrollo e Innovación en la Ingeniería
ISSN: en trámite

Fecha de recepción: 06 de marzo de 2026, fecha de publicación en línea: junio de 2026.

Space Vector Modulation for Feeding AC Motor Drives Using STM32F446RE Microcontroller

Roberto Morales-Caporal, Diego Aguayo-Díaz, Rafael Ordoñez-Flores, Mario E. Leal-López, and Edmundo Bonilla-Huerta
Tecnológico Nacional de México–Instituto Tecnológico de Apizaco. San Andrés Ahuashuatepec, Mpio, de Tzompantepec, Tlaxcala, C.P. 90491, México.

Author correspondence: Roberto Morales-Caporal (email: roberto.mc@apizaco.tecnm.mx)

Abstract- This paper presents the digital implementation of the space vector modulation (SVM) technique using the STM32 nucleo-F446RE microcontroller unit (MCU) to activate a voltage inverter that feeds three-phase AC motors. The SVM generates modulated trigger signals to activate the power switches of a voltage inverter in the correct sequence, creating AC waveforms to drive three-phase AC motors at the desired speed or torque. The SVM is characterized by increasing the efficiency of the electric drive and by generating higher-quality output waveforms with low harmonic distortion, compared to the conventional sinusoidal PWM technique. The implementation of the SVM algorithm using the STM32CubeMx and STM32CubeIDE development tools is explained in detail. To demonstrate the correct implementation of the MCU-based code, a three-phase squirrel-cage induction motor is connected to the output of a commercial voltage inverter, triggered by the implemented SVM. The pulse sequences generated to activate the voltage inverter are experimentally verified, as well as the voltage and current waveforms supplying the AC motor.

Keywords: AC motors, microcontroller unit, space vector modulation, voltage inverter, variable frequency drive.

I. INTRODUCTION

THE demand for alternating current (AC) motors drives for variable rotor speed applications is constantly increasing, driven by the global push for energy efficiency, stricter sustainability regulations, and cost reduction requirements in modern industrial automation [1]. AC motor drives are used to control the magnitude and frequency of the AC power that feeds the motor, achieving precise control of the motor shaft with high energy conversion efficiency when transferring electromagnetic torque to load torque. Furthermore, it is possible to enable other functions such as field weakening, which increases the dynamic performance of the system by allowing the motor to operate at speeds significantly higher than its nominal base speed, thus potentially doubling the motor's power output. These and other features make electric AC motor drives indispensable for modern industrial automation [2], [3].

The AC drive system consists of three distinct subsystems: 1) rectifier (usually uncontrolled), 2) capacitive filter in the DC-link, and 3) voltage source inverter (VSI), as shown in Fig. 1.

The three-phase, two-level (2-L) VSI is the most used inverter topology because it is simple and cheap [4]. Technological advances in power electronics development have enabled VSIs to become increasingly efficient and to operate at higher power levels and frequencies. In this regard,

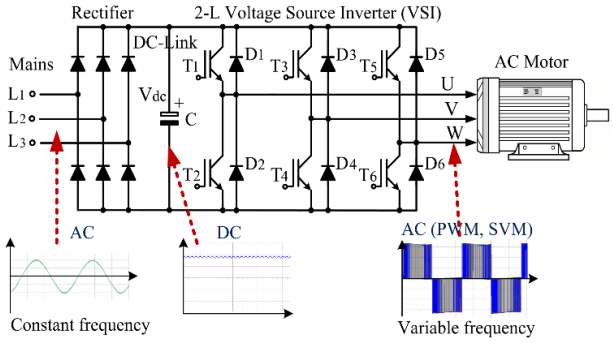

Figure 1. AC motor drive.

insulated-gate bipolar transistor (IGBT), and metal-oxide-semiconductor field-effect transistor (MOSFET) have become established as the most common power switches comprising the VSI.

The switching signals required to trigger the power switches of the 2-L VSI can be implemented using different methods, with two being the most common: 1) the sinusoidal pulse-width modulation (SPWM), and 2) the space vector modulation (SVM) [5]. Of the two, the SVM is better, as it performs calculations to adjust the duration of the "on"-times to synthesize the desired output voltage waveform to generate a sinusoidal current to feed AC motors drives with low total harmonic distortion (THD) [6], [7].

SVM is compute-intensive, nevertheless it is by far the most recommended modulation strategy for vector control of AC motors, as it increases system efficiency, compared to SPWM. Implementing the SVM algorithm is not trivial, as it requires several operations, decisions and good handling of timers and interruptions [8], [9]. However, with the development of microcontroller units (MCUs) that process in real time, nowadays it is possible to implement the SVM algorithm in a more affordable and faster way on low-power, low-cost digital platforms.

In summary, the main contributions of this paper are:

- The digital implementation of the SVM algorithm using the NUCLEO-F446RE MCU, which computes in real-time and is a low-power and low-cost digital platform.
- The method for implementing the SVM algorithm using the STM32CubeMx and STM32CubeIDE development tools is explained.
- A test bench is set-up using a squirrel-cage induction motor and a three-phase 2-L VSI to experimentally validate the MCU-based code.

II. THREE-PHASE, TWO-LEVEL VSI

In Fig. 1, it is possible to observe the circuit topology of the three-phase 2-L VSI. This topology consists of three branches, each with two IGBTs (in this case) and their respective freewheeling diodes. The number of voltage levels at the inverter's output is determined by its pole voltages. In the case of a 2-L VSI, $+V_{dc}/2$ and $-V_{dc}/2$ are the pole voltages.

With this VSI topology, it is possible to obtain eight different switching states. The switching states can be expressed as switching functions (S_U, S_V, S_W) with value "1" when the switch is set to the positive voltage or "0" when the switch is set to the negative voltage. Each switching state generates one voltage space vector. Six of them (see Fig. 2) are active voltage space vectors (AVSVs) designated as $\underline{u}_1(100)$, $\underline{u}_2(110)$, $\underline{u}_3(010)$, $\underline{u}_4(011)$, $\underline{u}_5(001)$, $\underline{u}_6(101)$, and two are zero voltage space vectors (ZVSVs) designated as $\underline{u}_0(000)$, and $\underline{u}_7(111)$ [9], [10].

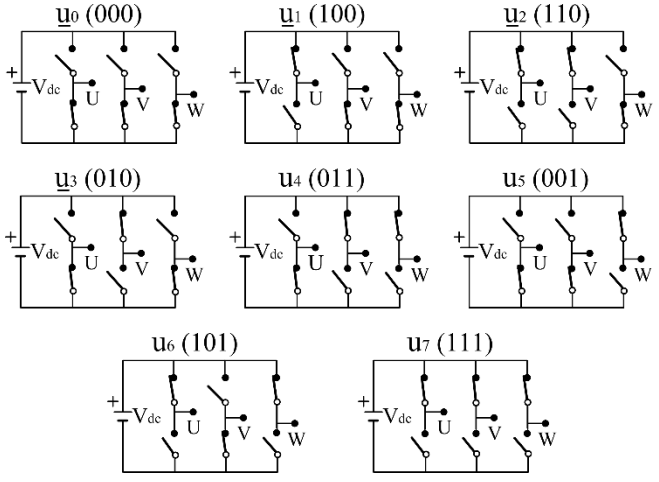

Figure 2. The 8 possible switching states of a three-phase 2-L VSI.

Table I. Switching states of the three-phase 2-L inverter and the generated line-to-line voltages.

Switching state	Conducting IGBTs	Vuv	Vvw	Vwu
000	T_2, T_4, T_6	0	0	0
001	T_2, T_4, T_5	0	-Vdc	+Vdc
010	T_2, T_3, T_6	-Vdc	+Vdc	0
011	T_2, T_3, T_5	-Vdc	0	+Vdc
100	T_1, T_4, T_6	+Vdc	0	-Vdc
101	T_1, T_4, T_5	+Vdc	-Vdc	0
110	T_1, T_3, T_6	0	+Vdc	-Vdc
111	T_1, T_3, T_5	0	0	0

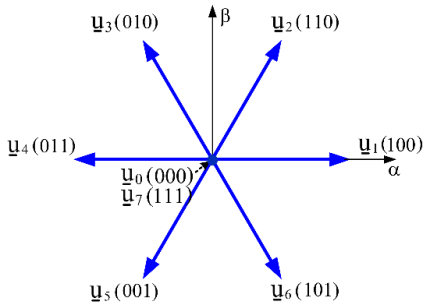

Figure 3. The 6 AVSP and 2 ZVSP generated by the three-phase 2L VSI in the complex plane α−β.

Table I summarizes the switching states as space vectors, the conducting IGBTs and the generated line-to-line voltages. The six AVSVs and the two ZVSVs are graphically represented in a complex plane α−β, as shown in Fig. 3.

To trigger the VSI in controlled sequence for generating AC waveforms as close as possible to sinusoidal waveforms from a constat DC source (V_{dc}), different methods have been developed. Among the existing methods, the SPWM is the simplest to implement and the most widely used to control VSI-powered AC motors [11]. However, one of the most significant drawbacks of SPWM is that, since the sinusoidal reference signals vary during a PWM period, the relationship between these signals and the carrier signal is not fixed, which produces a high THD in the output voltage, causing unwanted low-frequency ripples in the speed and torque of the electrical drive.

To mitigate some of the drawbacks of SPWM, the SVM technique was developed [12], [13]. With this technique, instead of using a separate modulator for each of the phases, a complex reference voltage space vector ($\underline{u}_s$*) is processed, which intrinsically considers the interaction between the three-phases. The SVM technique has been shown to generate less THD in both voltage and current signals, resulting in more efficient use of the power supplied to the AC motor compared to the SPWM technique. In addition, SVM is advantageous because it exhibits a constant switching frequency.

III. SPACE VECTOR MODULATION

A. Principle

The SVM technique is based on the geometric construction of a required voltage space vector $\underline{u}_s$* using some of the eight

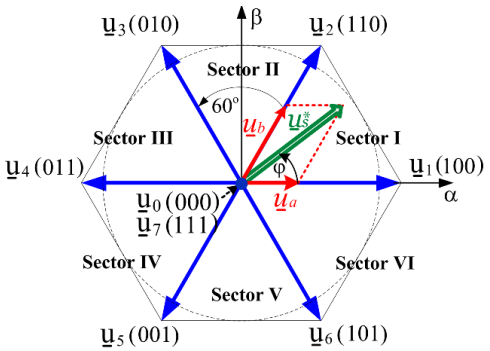

Figure 4. Synthesis of a $\underline{u}_s^*$ in sector I, using the AVSV $\underline{u}_1$ and $\underline{u}_2$.

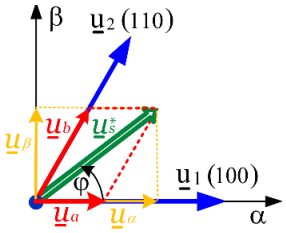

Figure 5. $\underline{u}_s^*$ in sector I and synthesized form two adjacent vectors.

switching states of a three-phase 2-L VSI. The $\underline{u}_s^*$ is synthesized from two adjacent AVSVs and using a ZVSV, which are selected according to the location of the $\underline{u}_s^*$ in the α–β plane (see Fig. 4) [14]. The AVSVs in the α–β plane, are symmetrically spaced at 60° intervals, and the area between any two of these vectors is called sector. For example, sector II in Fig. 4 is the area flanked by the AVSVs $\underline{u}_2$ and $\underline{u}_3$, and so on.

The goal is to construct a rotating space voltage vector (space voltage phasor) in the α–β plane, so that it subsequently reproduces three-phase sinusoidal waveforms when it becomes a function of time again, but with the possibility of being able to adjust its amplitude and frequency.

The main challenge when synthesizing three-phase voltages from the VSI lies in its discrete nature. To overcome this, the VSI must modulate the width of the output pulses, so that their instantaneous average magnitude equalizes the magnitude of the $\underline{u}_s^*$ [15].

B. Undermodulation Region

Fig. 5 shows a zoom of the case where $\underline{u}_s^*$ is located in sector I, this is constructed from the vector addition of two suitable adjacent AVSVs. For the case where $\underline{u}_s^*$ is in sector I with $\underline{u}_1$ and $\underline{u}_2$. Additionally, an appropriate ZVSV must be used (the selected ZVSV will be the one that exhibits the fewest switching state changes relative to the used AVSVs, to maintain the switching frequency low).

Considering the undermodulation region where the VSI transfer characteristics are naturally linear, i.e., the u_s^* will have a maximum amplitude of $u_s^* = \sqrt{3}/2\ V_{dc}$ (represented in Fig.4 by a circle), thus, when the $\underline{u}_s^*$ is in sector I, it can be calculated as [2]:

$$\underline{u}_s^* \ sin\left(\frac{\pi}{3} - \varphi\right) = \underline{u}_a \ sin\frac{\pi}{3} \qquad (1)$$

$$\underline{u}_s^* \ sin(\varphi) = \underline{u}_b \ sin\frac{\pi}{3} \qquad (2)$$

solving for both components, (1) and (2) are rewritten as

$$\underline{u}_a = \frac{2}{\sqrt{3}} \underline{u}_s^* \ sin\left(\frac{\pi}{3} - \varphi\right) \qquad (3)$$

$$\underline{u}_b = \frac{2}{\sqrt{3}} \underline{u}_s^* \ sin(\varphi) \qquad (4)$$

where $\underline{u}_a$ and $\underline{u}_b$ are the components of $\underline{u}_s^*$ that are arranged in the same direction as $\underline{u}_1$ and $\underline{u}_2$ respectively. φ represents the angle between the $\underline{u}_s^*$ and the AVSV immediately adjacent to its right side.

Considering a constant time (T_c) that lasts half the sampling time (T_S), the vector addition can be expressed as:

$$\underline{u}_s^* = \underline{u}_a + \underline{u}_b = \underline{u}_1 \frac{t_a}{T_c} + \underline{u}_2 \frac{t_b}{T_c} + \left(\underline{u}_0 \ or \ \underline{u}_7\right) \frac{t_0}{T_c} \qquad (5)$$

being

$$t_a = \frac{u_a}{\underline{u}_1} T_c \quad (6), \qquad t_b = \frac{u_b}{\underline{u}_2} T_c \quad (7) \text{ and } \quad t_0 = T_c - (t_a + t_b) \quad (8)$$

It is observed that the time intervals t_a and t_b satisfy the refe rence voltage, but the time t_0 fills the remaining space for T_c with the ZVSV either $\underline{u}_0$ or $\underline{u}_7$.

C. Switching Pattern

After calculating the switching time, these are arranged according to a symmetrical pattern corresponding to the sector where the $\underline{u}_s^*$ is located. The main objective of this symmetrical pattern is to reduce THD in the output waveform.

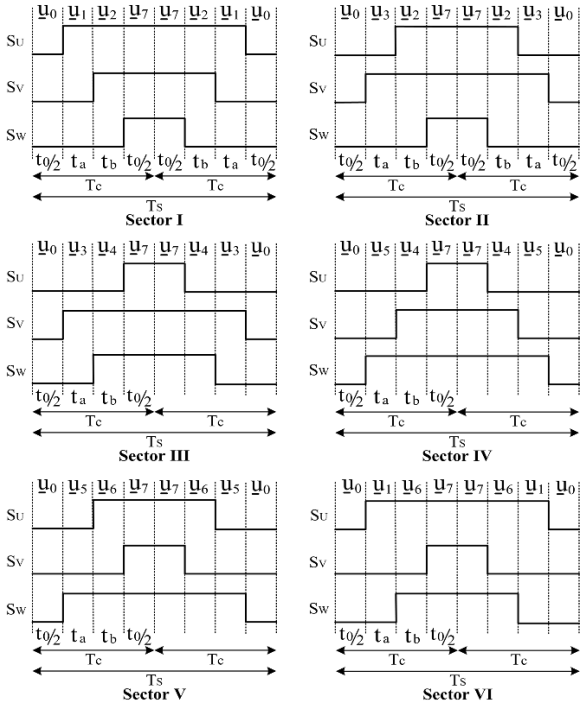

Figure 6. Switching signal sequence using thew SVM technique for the six sectors, when rotor turns counterclockwise.

The arrangement of commutation times and the switching pattern for each sector when the rotor turns counterclockwise are shown in Fig. 6.

The sequence within the first half of the *Ts* is reversed in the second half, ensuring symmetry, which helps to reduce harmonic content and facilitates the digital implementation.

IV. DIGITAL SYSTEM

A. The Microcontroller

The STM32 NUCLEO-F446RE is a 32-bit MCU developed by STMicroelectronics. It is based on the ARM CortexM4 core with floating-point unit (FPU) [16].

Common Characteristics

- STM32 MCU in LQFP64 encapsulation.
- Flash memory: 512 KB
- 128 KB SRAM(divided into several blocks).
- 2 advanced timers with support for PWM and dead-time.
- General and special purpose timers.
- 50 GPIOs programmable as in, out or interrupt.

Specific Features

- External or internal SMPS to generate logical supply
- 180 MHz (Maximum operating frequency).
- 3 12-bit ADCs, up to 2.4 MSPS, with multiple channels.
- 1 12-bit DAC with 2 channels.
- Comm: 3xUSART, 3x SPI, 2x I2C.
- Debugger/ST-LINK programmer (STLINK V2-1, V3E, and V2EC) integrated with USB renumeration capability.

B. STM32CubeMx

STM32CubeMX is a graphical tool that allows for the simple and intuitive configuration of MCUs and development platforms of the STM32 family. This tool allows the configuration of peripherals, timers, and other MCU features, as well as the generation of the corresponding preliminary C code for the Arm® Cortex®-M core through a step-by-step process [17].

First, the user intuitively selects the STM32 development platform that matches the required set of peripherals. The next step consists in configuring each required embedded software thanks to a pinout-conflict solver, a clock-tree setting helper, a power-consumption calculator, and a utility that configures the peripherals and the middleware stacks. In fact, it creates blueprints (the project configuration) and initializes all the hardware peripherals like clocks, GPIO, UART, I2C, etc.

C. STM32CubeIDE

STM32CubeIDE is an all-in-one multi-SO development tool, part of the STM32Cube software ecosystem. It is an advanced C/C++ development platform with peripheral configuration, code generation, code compilation and debugging functions for STM32 microcontrollers [18].

After selecting an empty STM32 MCU or a pre-configured microcontroller, the project is created and the initialization code is generated. This interface integrates projects creation of STM32 and configuration capabilities to save development time. It includes compiler analyzers, memory requirements, debugging functions (that include CPU kernel log views), peripheral memories and registers, real-time variables observation, a serial cable interface and a fault analyzer.

V. DIGITAL IMPLEMENTATION

A. Algorithm Implementation Sequence

Implementing the SVM strategy in a digital system involves not only translating the theory into code, but also translating and structuring the algorithm to be compatible with hardware and software resources of the digital system.

In this case, the first step will be to describe the SVM and structure it into a series of steps.

Algorithm Implementation Sequence

1. Generate a rotating reference space vector with a rotation speed equal to $\omega = 2\pi f$ (where f is the fundamental freq.).
2. Establish a space (in α-β plane) where the vector rotates; this space will be divided into 6 sectors.
3. Determine the position of $\underline{u}_s$* in the α-β plane.
4. Obtain components $\underline{u}_\alpha$ and $\underline{u}_\beta$ of the $\underline{u}_s$* in α-β (see Fig. 3).
5. Define the PWM period.
6. Obtain the angle φ of the $\underline{u}_s$*.
7. Determine the projections $\underline{u}_a$ and $\underline{u}_b$ of $\underline{u}_s$*.
8. Calculate the times t_a, t_b, and t_0.
9. Generate the three PWM outputs and their complements.

Note that this implementation sequence does not require the three reference sinusoidal waves, nor does it apply the Clarke transform, since it directly generates the rotating space vector with the required rotation components.

B. Timers

The STM32-F446RE MCU has timers with features necessary for generating PWM signals, which facilitate the digital implementation of the algorithm.

Timer TIM1

The SVM technique requires the generation of three PWM signals (to activate the upper IGBTs of the inverter) and three complementary negated signals (to activate the lower IGBTs of the inverter). The TIM1 timer has three main channels and three complementary channels, making it ideal for controlling the three-phase inverter.

The SVM technique requires symmetrical PWM signals, so selecting the center-aligned PWM counting mode of the TIM1 meets the symmetry requirements. Since a switching frequency of 10 kHz for the VSI is selected, the timer meets the frequency requirements, in addition the TIM1 timer has a configurable dead-time insertion for complementary signals, which is necessary to prevent short circuits in any of the three inverter branches. For the implementation of the algorithm on the STM32-F446RE, timer TIM1 will be used to generate the trigger signals along with their respective complementary signals. However, a second timer is necessary to update the

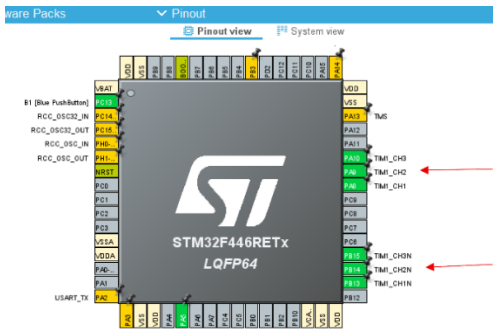

Figure 7. Assigning physical pins to TIM1 channels using the STM32CubeMX tool.

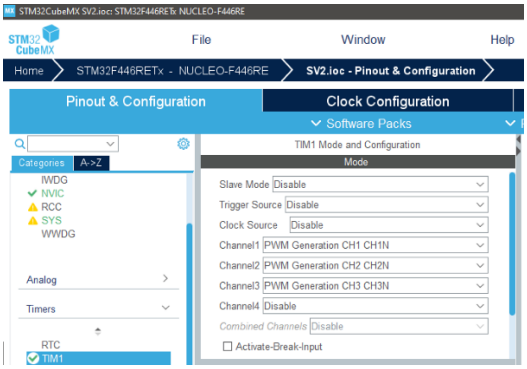

Figure 8. Configuring channels as PWM Generation.

process variables via interrupts, thus reducing the processing load on TIM1 by delegating the value updates to a second timer. Any general-purpose timer, such as TIM3, can be used for this purpose.

Assigning Output Pins to Timer TIM1

Fig. 7 shows the CubeMX interface, where the selected MCU is displayed in the foreground. In this case, 6 outputs are used, 3 of which are complementary. Therefore, pins PA8, PA9, and PA10 have been selected as channels 1, 2, and 3, and pins PB13, PB14, and PB15 as complementary channels of timer TIM1, as depicted in Fig. 7.

Switching Frequency and PWM Mode Configuration

Locate the configuration section on the left side of the interface and go to "TIM1 Mode and Configuration" and configure the three TIM1 channels as PWM Generation, as detailed in Fig. 8. In this case, the period value for ARR was set to 50 to generate 20 kHz frequency and to produce a symmetrical 10 kHz PWM signal.

The PWM counting is also selected as center aligned mode as shown in Fig. 9 [19]. In this mode, the counter counts-up from 0 to ARR and then counts-down from ARR to 0, generating one full cycle. This results in a symmetrical PWM signal: the rising and falling edges are evenly spaced with respect to the center of the period. The PWM signal will be center-aligned and will increase or decrease on both sides.

Dead-Time

Dead-time is the time delay between the gating signals of

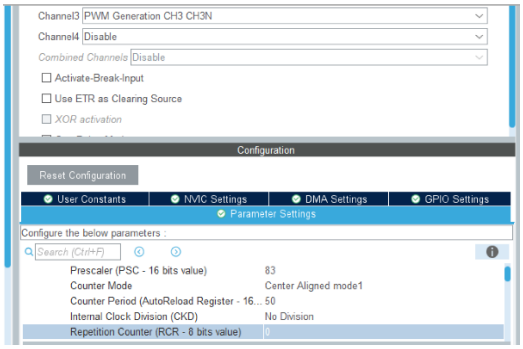

Figure 9. Frequency and PWM mode settings in STM32CubeMX.

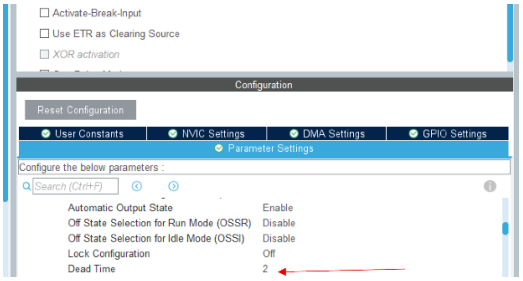

Figure 10. Dead-time setting.

switching devices to prevent them from turn-"on" or turn-"off" at the same time and therefore avoiding short circuits. It is a critical parameter for generating PWM signals.

On the "Stop and downtime management" tab, it is possible to set the dead-time, in this case to 2 µs, using the procedure to calculate the timer ticks based as function of the operating frequency (see Fig. 10) [20].

C. Code Programming

Main Code

The main code begins by defining some important constants, such as the π value, the frequency of the desired fundamental output sine wave, the frequency of the PWM signal, and the ARR period for the 10 kHz signal. Then, the $\underline{u}_s$* is generated as shown in Fig. 11. It should be noted that the angle theta must be encapsulated, this is so that if the angle theta exceeds 2π, the rotation calculation resets to 0. This ensures that the advance of theta remains between 0 and 2π.

Once the reference vector is established, the space where it will rotate is generated. This space must be divided into 6 parts, and its position at each instant must be known.

First, within the TIM3 callback, a variable "sector" is defined as an integer value. This variable will be the result of the sector calculation in the main block called "calculate_sector", as seen in the upper lines of Fig. 12. The sector is the result of dividing the angle theta where the vector is located by π/3, or 60°. If the calculation yields a value greater than 6, then it is assumed that the vector is in sector I. The alpha-beta components of $\underline{u}_s$* are computed using trigonometry as: v_α=cos(φ), v_β=sin(φ) and the magnitude and the angle φ of the $\underline{u}_s$* are calculated using: $\underline{u}_s$*= $\sqrt{(u_\alpha^2 + u_\beta^2)}$ and $\varphi = \tan^{-1} u_\beta / u_\alpha$, as detailed in the lower lines of Fig. 12.

```
/* Private user code ---------------------------------------
/* USER CODE BEGIN 0 */
#include <math.h>

#define PI 3.14159265f
#define FREC_SENOIDE 20.0f
#define FREC_PWM 10000.0f
#define PWM_TIEMPO 50


float theta = 0.0f;
float avance_theta = 2.0f * PI * FREC_SENOIDE / FREC_PWM;
```

Figure 11. Declaration of constants and generation of the $\underline{u}_s$* using the STM32CubeIDE interface.

```
int calcular_sector(float theta)
{
    sector = (int)(theta / (PI / 3.0f)) + 1;
    return (sector > 6) ? 1 : sector;
}

void calcular_tiempos_svm(float Va, float Vb, )
{
    float Vref = sqrtf(Va * Va + Vb * Vb);
    float angulo = atan2f(Vb, Va);
    if (angulo < 0)
        angulo += 2 * PI;
```

Figure 12. Calculation of the sector and determination of the magnitude and angle of the $\underline{u}_s$*.

For the declaration of times t_a, t_b, and t_0: within the TIM3 callback, the times t_a, t_b, and t_0 are declared, as well as in the main block within the CALCULATE_SVM_TIMES instruction block. The time *Ts* or PWM period is also adjusted. After obtaining all the necessary elements, the projections $\underline{u}_a$ and $\underline{u}_b$ of $\underline{u}_s$* onto the adjacent space vectors are calculated [21]. Times t_a, t_b, and $t_0/2$ are assigned, depending o n which sector the $\underline{u}_s$* is, and the following assignment of times t_a, t_b, and $t_0/2$ will be carried out according to table II.

Table II. Assignment of times ta, tb and $t_0/2$, depending on which sector the $\underline{u}_s$* is.

Sector	Su	Sv	Sw
I	ta+tb+ $t_0/2$	tb+ $t_0/2$	$t_0/2$
II	ta+ $t_0/2$	ta+tb+ $t_0/2$	$t_0/2$
III	$t_0/2$	ta+tb+ $t_0/2$	tb+ $t_0/2$
IV	$t_0/2$	ta+ $t_0/2$	ta+tb+ $t_0/2$
V	tb+ $t_0/2$	$t_0/2$	ta+tb+ $t_0/2$
VI	ta+tb+ $t_0/2$	$t_0/2$	ta+ $t_0/2$

Generation of Trigger Signals

To generate the trigger signals, the Capture and Compare Register (CCR) of TIM1 is used. This is called through the HAL_TIM_SET_COMPARE macro, which converts the application times into timer ticks and then into PWM cycles. Within the CCR macro, it is specified that it is a TIM1 register with the line &htim1 and the channel with TIM_CHANNEL _X, thus specifying the comparison for each TIM1 channel. The three main and complementary channels of TIM1 must be enabled to obtain the six signal outputs for inverter control. This is done in the Peripheral Initialization section (initialize all channels are enabled using the HAL_TIM_PWM_Start macro. The additional outputs are enabled with the HAL_TIMEx_PWMN_Start macro with their respective TIM1 and channel addresses, as shown in Fig. 13.

```
  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_TIM1_Init();
  MX_TIM3_Init();
  /* USER CODE BEGIN 2 */
  HAL_TIM_PWM_Start(&htim1, TIM_CHANNEL_1);
  HAL_TIMEx_PWMN_Start(&htim1, TIM_CHANNEL_1);
  HAL_TIM_PWM_Start(&htim1, TIM_CHANNEL_2);
  HAL_TIMEx_PWMN_Start(&htim1, TIM_CHANNEL_2);
  HAL_TIM_PWM_Start(&htim1, TIM_CHANNEL_3);
  HAL_TIMEx_PWMN_Start(&htim1, TIM_CHANNEL_3);
```

Figure 13. Enabling the PWM outputs in the TM32CubeIDE.

VI. RESULTS

A. The Experimental Setup

The algorithm was programmed using a laptop computer and once completed and compiled, it was uploaded to the MCU via a USB port. After this, the laptop is no longer required. Since the voltage inverter requires $15V_{dc}$ CMOS logic signals to trigger the IGBTs, and since the development board cann ot supply logic signals at these voltage levels, a circuit was implemented to convert the 3.3 V_{dc} output signal from the MCU to signals of $15V_{dc}$. For this purpose, a transistor-based IC operating in cutoff and saturation modes has been selected.

For the power stage, a commercially available three-phase, 2-L inverter by Semikron (SKiiP 232GD120-313CTV) and a 1 hp squirrel-cage induction motor have used. To power the inverter with high DC voltage, the single-phase line voltage was rectified and filtered using power capacitors.

A 500 MHz, 4-channel LeCroy HDO6054 oscilloscope was used to capture the waveforms. Fig. 14 shows a photograph of the experimental setup.

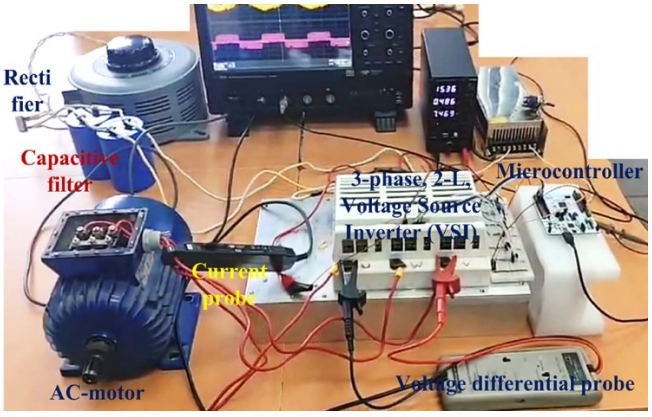

Figure 14. The experimental setup.

A. Experimental Results

Figure 15 displays two experimental trigger signals for one branch of the three-phase inverter, in order to show the programmed dead-time, which in this work was set to 2 µs.

Figs. 16 a) and 16 b) show the logic signals for triggering the inverter when the $\underline{u}_s$* is in sectors I, and in sector III, respectively. These signals are measured at the MCU's outputs PA8, PA9, and PA10 using the oscilloscope to verify the SVM's switching pattern.

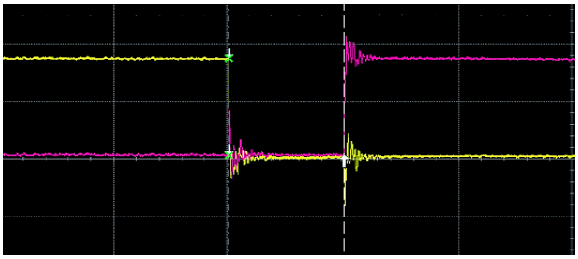

Figure 15. The programmed dead-time, [2 us/div], [2 V/div].

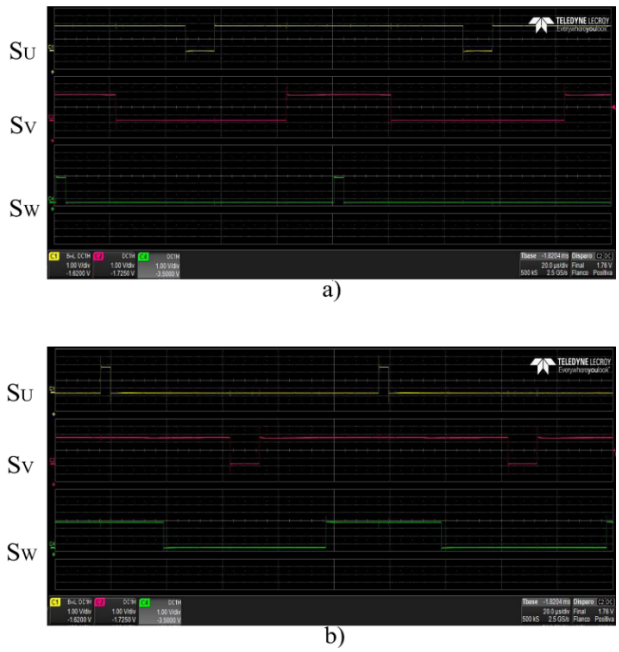

Figure 16. SVM trigger signals, [20 us/div], [1 V/div]. a) $\underline{u}_s$* is in sector I, b) $\underline{u}_s$* is in sector III.

Fig. 17 shows the line-to-line voltage V_{UW}, and the current of the phase U with half load. These measurements were taken at the input of the AC motor. Fig. 18 shows a zoomed-in view of Fig. 18, to appreciate the modulated voltage and the low THD that the phase current presents.

Fig. 19 shows the phase-to-line voltage V_{UN}, and the current of phase W, when the motor is fully loaded. This demonstrates that the SVM algorithm has been successfully implemented affordably on the low-power, low-cost development board.

VII. CONCLUSIONS

In this paper, the digital implementation of the SVM algorithm for three-phase AC-motor drives is explained. The development board STM32F446RE was used. This microcontroller, besides being affordable and readily available, has been proved that it can be easy and quick to configure and program using its development tools.

A commercial inverter (by Semikron) integrated with IGBTs has been used. It was necessary to implement a board with integrated circuits to convert the logic (TTL) signals to 15 V_{dc} CMOS signals, which is the voltage level required to trigger the inverter. The trigger signals were programmed so that the inverter operates at a frequency of 10 kHz. A 1 hp three-phase squirrel-cage induction motor was connected as the load.

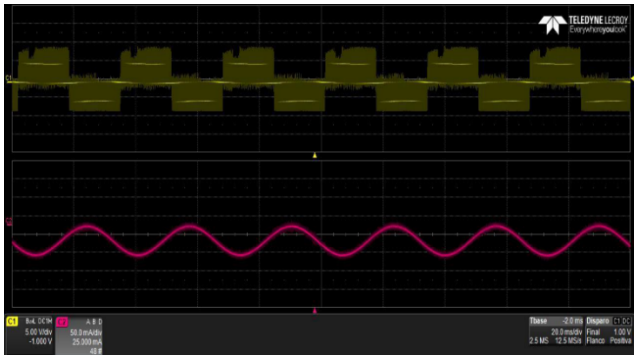

Figure 17. Line voltage Vuw and current of the phase U, [20 ms/div], [50 V/div], [0.5 A/div].

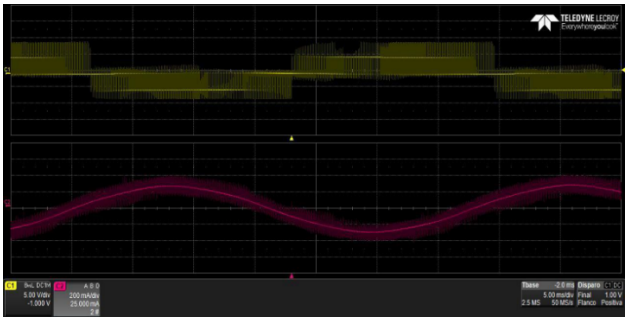

Figure 18. Zoom of Fig. 17, [5 ms/div], [50 V/div], [0.5 A/div].

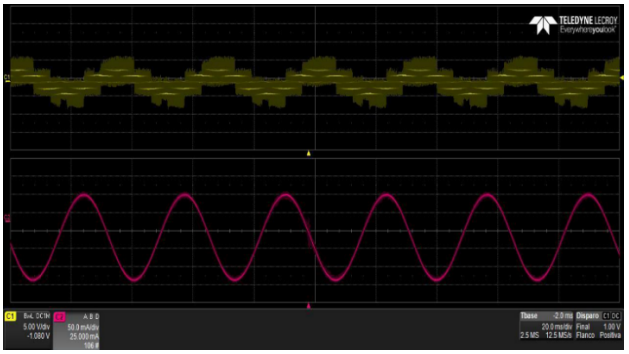

Figure 19. Phase voltage and current of the phase W, [20 ms/div], [50 V/div], [0.5 A/div].

The experimental results show a suitable implementation of the SVM algorithm and excellent performance of the AC drive.

REFERENCES

[1] Technavio "Variable frequency drive (VFD) market size 2025-2029", May 2025. [Online]. Available: https://www.technavio.com/report/variable-frequency-drives-market-industry-size-analysis
[2] B. Bose, "Modern power electronics and AC drives," Prentice H. 2001.
[3] R. Krishnan, "Electric motor drives: modeling, analysis, and control," Prentice Hall. 2001.
[4] T. M. Jahns and E. L. Owen, "AC adjustable-speed drives at the millennium: how did we get here?" *IEEE Trans. Power Electron.*, vol. 16, no. 1, pp. 17-25, Jan. 2001.
[5] B. Batkhishig, P. F. D. Gonçalves, G. Pietrini, B. Nahid-Mobarakeh and A. Emadi, "PWM techniques for two-level voltage source inverters: a comparative study," *IEEE Access*, vol. 13, pp 86235-86255, Apr. 2025.

[6] K. Zhou and D. Wang, "Relationship between space-vector modulation and three-phase carrier-based PWM: a comprehensive analysis," *IEEE Trans. Ind. Electron.*, vol. 49, no. 1, pp. 186-196, Feb. 2002.
[7] Rashid, M. H. "Power electronics Handbook. devices, circuits, and applications," 3rd ed. Elsevier: Pensacola, FL, USA, 2015.
[8] J. P. Moreno Beltran, O. Sandre Hernandez, R. Morales Caporal, P. Ordaz-Oliver and C. Cuvas-Castillo, "Model predictive torque control of an induction motor with discrete space vector modulation, " *17th Int. Conf. Electr. Eng., Comput. Sci. Autom. Control (CCE)*, Mexico City, Mexico, 2020, pp. 1-6 Nov. 2020.
[9] F. A. Ramirez, M. A. Arjona and C. Hernandez, "A space-vector PWM voltage-source inverter for a three-phase induction motor based on the dsPIC30F3011," in Proc. *6th Electron., Robotics and Automot. Mechs. Conf., (CERMA)*, Cuernavaca, Mexico, 2009, pp. 429-434, Sep. 2009.
[10] R. Morales-Caporal, O. Sandre-Hernández, E. Bonilla-Huerta, J. C. Hernández-Hernández and J. J. Hernández-Mora, "DSP-based space vector modulation for a VSI-fed permanent magnet drive," *9th Electron., Robotics and Automotive Mechs. Conf., (CERMA),* Cuernavaca, Mexico, 2012, pp. 261-266, Nov. 2012.
[11] F. J. Bartos, "AC drives stay vital for the 21st century," *Control Engineering,* Sep. 2004.
[12] S. Ogasawara, H. Akagi, and A. A. Nabae, "A novel PWM scheme of voltage source inverters based on space vector theory," *Archiv f. Elektrotechnik,* vol. 74, p.p. 33–41, Jan. 1990.
[13] J. Holtz, "Pulsewidth modulation for electronic power conversion," in *Proc. of the IEEE*, vol. 82, no. 8, pp. 1194-1214, Aug. 1994.
[14] Y. Solbakken "Space vector PWM intro", May 2017. [Online]. /www.switchcraft.org/learning/2017/3/15/space-vector-pwm-intro
[15] A. Y. Yousef, "Space vector pulse width modulation technique," *IJETCSE*, vol. 15, no. 1, pp. 159-165, May 2015.
[16] STM32 Nucleo-64 development board with STM32F446RE MCU. [Online]. Available: https://www.technavio.com/report/ variable-frequency-drives-market-industry-size-analysis
[17] STM32CubeMx. [Online]. Available: https://www-st-com.translate.goog/en/development-tools/stm32cubemx.html?_x_tr_sl=en&_x_tr_tl=es&_x_tr_hl=es&_x_tr_pto=tc
[18] STM32CubeIDE. [Online]. Available: https://www.st.com/en/development-tools/stm32cubeide.html
[19] STM32 3 Phase PWM (Center-Aligned). [Online]. Available: https://deepbluembedded.com/stm32-3-phase-pwm-center-aligned-example-code/
[20] How to generate dead time in complementary PWM signals using STM32's advanced timer peripheral. [Online]. Available: https://gist.ly/youtube-summarizer/how-to-generate-dead-time-in-complementary-pwm-signals-using-stm32s-advanced-timer-peripheral
[21] Space vector modulation using 8-bit ST7MC Microcontroller. Application note. [Online]. Available: file:///C:/Users/rmcap/Downloads/cd00055518-space-vector-modulation-using-8bit-st7mc-microcontroller-and-st7mckitbldc-starter-kit-stmicroelectronics-5.pdf

ROBERTO MORALES-CAPORAL received the B.E. degree in Electromechanical Engineering from the National Technological Institute of Mexico–Campus Apizaco (TecNM-ITA), Apizaco, Mexico, in 1999, the M.Sc. degree in Electrical Engineering from the Graduate and Research Department, Superior School of Mechanical and Electrical Engineering (ESIME-Z), National Polytechnic Institute (IPN), México City, Mexico, in 2001, and the Dr.-Ing. degree in Electrical Engineering from the University of Siegen, Siegen, Germany, in 2007. From 2001 to 2003, he was a Lecturer with the Interdisciplinary Professional Unit in Engineering and Advanced Technologies (UPIITA), IPN.

He is currently a full-time Professor at the Division of Graduate Studies and Research of the TecNM-ITA. His research interests include discrete-time control systems, predictive control of power converters, AC-motor drives, hardware development and IoT.

Prof. Morales-Caporal is a member of the National Research Fellows System Level 2 (SNII-2), Secretariat of Science, Humanities, Technology and Innovation (SECIHTI), México.

DIEGO AGUAYO-DIAZ received the title of Electronic Engineer with a specialization in control and automation in industry 4.0 from the National Technological Institute of Mexico-Campus Apizaco (TecNM-ITA), Apizaco, Mexico, in 2026.

His areas of professional interests include Industry 4.0, power electronics and solutions using microcontrollers.

RAFAEL ODOÑEZ-FLORES received the B.E. degree in Electronics Engineering from the Instituto Tecnologico de Puebla (TecNM-ITP), Puebla, Mexico, in 1995, the M.Sc. degree in Electronics Engineering from the National Center for Research and Technological Development (CENIDET), Cuernavaca, Mexico, in 1998, and the D.Sc. degree in Electrical Engineering from the University of Paris 11, Higher School of Electricity (SUPELEC), Orsay, France, in 2007.

He is currently a full-time Professor with the Division of Graduate Studies and Research, ITA. His research interests include control of power electronics, electrical energy quality, renewable energy, and magnetic induction heating.

MARIO E. LEAL-LOPEZ received the B.E. and the M.Sc degrees both in Electronic Engineering from the National Technological Institute of Mexico–Campus Orizaba (TecNM-ITO), Orizaba, Mexico, in 2006 and 2010, respectively. He also holds a specialization in Mechatronics Technology from CIDESI, Querétaro, Mexico.

He is currently a full-time Professor with the Division of Graduate Studies and Research of the TecNM–Campus Apizaco. He has served as head of the Department of Electrical & Electronic Engineering and of the Division of Graduate Studies and Research. His research interests include control of mechatronic systems, robotics, land and air vehicles, and automation of methane gas quantification anaerobic biodigesters.

EDMUNDO BONILLA-HUERTA received the B.Sc. and M.Sc. degrees both in Computer Science from the National Technological Institute of Mexico–Campus Apizaco (TecNM-ITA), Apizaco, Mexico, in 1994 and 1996, respectively, and the Ph.D. degree in Computer Science from the University of Angers, Angers, France, in 2008.

He is currently a full-time Professor with the Division of Graduate Studies and Research of the TecNM-ITA. His research interests include optimization, fuzzy logic, machine learning, techniques for microarray data analysis, and bioinformatics.

Prof. Bonilla-Huerta is a member of the National Research Fellows System Level 1 (SNII-1), Secretariat of Science, Humanities, Technology and Innovation (SECIHTI), México.