UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIEROS INDUSTRIALES VALENCIA

TRABAJO FIN DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

DESARROLLO DE UN SISTEMA DE CONTROL DE INVERSOR BASADO EN MICROCONTROLADOR PARA UN BANCO DE ENSAYOS DE ACCIONAMIENTOS ELÉCTRICOS

AUTOR: JORGE MIRALLES GUIMERÁ

TUTOR: JAVIER ANDRÉS MARTÍNEZ ROMÁN

COTUTOR: ÁNGEL SAPENA BAÑO

Curso Académico: 2018-19

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIEROS INDUSTRIALES VALENCIA

DESARROLLO DE UN SISTEMA DE CONTROL DE INVERSOR BASADO EN MICROCONTROLADOR PARA UN BANCO DE ENSAYOS DE ACCIONAMIENTOS ELÉCTRICOS

MEMORIA

AUTOR: JORGE MIRALLES GUIMERÁ

TUTOR: JAVIER ANDRÉS MARTÍNEZ ROMÁN

COTUTOR: ÁNGEL SAPENA BAÑO

Curso Académico: 2018-19

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

1. ANTECEDENTES

Los motores de inducción han sido y siguen siendo los más utilizados en la industria por su bajo coste económico, su durabilidad, su robustez, la posibilidad de conectarse directamente a la red y su buen rendimiento. Estas características hacen que este motor haya sido utilizado durante años para todo tipo de aplicaciones.

Sin embargo, con alimentación a frecuencia de red, este motor no podía ser utilizado fácilmente en aplicaciones en las que el motor requiriera realizar cambios en su velocidad. Y pese a que se podían realizar cambios en la velocidad de operación variando el número de polos del motor, no se podía hacer girar a una velocidad ajustable con buena resolución sino sólo cerca de dos o tres velocidades predefinidas.

En un principio, la variación de velocidad de estos motores se realizaba con distintas técnicas capaces de aumentar el deslizamiento del rotor, como son la disminución de la tensión de los devanados o el aumento de la resistencia interna del rotor. Sin embargo, estas técnicas tenían la desventaja de que disminuían en gran medida el rendimiento del motor.

Este problema desaparece con la mejora de la electrónica y la aparición de los variadores de frecuencia, capaces de generar las tres señales que requiere el motor a frecuencia variable (la necesaria para cambiar la velocidad de giro del campo en el entrehierro o velocidad de sincronismo), pudiendo esta ser gobernada por el usuario de forma sencilla y conociendo su valor a través de una pantalla. Es a partir de este momento cuando estos motores se pueden utilizar para todo tipo de aplicaciones aunque requieran variaciones en la velocidad de operación, manteniendo un rendimiento elevado, todo ello a un precio económico debido a los avances en electrónica que han hecho que el precio de estos dispositivos haya bajado drásticamente en estos años.

Fruto de los avances en electrónica de potencia han aparecido en el mercado convertidores electrónicos para el desarrollo de prototipos altamente configurables y que permiten un control externo sencillo, por un precio asequible facilitando la realización de proyectos de investigación y desarrollo con motores y estos equipos.

Los avances desarrollados en el campo de la microelectrónica que han permitido la irrupción de los microcontroladores en el mercado con precio inferior a los 5€ y que pese a su bajo coste permiten disponer de sistemas de control con entradas y salidas analógicas y digitales, comunicaciones inalámbricas y cableadas y los requerimientos necesarios en términos de frecuencia de reloj, tiempos de operación en coma flotante y memoria RAM y flash para integrar el control de un accionamiento eléctrico.

2. OBJETIVOS

En el desarrollo de este trabajo se persigue desarrollar un sistema de control de un motor de inducción desde el teléfono móvil implementando varias opciones de control y con la posibilidad de monitorización en el Smartphone de parámetros del motor. Para ello, se deben alcanzar una serie de objetivos, que son los siguientes:

- Desarrollo e implementación en microcontrolador de un sistema de modulación de ancho de impulso para la operación del convertidor electrónico a partir de las consignas de tensión (vector espacial de tensión).

- Desarrollo e implementación de una interfaz inalámbrica de comunicación y control entre el sistema de control del accionamiento y un dispositivo Android que hace la función de interfaz de usuario para operar el motor.

- Desarrollo e implementación en microcontrolador de dos sistemas de control del accionamiento:

    - Control vectorial de velocidad en lazo abierto.
    - Control vectorial del grado de carga.

- Desarrollo de una aplicación Android con las funciones de interfaz de usuario para el control del accionamiento incluyendo recepción y envío al sistema de control de los comandos de operación y la monitorización las variables del accionamiento en el dispositivo móvil.

Es necesario comentar que este TFG se encuentra integrado dentro de un proyecto más amplio de desarrollo de un banco de ensayos para el prototipado rápido de sistemas de control y operación de accionamientos eléctricos. Este banco de ensayos se ha diseñado para ser utilizado en las asignaturas de Ampliación de Máquinas Eléctricas y de Análisis Dinámico y Control de Accionamientos Eléctricos del Máster en Ingeniería en Tecnologías Industriales e incluye desarrollos en otras dos áreas. La primera de ellas es la realimentación de posición y velocidad para poder implementar el control vectorial de velocidad en lazo cerrado y en carga en lazo abierto, mientras que la segunda de ellas es la realimentación de corriente para la implementación del control de campo y carga en lazo cerrado.

3. DESCRIPCIÓN DE LA MEMORIA

En este capítulo se va a comentar los distintos temas que se abordarán en cada uno de los capítulos que aparecerán a lo largo de la memoria de este documento resaltando los puntos más importantes de cada uno de ellos.

En el primer capítulo que aparecerá a continuación se explicará la técnica y el algoritmo SVPWM y se razonará el porqué de la elección de esta técnica frente a la técnica PWM por portadora triangular. Además, se explica el algoritmo descentrado y cuáles son las razones para seleccionar este algoritmo frente al algoritmo clásico.

En el siguiente capítulo se procede a realizar una descripción detallada del montaje que se va a utilizar, tanto de la parte de potencia como de la parte de control resaltando las características más importantes y explicando las razones de su elección, realizando una comparación de las alternativas en lo relacionado al microcontrolador.

En el capítulo número seis se comienza justificando el uso del control de velocidad basado en la relación tensión-frecuencia constante, o control VF, explicando detalladamente el código de programación del microcontrolador, comentando cual es la función de ellos, fragmentos de código más significativos. A continuación se detallan las pruebas realizadas para la comprobación del correcto funcionamiento del código de acuerdo con los objetivos de control VF establecidos y los resultados obtenidos con ellas así como las primeras pruebas de funcionamiento del motor. Una vez hecho esto se procede a justificar la elección del uso de una interfaz de comunicación por Bluetooth con el dispositivo móvil que se va a utilizar como panel de operador y a detallar el código del microcontrolador y sus funciones, así como las modificaciones al código anterior para el correcto funcionamiento del mismo. Por último se muestra la parte de código relacionada con el control de la pantalla LCD para visualización local explicando las funciones del mismo.

A continuación de este capítulo se procede a introducir el control en carga explicando las razones para su inclusión en este proyecto y cuál es la estrategia para implementarlo en el microcontrolador, así como la argumentación de la necesidad de obtener medidas de posición y velocidad del eje del motor. Una vez hecha esta introducción se sigue con el desarrollo del código empleado y su explicación, para después proceder a comentar los errores que se detectaron al realizar las primeras pruebas y la manera de solventarlos, tanto para los primeros ensayos en vacío como para los ensayos en carga, comparando los resultados obtenidos antes y después de la corrección de los errores detectados.

En el siguiente apartado se realiza una explicación detallada de los ensayos en carga a distintas velocidades y grados de carga mostrando gráficas de voltaje y corriente antes y después del filtrado de las mismas, así como resultados de la pantalla de la aplicación

y del analizador de energía utilizado para medir tensión, corriente y potencias activa y reactiva consumidas por la máquina. Para finalizar este apartado se muestra una comparación fruto del análisis de los distintos ensayos realizados de ocho parámetros distintos para 3 grados de carga y tres velocidades distintas, mostrando los resultados obtenidos en forma de gráfica para una interpretación más cómoda.

Con la finalización del capítulo de los ensayos empieza el relacionado con la creación de la aplicación Android donde se explica la necesidad de creación de la misma, sus funciones básicas y el lenguaje empleado para su implementación. A continuación, se empieza mostrando la interfaz de usuario explicando las pantallas que la componen y la forma de operación de cada una. Por último se muestra la parte lógica de la aplicación exponiendo las partes clave del código de la aplicación y realizando una explicación detallada de las mismas.

En el penúltimo capítulo se procede a exponer las conclusiones extraídas de la realización del trabajo resaltando el aprendizaje intenso que se ha realizado como consecuencia de este trabajo y las implicaciones que el mismo tiene. La memoria finaliza mostrando las referencias bibliográficas que se han consultado durante la redacción de este Trabajo Final de Grado.

4. INTRODUCCIÓN AL SVPWM

La técnica SVPWM (Space Vector Pulse Width Modulation, modulación de ancho de impulso por vector espacial en castellano) es un modo de generar las señales de control para las tres ramas de un inversor trifásico de entrada por tensión constante a partir de la consigna de vector espacial de tensión basándose en la modulación por anchura de pulsos (Pulse Width Modulation, PWM).

En esta técnica se tratan las tres fases como una unidad y se calcula la anchura de los pulsos (conexión al terminal + y – del bus de continua) de las tres fases usando un único modulador para todas y, por tanto, teniendo en cuenta la interacción entre ellas, a diferencia de otras técnicas como la PWM basada en portadora triangular que calcula cada una de las consignas de las tres fases por separado.

El sistema SVPWM, por lo tanto, se integra entre el sistema de control del accionamiento y el inversor trifásico de entrada por tensión constante: toma como entrada la consigna de vector espacial de tensión que produce el sistema de control del accionamiento y realiza los cálculos necesarios para obtener los ciclos de trabajo durante el siguiente periodo de modulación PWM de forma que el vector espacial de tensión medio aplicado a la salida del inversor durante ese periodo de modulación coincida con el valor de consigna recibido.

Con esta idea en mente, para probar el algoritmo, lo que se hace es crear un vector espacial de amplitud constante rotando en el plano αβ (sistema de referencia fijo, es decir, ligado al estator) a velocidad también constante, el cual actuará como consigna para, de forma similar a como ocurre en la modulación PWM por portadora triangular trifásica, generar, en régimen permanente, tres señales senoidales desfasadas 120º en el tiempo.

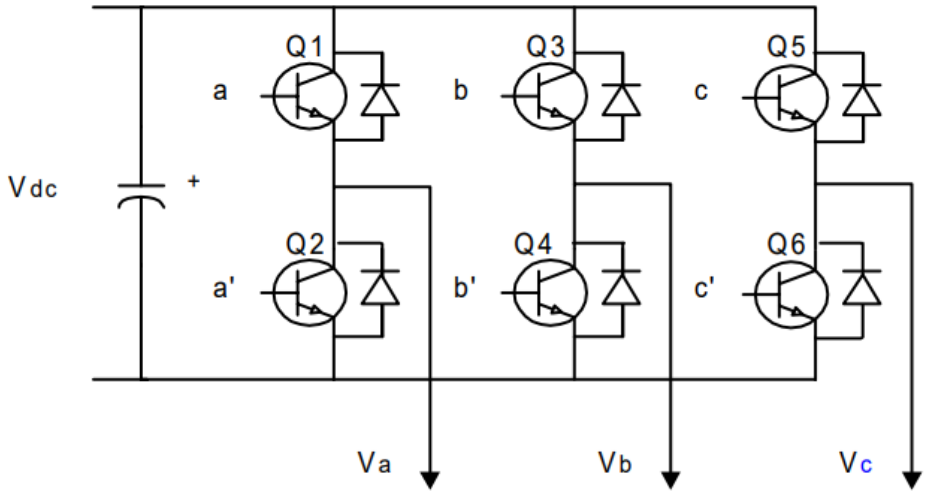


Figura 1. Inversor trifásico

Al tratarse de un inversor trifásico se podrá actuar sobre las señales de conmutación a, b y c, que podrán tener dos estados: 0 o 1, según estén a nivel alto o bajo respectivamente. Las señales a', b' y c' serán las negadas de a, b y c, de forma que se evite que los dos transistores de una misma rama del inversor permitan pasar la corriente simultáneamente provocando un cortocircuito entre los bornes de la fuente de continua. El propio inversor es habitual que incorpore, como es nuestro caso, tal y como muestra el manual del módulo inteligente de control de potencia STGIPS20K60 en su apartado número 3.1 Dead Time, un tiempo muerto de separación entre la desactivación de la señal de control de la parte positiva de la rama y la activación de la negativa, y viceversa. La tensión fase neutro, de forma simplificada, considerando que se trata de una trifásica equilibrada, se puede calcular a partir de las señales de conmutación:

$$\begin{bmatrix} V_a \\ V_b \\ V_c \end{bmatrix} = \frac{1}{3} V_{dc} \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

Ecuación 1. Tensión fase-Tensión generada

Por tanto, los estados posibles para el inversor serán:

	a	b	c	Va	Vb	Vc
V_0	0	0	0	0	0	0
V_1	1	0	0	2/3Vdc	-1/3Vdc	-1/3Vdc
V_2	1	1	0	1/3Vdc	1/3Vdc	-2/3Vdc
V_3	0	1	0	-1/3Vdc	2/3Vdc	-1/3Vdc
V_4	0	1	1	-2/3Vdc	1/3Vdc	1/3Vdc
V_5	0	0	1	-1/3Vdc	-1/3Vdc	2/3Vdc
V_6	1	0	1	1/3Vdc	-2/3Vdc	1/3Vdc
V_7	1	1	1	0	0	0

	V_0	V_1	V_2	V_3	V_4	V_5	V_6	V_7
α	0	V_{dc}	$0{,}5V_{dc}$	$-0{,}5V_{dc}$	$-V_{dc}$	$-0{,}5V_{dc}$	$0{,}5V_{dc}$	0
β	0	0	$\sqrt{3}/2V_{dc}$	$\sqrt{3}/2V_{dc}$	0	$-\sqrt{3}/2V_{dc}$	$-\sqrt{3}/2V_{dc}$	0
$\lvert V_i \rvert$	0	V_{dc}	V_{dc}	V_{dc}	V_{dc}	V_{dc}	V_{dc}	0
arg	-	0º	60º	120º	180º	240º	300º	-

El desarrollo de la tabla anterior aparece con más detalle y con información ampliada en el apartado 3.4 de los Apuntes de Ampliación de Equipos e Instalaciones Eléctricas [12].

Por ello, se dispone de un total de 8 estados diferentes en el inversor, donde 6 son estados activos (con tensión en las fases) y 2 son estados nulos. Partiendo de esta idea, se pueden dibujar los 6 estados activos en el espacio de forma similar a la representación de los vectores trifásicos, resultando un hexágono en el que se indican los vectores espaciales que corresponden a las diferentes combinaciones binarias abc posibles:

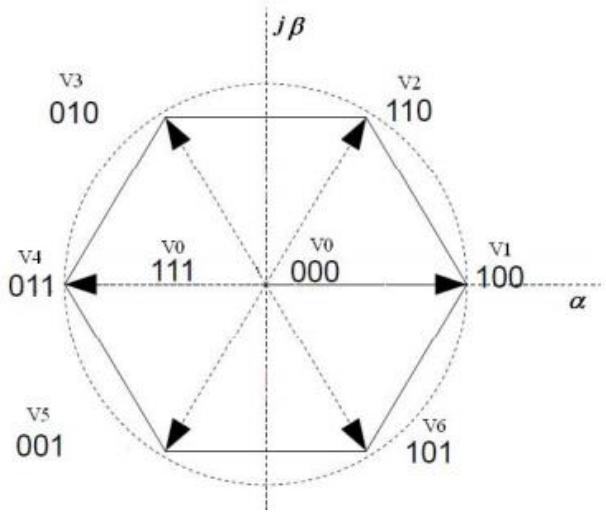


Figura 2. Sectores en la técnica SVPWM

Ahora lo que se quiere obtener es un vector espacial determinado (con una cierta orientación y módulo, establecidos por el sistema de control del accionamiento) combinando los distintos estados que nos permite el inversor, como se ha explicado anteriormente. Esta técnica utiliza los dos estados más próximos al vector espacial que quiere generar y los dos estados nulos para, por combinación lineal, obtener un vector espacial de salida igual al consignado. Para ello se divide el plano complejo en seis sectores triangulares donde la consigna para cada una de las fases del inversor se calculará de forma distinta y donde se cambiará la secuencia de activación de los transistores dependiendo del sector en el que se halle el vector espacial. El motivo de esta segmentación del plano complejo es conseguir que la combinación lineal, al utilizar los dos vectores fijos (los dos radios del hexágono) más próximos al vector espacial de consigna, proporcione términos positivos siempre que el vector espacial de consigna tenga su afijo en el interior del sector.

Para hacer el cálculo de los tiempos de apertura y cierre de los transistores se toma un vector cualquiera del primer sector (entre 0 y 60º). Su fórmula vendrá dada por:

$$\overrightarrow{V^*} = \frac{T_1}{T_s}\vec{V}_1 + \frac{T_2}{T_s}\vec{V}_2 + \frac{T_0}{T_s}\vec{V}_0 + \frac{T_7}{T_s}\vec{V}_7$$

Ecuación 2. Cálculo del vector espacial

Donde T_0 ,T_1 …..T_7 es el tiempo que pasa cada uno de los estados activados y donde T_s es el tiempo de conmutación o de cada ciclo de transistores.

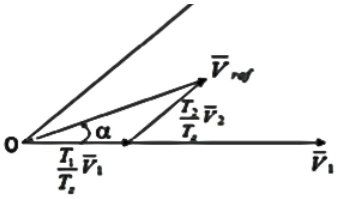


Figura 3. Descomposición del vector

Al elegir un vector cualquiera del primer sextante para el cual se quieren obtener los tiempos de apertura y cierre de transistores determinado por el índice de modulación M = Vref/Vdc (la modulación de voltaje que queremos realizar) y su orientación α en el plano

complejo quedan definidos los tiempos de aplicación de los vectores V_1 y V_2 obligando a que la combinación lineal de ambos coincida con el vector espacial de consigna:

$$V_{ref} = \frac{T_1}{T_z} \times V_1 + \frac{T_2}{T_z} \times V_2$$

Ecuación 3. Cálculo del vector espacial

Aplicando la ley de senos a la anterior figura y resolviendo las ecuaciones se obtiene:

$$T_1 = \frac{\sqrt{3} \cdot T_z \cdot \left|\overline{V}_{ref}\right|}{V_{dc}} \left(\sin \frac{n}{3}\pi - \alpha \right) \quad T_2 = \frac{\sqrt{3} \cdot T_z \cdot \left|\overline{V}_{ref}\right|}{V_{dc}} \left(\sin \left(\alpha - \frac{n-1}{3}\pi \right) \right) \quad T_0 = T_z - T_1 - T_2$$

Ecuación 4. Tiempos de apertura de los transistores

Que es la fórmula general para los distintos sectores donde n va de 1 a 6, según en el sector en que se encuentre el vector espacial y donde T_1 y T_2 se calculan como aparece en la ecuación anterior en los sectores impares (1, 3 y 5), sin embargo la forma de calcular los tiempos T_1 y T_2 cambia para los sectores pares(2, 4 y 6), donde las ecuaciones de cálculo de T_1 y T_2 se invierten, utilizando para T_1 la de T_2 y viceversa. Donde los sectores del hexágono quedan definidos por: sector 1 entre 0 y 60º, sector 2 entre 60ºy 120º y así sucesivamente.

La división entre V_{ref} y V_{dc} es la modulación de voltaje y en las siguientes páginas se hará referencia a este valor multiplicado por √3 como mag, valor que variará entre 0 y 1 para realizar la modulación de voltaje en el inversor.

En la siguiente figura se muestra la secuencia de apertura de transistores dependiendo del sector en que se encuentre:

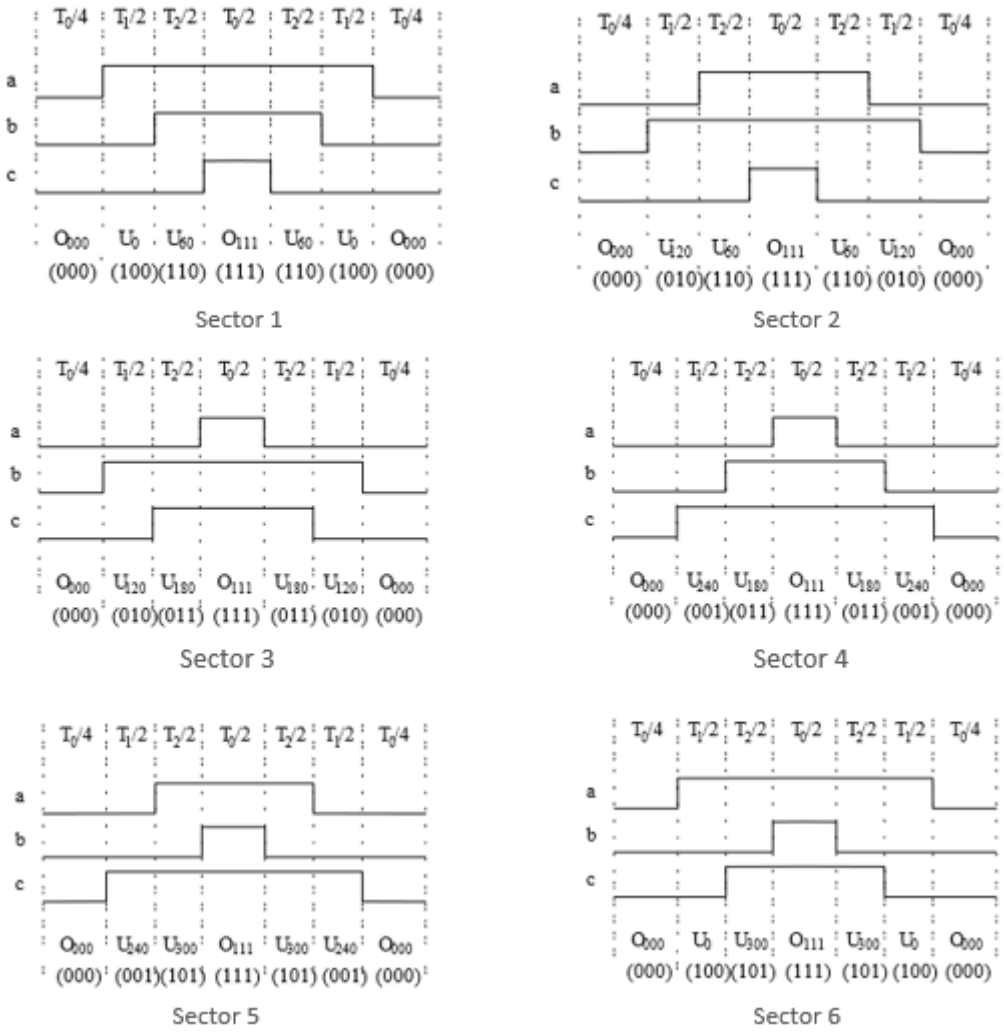


Figura 4. Apertura de transistores según el sector

Generando un ángulo α que varía linealmente con el tiempo es posible crear un vector espacial de módulo constante y giratorio a velocidad constante y, aplicando estas fórmulas, teniendo en cuenta en que sector está, se pueden generar tres señales PWM que aplicadas al inversor, proporcionarán, en cada periodo de modulación, un vector espacial de salida promedio igual al consignado.

La SVPWM tiene la ventaja de consegir generar senoidales de un valor de voltaje mayor frente a otras como la PWM por portadora triangular. Mientras que la primera es capaz de generar una senoidal de valor máximo igual a $V_{dc}/\sqrt{3}$, la segunda solo puede hacerlo de un valor de $V_{dc}/2$, lo cual supone aproximadamente un 15% más de voltaje aprovechable por la máquina a la que conectemos el inversor.

4.1. Variaciones al algoritmo del SVPWM

A la hora de generar las señales se ha cambiado el patrón en que se produce la apertura de los transistores. En el SVPWM original se controlaba que la señal que se generara fuese simétrica para evitar la aparición de algunos armónicos, sin embargo, se decidió descentrar la señal y empezar los flancos de subida al principio, con tres objetivos:

- Simplificar el cálculo de los patrones de cada fase mediante el microcontrolador.
- Reducir el número de conmutaciones de las ramas del inversor.
- Reducir la tensión media del neutro y de las fases respecto de tierra.

El microcontrolador tiene internas una serie de funciones que generan una señal PWM de frecuencia constante y anchura de pulso variable. Estas se utilizan para generar las tres señales de apertura y cierre de los transistores, ya que lo realmente importante es el valor medio de cada una de las señales de los transistores y, por tanto, no tenía tanta importancia el hecho de que estas fuesen simétricas ya que esto únicamente afecta al contenido en armónicos de la tensión generada.

Esto libera mucho al dispositivo, ya que con esta variación únicamente controla el tiempo de conmutación y mandar las dos consignas de amplitud o índices de modulación al inversor por los 2 canales activos en el sector mientras que, con el algoritmo original, además de calcular los tiempos de apertura de cada uno de los transistores, tenía que controlar los tiempos de las siete partes diferentes que se aprecian en la Figura 4 para dar las órdenes de activación y desactivación de cada uno de los transistores. Dicho de otra forma, con el patrón clásico de SVPWM, el microcontrolador debe, en cada ciclo de modulación, al principio, calcular los instantes de tiempo de conmutación de cada rama y, una vez calculados, supervisar continuamente el tiempo transcurrido del ciclo, compararlo con el siguiente punto de conmutación y, si se ha alcanzado, ordenar la conmutación de la rama del inversor asociada.

Por el contrario, al utilizar las rutinas internas de generación de patrones PWM individuales, uno por cada rama del inversor, se aprovechan los elementos integrados en el propio microcontrolador (control de interrupciones), haciendo así más eficiente su operación. Por tanto, al utilizar la alternativa elegida, queda mucho más tiempo libre en el dispositivo dentro de cada periodo de modulación para la realización de los distintos cálculos necesarios para la actualización del ángulo del vector espacial y los cálculos de las consignas para el ángulo del vector espacial y se deja margen de mejora con cálculos adicionales para poder implementar sistemas de control más complejos en el futuro.

Por otra parte, una vez se ha decidido no utilizar un patrón simétrico, no tiene sentido mantener el reparto del tiempo de vector espacial nulo, t_0 (estados V_0 y V_7), a partes iguales. Por eso se elige un patrón en el que el que t_0 se debe aplicar por entero a los estados V_0 o V_7, evitando así, en cada periodo de modulación, la conmutación de una de las ramas del inversor y por tanto las pérdidas asociadas a dicha conmutación.

Por último, atendiendo al hecho de que, para mejorar la seguridad de las personas, el terminal negativo del bus de continua del inversor está puesto a tierra y al terminal de referencia de la electrónica de control (masa), se ha decidido no utilizar el estado V_7 en el que todas las líneas están a voltaje máximo, y hacer solo uso del estado nulo V_0 en el que todos los transistores están apagados. De esta forma, al utilizar un bus de continua cuyo terminal negativo está puesto a masa, se disminuye el voltaje medio del neutro con respecto a masa, así como el voltaje medio de cada una de las líneas también respecto a masa. Esto conlleva una menor carga para el aislamiento del motor.

Esta es la gráfica de cómo se genera la señal en el sector 1 con las variaciones expuestas anteriormente:

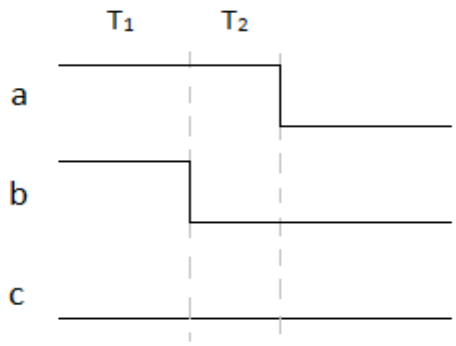


Figura 5. Algoritmo descentrado

5. MONTAJE

Para la realización de este proyecto ha sido necesario un montaje que consta de los siguientes elementos:

5.1. Circuito de potencia

5.1.1. Elementos del circuito de potencia

Los elementos que forman parte del circuito de potencia son: un interruptor automático, un transformador, el inversor, el analizador de energía y el motor trifásico.

- Interruptor automático: Encargado de cortar la corriente si existe un cortocircuito y utilizado como interruptor para cortar la corriente en caso de avería.

- Transformador: Con relación de transformación la unidad, crea un voltaje en el inversor de valor igual que el de la red, pero independiente de ella para así poder colocar la masa del inversor a tierra y la manipulación de los elementos de electrónica de control se realice en condiciones de seguridad frente a contactos indirectos. Se trata de un transformador de 3 KVA y tensiones nominales de 380V/380V Yy y con una corriente de línea nominal de 4,56 A en ambos devanados.

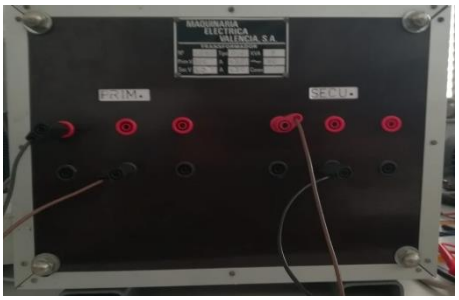


Figura 6. Transformador

- La placa de inversión STEVAL-IHM028V1: Es un convertidor de frecuencia trifásico genérico desarrollado para prototipado rápido de sistemas control de accionamientos eléctricos que permite modificar las aperturas de los transistores mediante un dispositivo externo al disponer de 3 canales habilitados para este fin. Para ello, esta placa rectifica la corriente monofásica alterna para pasarla a continua y activa los transistores siguiendo los comandos recibidos en las entradas de control de rama para generar una salida de potencia controlada por cada una de las señales externas introducidas.

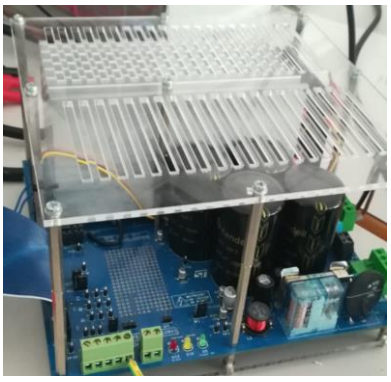


Figura 7. Inversor trifásico

Se trata de una placa para el accionamiento de motores de hasta 2kW que funciona para entrada de voltaje entre 90 y 285 V en alterna y entre 125 y 400 V en corriente continua, esta placa cuenta con un IPM (Intelligent Power Module, en castellano módulo de potencia inteligente) STGIPS20K60, se trata de un circuito que integra los transistores IGBT y el sistema de control de apertura de los transistores, que evita el cortocircuito al variar los dos transistores de una rama del inversor al mismo tiempo. Este sistema es el

encargado de gestionar la apertura y cierre de los transistores evitando el cortocircuito entre los bornes de la fuente de continua.
Esta placa integra también un rectificador y un conjunto de condensadores, para poder realizar la conversión de alterna a continua, protecciones frente a sobretemperaturas e intensidades elevadas y convertidores a 5 y 3,3 V para que resulte sencilla la realización de prototipos con ella.

- Analizador de energía: Es utilizado para medir las tensiones, corrientes y potencias consumidas por el motor. Dependiendo de en qué pantalla se encuentre mostrará unos valores u otros según interese. En este caso se han utilizado las medidas de corrientes y de voltajes introducidas al motor que se proporcionan en amperios y voltios respectivamente, así como las medidas de potencia activa, reactiva y aparente que se muestran en kW, kVAr y kVA. Este analizador está integrado en un sistema con transformadores de corriente 10/1 para conseguir un rango de corriente de entrada de 10A, que junto con los 400V de línea lo hace ideal para realizar medidas de potencia con conexión sencilla en equipos trifásicos de hasta 5kVA.

Figura 8. Analizador de energía

- Motor: Se trata de un modelo específico para uso en laboratorio docente, por lo que alguna de sus características es llamativa por comparación con motores industriales de la misma potencia como la corriente de vacío que es casi igual a la nominal o la presencia de anillos rozantes en el rotor, necesaria para medir las corrientes que circulan por este. Es el motor que vamos a ensayar, cuenta con una potencia nominal de 600W y tensión nominal de 230V.

Figura 9. Motor trifásico

El esquema de conexión del circuito de potencia es el siguiente:

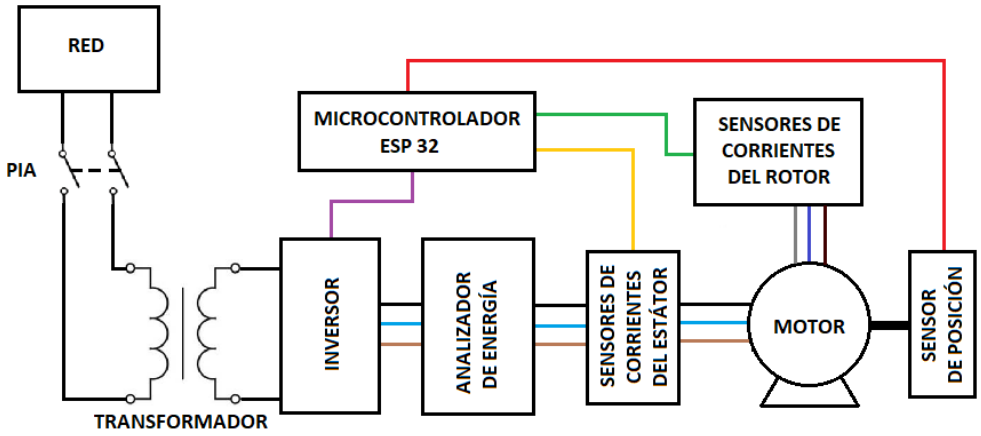


Figura 10. Esquema del circuito de potencia

La parte de realimentación de la posición del motor y de la medida de corrientes se ha realizado en otros dos TFG en los que se desarrollaba el banco de ensayos de forma paralela a éste y cuyo objetivo común era la integración en un único banco de ensayos de utilidad para las asignaturas de Ampliación de Máquinas Eléctricas y de Análisis Dinámico y Control de Accionamientos Eléctricos del Máster en Ingeniería en Tecnologías Industriales.

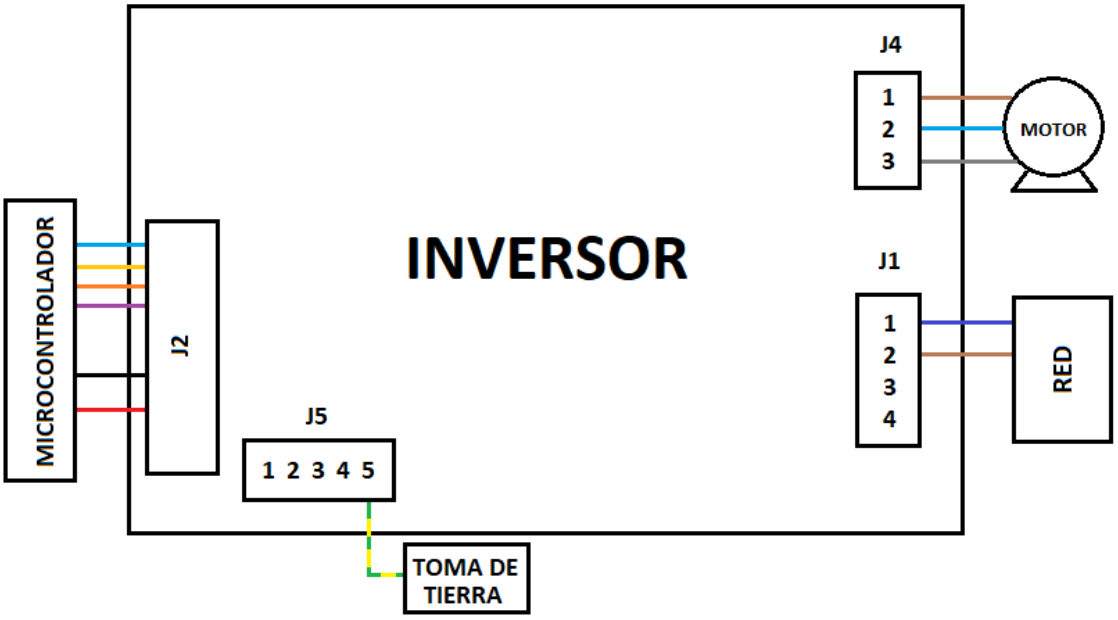


Figura 11. Detalle de conexión en el inversor

En la figura anterior se muestra un esquema de las conexiones del inversor que van a ser utilizadas en el banco de ensayos, las conexiones entre microcontrolador e inversor no aparecen detalladas porque se describirán en el siguiente apartado.

5.2. Circuito de control

El circuito de control consta de un microcontrolador conectado al inversor trifásico que será el encargado de generar las 3 señales de apertura de los transistores incluidos en el inversor. Este dispositivo se manejará vía Bluetooth desde una aplicación Android.

Para realizar el control del inversor se barajó la opción de utilizar un microcontrolador basado en el chip Esp8266 o uno basado en el Esp32, para ello se compararon las siguientes características:

	ESP 8266	ESP 32
Nº núcleos	Single core	Dual core
Velocidad	80Mhz	160 MHz
SRAM	160 kB	512 kB
Wifi	Sí	Sí
Bluetooth	No	Sí (BLE)
GPIO	17	36
GPIO PWM	8	16
ADC	1 (10 bits)	18 (12 bits)
I2C	1	2
SPI	2	4
I2S	1	2
Sensor Touch	No	10
Sensor temperatura	No	Sí
Sensor HALL	No	Sí
Precio	2,72€	4,66€

Por la diferencia de prestaciones que existe entre los dos chips, se elige una placa basada en el ESP 32 ya que ofrece unas características muy superiores por un incremento de precio inferior a 2€. Las características más relevantes que conducen a la elección de esta placa son la presencia de dos núcleos, ya que se quiere utilizar uno para las comunicaciones con la aplicación y la pantalla, la opción de ser controlado por Bluetooth y la existencia de varios conversores analógico-digital necesarios para la medida de corrientes. En especial, la presencia de dos núcleos es fundamental para simplificar el diseño del programa del microcontrolador, ya que se reserva un núcleo para las tareas de control del accionamiento y otro para las de comunicación. Así, las tareas de comunicación, que a veces pueden tener tiempos de espera largos (del orden de ms) no interfieren con el control del accionamiento que, en las versiones desarrolladas, tiene un tiempo de ciclo de 100µs y para el que sería catastrófico dejar un tiempo en blanco, sin control, de varios ms.

El esquema de conexión entre controlador e inversor es el siguiente:

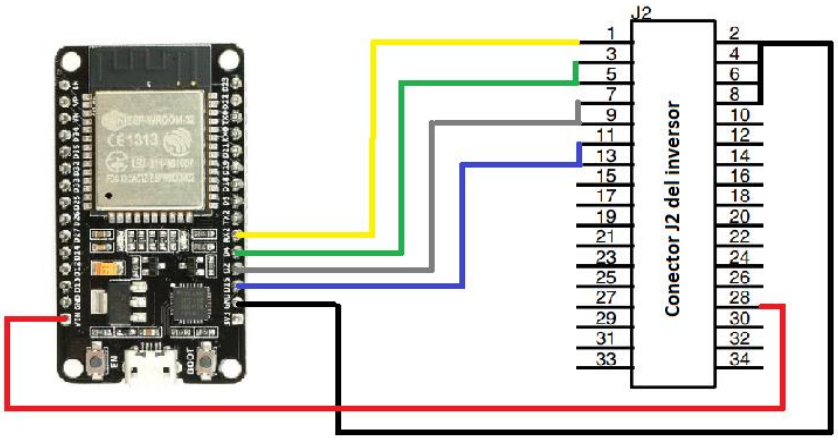


Figura 12. Esquema de conexión de control

En la figura anterior se muestra el esquema de conexión entre el microcontrolador y el inversor, se conecta el GPIO 16 con el pin 1 del conector J2 que es el pin de activación del IPM(Intelligent Power Module), sin la activación de esta señal no tendrán ningún efecto los cambios en las entradas de las tres ramas del inversor. Los pines 3 y 5 del conector J2 son los de activación de los dos transistores de una de las ramas del inversor, estos se pueden conectar de forma conjunta ya que el IPC con el que cuenta la placa es capaz de retardar las aperturas consecutivas de los transistores para evitar los cortocircuitos. De igual forma para los pines 7 y 9 y los pines 11 y 13, respectivamente para las otras dos ramas del inversor. Cada pareja de pines se conecta con una de las salidas del microcontrolador para operar las tres ramas del inversor.

Los pines 2, 4, 6 y 8 van directamente a la masa del inversor y por tanto se deben unir con la masa del microcontrolador, además el pin número 28 del inversor conectado internamente a 5V se debe conectar al pin V_{in} del Esp32, para la alimentación del mismo. Además, requiere poner el conector tipo jumper W1 de la placa entre el pin central y el B en la placa del inversor para que en el pin 28 del conector J2 aparezcan 5V y no 3,3V que alimentarán al microcontrolador por su pin V_{in} y que se utilizan para alimentar los transformadores de efecto Hall para medida de corriente.

5.3. Motor auxiliar

Para realizar los ensayos el motor se conecta mecánicamente a otro motor de mayor potencia conectado a un inversor distinto para poder ajustar las condiciones de funcionamiento cambiando la frecuencia de alimentación del motor auxiliar y así poder operar en carga ajustable el conjunto de motores. Al estar acoplados mecánicamente los dos motores y el motor auxiliar trabajar a una misma velocidad ajustando el nivel de carga al trabajar cada motor con una velocidad de sincronismo distinta asociada a su propia frecuencia de alimentación.

El montaje es el que se muestra a continuación:

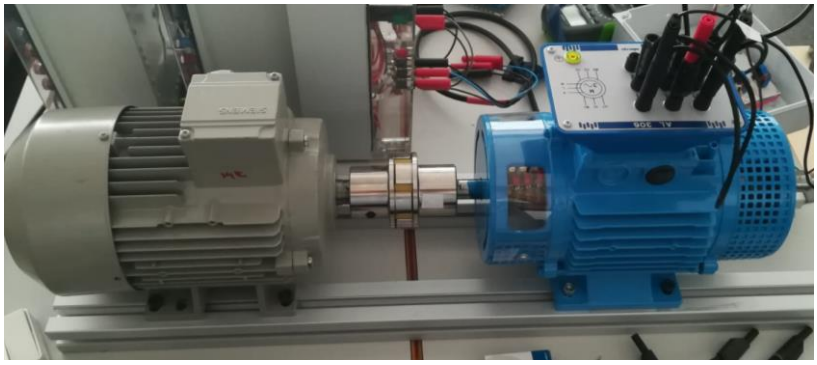


Figura 13. Montaje de los motores

Figura 14. Inversor auxiliar

6. CONTROL DE LA VELOCIDAD DE GIRO DEL MOTOR BASADO EN LA VARIACIÓN DE LA FRECUENCIA

La variación de la frecuencia de alimentación para controlar la velocidad de giro del motor es una solución adoptada en la industria para el control de motores trifásicos. Esto se debe a que el deslizamiento es muy pequeño y por tanto partimos de que existe una relación directa muy aproximada entre frecuencia y velocidad de giro.

Partiendo de esta idea, se decide implementar en primera instancia un control para la velocidad angular del motor basada en la variación de la frecuencia y valor eficaz de la tensión de las 3 fases, controlada mediante el microcontrolador y gestionada a través de una aplicación móvil vía Bluetooth.

Para esta finalidad se utilizó el chip ESP32, que cuenta con Bluetooth y Wifi. Además, también dispone de dos procesadores, uno de los cuales nos servirá para las comunicaciones Bluetooth y con la pantalla LCD local y el otro para los cálculos y la generación de las tres señales PWM. Todo ello está montado en la placa DEVKITV1, que cuenta con los siguientes pines:

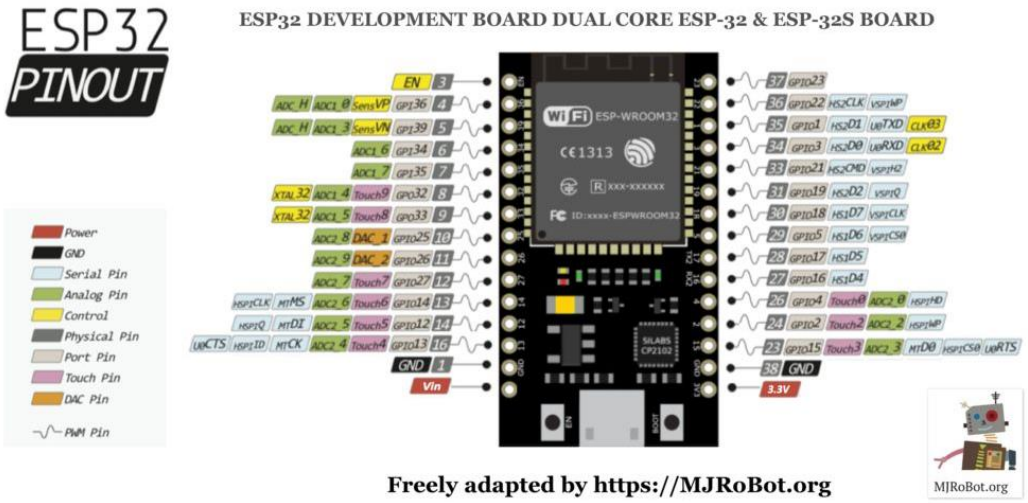


Figura 15. Pines del microcontrolador esp32

El entorno de desarrollo que se ha utilizado para programar el Esp32 es el Arduino IDE. Aplicando las librerías adecuadas para gestionar la placa utilizada, se desarrolló una primera versión de prueba para el control de velocidad variando la frecuencia y la tensión de alimentación. Ésta versión de prueba no utilizaba la comunicación Bluetooth ni se mostraban valores por la LCD para simplificar y acelerar el diseño. En su lugar, se leía la consigna de frecuencia de un potenciómetro y, en una primera etapa, se utilizó un osciloscopio para comprobar la coherencia de los patrones PWM generados para las tres fases con la consigna establecida de velocidad para, así, detectar posibles fallos en

el código antes de conectar las salida PWM del microcontrolador al inversor de potencia y al motor. Esto permitía la generación de las señales de control de las ramas del inversor en condiciones seguras (antes de conectarlo al motor) evitando así una posible avería del inversor o del propio motor.

6.1. Código del control V-F

En esta parte se muestra el código utilizado para programar el microcontrolador, para ello como se ha comentado se utiliza el programa Arduino IDE, el código de este programa sigue la siguiente estructura:

- Declaración de variables globales: Se realiza una declaración de las variables comunes que podrán ser utilizadas por el resto de funciones, se sitúa al principio del código.
- Función setup: Se trata de una función que será ejecutada una sola vez al arrancar el microprocesador, utilizada para inicializar los pines como entradas o salidas, establecer una salida a nivel alto o bajo al inicio y cualquier otra acción que requiera ser ejecutada una sola vez al iniciar el procesador.
- Función loop: Se trata de una función que se repite cíclicamente, utilizada para el desarrollo del código que se repita de forma ininterrumpida y donde se suele ejecutar el grueso de código de la aplicación.
- Funciones auxiliares: Parte del código que se encuentra fuera de la función loop y setup, sirven para crear tareas que se ejecutarán de forma paralela a la función loop o para crear funciones ejecutadas con una llamada desde la función loop (utilizando un enfoque estructurado).

Para este código se ha utilizado un pulsador conectado al microcontrolador de forma que al pulsarse arrancara el código de generación de señales PWM y al producirse otra pulsación del mismo se detuviera. Además, se ha utilizado un potenciómetro que servirá para introducir una consigna de frecuencia de las tensiones de salida al motor, de forma que ajustando el valor del potenciómetro se pueda obtener la frecuencia deseada y por tanto haciendo así que el motor gire a la velocidad consignada. Con este valor se regulará también el índice de modulación ya que la tensión es directamente proporcional a la frecuencia. Por otro lado, es necesario establecer un límite de rampa para evitar incrementos bruscos de corriente al existir deslizamientos elevados en el caso de producirse cambios pronunciados en la frecuencia, por tanto se establece una rampa progresiva para evitar este efecto.

6.1.1. Declaración de variables

```
const int freqcom=10000;//frecuencia de commutacion en Hz
const int resolucion=10; //Resolución AnalogWrite en bits
int const  tcom=100;//tiempo de commutacion en microsegundos
int const mod=1023;//multiplicar el Duty cicle dependindo de resolución
const int R =0;//Canales por los que va a ir cada fase
const int S =1;
const int T =2;
#define botonencendido 13 //Pin para el botón de pruebas
#define salidaencendido 16 //Salida de activación del inversor
#define pinfreq 12 //Pin para el potenciometro
#define Rpin 15//Pines de salida de las señales transistores
#define Spin 2
#define Tpin 4
const int freqmin=5;
const int freqmax=50;
const float trampa=10; //Tiempo tarda alcanzar frecuencia máxima(s)
float a=0;//Ángulo del vector espacial (rad)
int b=0;//Variable para el control de tiempos
float mag=0;//modulación de voltaje
float rampamax;
float freqp=0;//Consigna de frecuencia del potenciómetro
float freq=0;//Frecuencia real que generamos la señal
int estado=0;//Variable para la marcha/paro del motor
```

En esta primera parte, lo que se hace es la declaración de las distintas variables que se van a utilizar durante la ejecución del programa, ya que el software que se está utilizando está basado en el lenguaje de programación C y, por tanto, todas las variables se deben declarar antes de ser utilizadas. Además, están acompañadas de un pequeño comentario explicativo de la función de cada una de ellas.

6.1.2. Función setup

```
void setup() {
  pinMode(Rpin,OUTPUT);
  pinMode(Spin,OUTPUT);
  pinMode(Tpin,OUTPUT);
  pinMode(salidaencendido,OUTPUT);
  pinMode(botonencendido,INPUT);
  pinMode(pinfreq,INPUT);
  ledcSetup(R,freqcom,resolucion);
  ledcAttachPin(Rpin,R);
  ledcSetup(S,freqcom,resolucion);
  ledcAttachPin(Spin,S);
  ledcSetup(T,freqcom,resolucion);
  ledcAttachPin(Tpin,T);
  rampamax=freqmax*1.0/(trampa*1000000/tcom);
}
```

Una vez hecha la declaración de variables, lo que se hace es definir la función setup(), que es una función que se ejecutará una sola vez, y en la que se hace lo siguiente:

- Definir ciertos pines como entradas o salidas mediante la función pinMode(Pin, INPUT/OUTPUT).

- Asociar a 3 canales PWM internos del microcontrolador una frecuencia de conmutación y una resolución (en este caso 10000Hz y 10bits) mediante la función ledcSetup(canal, frecuencia, resolución).

- Asociar cada uno de los tres canales creados a un Pin de la placa para que las señales de cada canal se ejecuten por el Pin deseado con la función ledcAttachPin(Pin, Canal).

- Definir el incremento máximo de frecuencia en cada pasada (rampamax) como la frecuencia máxima dividido entre el número de pasadas que dará el bucle en el tiempo de rampa.

6.1.3. Función loop

Una vez definidos cada uno de los pines, lo que se hace es pasar a ejecutar la función loop(), que es una función que se ejecuta en el procesador de forma cíclica e indefinida. El código de la función es el siguiente:

```
void loop() {
  int da=0;
  int db=0;
  if(digitalRead(botonencendido)==HIGH){
    estado=1;
    delay(500);
  }
  while(estado){
      digitalWrite(salidaencendido,HIGH);
    if (a<PI/3 && a>=0){ //Sector 1
      db=mag*sin(a)*mod;
      da=mag*sin(PI/3-a)*mod+db;
      while(micros()-b<tcom)
      {
      }
      b=micros();
      ledcWrite(R,da);
      ledcWrite(S,db);
      ledcWrite(T,0);
    }
    if (a>=PI/3 && a<2*PI/3){ //Sector 2
      db=mag*mod*sin(2*PI/3-a);
      da=mag*mod*sin(a-PI/3)+db;
      while(micros()-b<tcom){}
      b=micros();
      ledcWrite(S,da);
      ledcWrite(R,db);
      ledcWrite(T,0);
    }
   if (a>=2*PI/3 && a<PI){ //Sector 3
     db=mag*mod*sin(a-2*PI/3);
     da=mag*mod*sin(-a+PI)+db;
     while(micros()-b<tcom){}
     b=micros();
     ledcWrite(S,da);
     ledcWrite(T,db);
     ledcWrite(R,0);
   }
```

```
if (a>=PI && a<4*PI/3){ //Sector 4
  db=mag*mod*sin(-a+4*PI/3);
  da=mag*mod*sin(a-PI)+db;
  while(micros()-b<tcom){}
  b=micros();
  ledcWrite(T,da);
  ledcWrite(S,db);
  ledcWrite(R,0);
}

if (a>=4*PI/3 && a<5*PI/3){ //Sector 5
  db=mag*mod*sin(a-4*PI/3);
  da=mag*mod*sin(-a+5*PI/3)+db;
  while(micros()-b<tcom){}
  b=micros();
  ledcWrite(T,da);
  ledcWrite(R,db);
  ledcWrite(S,0);
}
if(a>=5*PI/3 && a<2*PI){//Sector 6
db=mag*mod*sin(-a);
da=mag*mod*sin(a-5*PI/3)+db;
while(micros()-b<tcom){}
  b=micros();
 ledcWrite(R,da);
 ledcWrite(T,db);
 ledcWrite(S,0);
}
```

En esta primera parte de la función loop() lo que se hace es:

- En primer lugar, declarar las variables da, db e inicializarlas a 0. Estas variables se utilizarán para asignar el valor de la anchura de cada pulso de las dos ramas del inversor activos en cada sector.

- A continuación, se comprueba si el pulsador está activo y, en caso de que lo esté, se pone la variable estado, que es la que controla el bucle, a 1, para que entre en el bucle de operación. Además, se pone un retraso de medio segundo para que dé tiempo a soltar el botón y evitar que entre y salga del bucle sin desearlo.

- Una vez que se entra en el bucle primero se activa la salida de activación del inversor con la función digitalWrite(pin, HIGH).

- El siguiente paso es comprobar en qué sector se encuentra el ángulo para saber cómo se deben calcular las consignas de anchura de pulsos. Posteriormente, una vez que se cumple una de las condiciones de uno de los seis sectores, se entra dentro de uno de los if(condición) y se realiza lo siguiente:

    - Se calculan las consignas de los transistores como está explicado anteriormente en la teoría del SVPWM en el Capítulo 4 en la ecuación 4 y se multiplica con el mod, que es un valor entre 0 y 1023, ya que se trata de una resolución de 10bits que equivale a 1024 valores posibles. Además se multiplica este valor por la variable mag, que varía entre 0 y 1 para realizar la modulación de voltaje, ya que al multiplicar por este valor inferior a la unidad se reducirá la anchura del pulso y por tanto también su valor medio de la salida, cambiando la amplitud fundamental de la onda en la salida del inversor.

    - A continuación se entra en bucle while(condición), que se ejecuta sin hacer nada hasta que haya pasado el tiempo de conmutación definido. Después se actualiza la nueva variable de tiempo, para prepararla para comprobarse en la siguiente repetición del bucle.

    - Por último, se lanzan las tres consignas calculadas anteriormente a los canales R, S y T que están asociados a cada uno de los pines de las salidas de activación de los transistores. Se decide realizar el cambio de consigna después de la espera y no justo después de su obtención, ya que así se evita que existan adelantos o retrasos en el tiempo de activación de los transistores al producirse cambios en el tiempo de ejecución de todos los cálculos.

Este es el diagrama de flujo de un sector cualquiera de los anteriores, para que se comprenda mejor su funcionamiento:

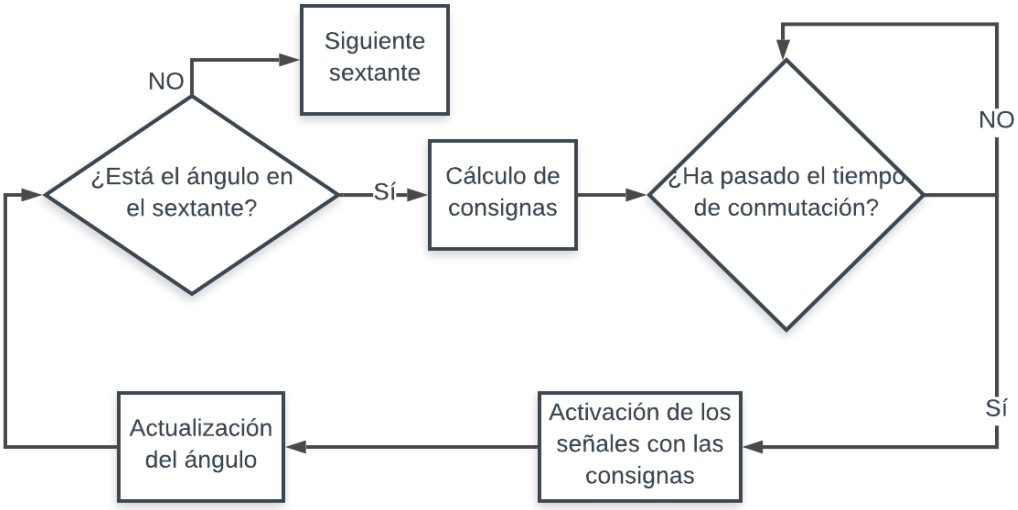


Figura 16. Esquema de funcionamiento del código de selección de sextante

```
    freqp=freqmin +analogRead(pinfreq)/4095.0 *(freqmax-freqmin);
     if(abs(freqp-freq)<=rampamax){
       freq=freqp;
     }else if(freqp-freq>rampamax){
       freq=freq+rampamax;
     }else freq=freq-rampamax;
     mag=freq/freqmax;
     a=a+freq*2*PI/(1000000/tcom);
      if(digitalRead(botonencendido)==HIGH){
      estado=0;
      flag=1;
      }
      if (a>=2*PI)
       a=a-2*PI;
}
```

En esta parte del código se procede a la **actualización del ángulo (a)** que será para el cual se realicen todos los cálculos mencionados anteriormente:

- Lo primero que se hace es calcular la frecuencia de consigna del potenciómetro leyendo el valor con la función analogRead(pin) (valor entre 0 y 4095 por ser de 12 bits) y escalándolo entre la frecuencia máxima y mínima.

- A continuación, se comprueba si la diferencia entre la consigna de frecuencia (freqp) y la frecuencia real (freq) es menor que el valor máximo de cambio por paso del bucle (rampamax) y si lo es se iguala freq a freqp.

- En caso de ser mayor esta diferencia, lo que se hace es incrementar o decrementar la frecuencia real en el valor del incremento máximo según la frecuencia de consigna sea mayor o menor que la real respectivamente. Con este paso se consigue la limitación de rampa en el cambio de frecuencia y tensión aplicada al motor de forma sencilla y efectiva.

- Posteriormente, se calcula mag, que es el índice de modulación de voltaje, que se calcula dividiendo la frecuencia real entre la frecuencia máxima. Ya se debe introducir al motor una tensión proporcional a la frecuencia.

- En la siguiente línea se produce la actualización del ángulo, que se realiza sumando al ángulo anterior la frecuencia en s^{-1} multiplicada por 2*PI porque el ángulo está en radianes, y dividida entre el número de pasadas que se dan en un segundo(1000000us/tcom). En otras palabras, la frecuencia instantánea de alimentación al motor marca la velocidad fasorial de la tensión de alimentación y el ángulo de esta, en cada paso de programa, se obtiene por integración numérica de dicha velocidad.

- Después, se comprueba si hay tensión en el pin asociado al pulsador digitalRead(pin) y si la hay se pone la variable estado a 0 para salir del bucle y la variable flag para saber cuándo se ha salido del bucle.

- Lo último que se realiza dentro del bucle while(estado) es comprobar si el ángulo ha pasado de 2PI radianes y si lo ha hecho restarle una vuelta completa.

El diagrama de flujo asociado a las líneas de código anterior es el siguiente:

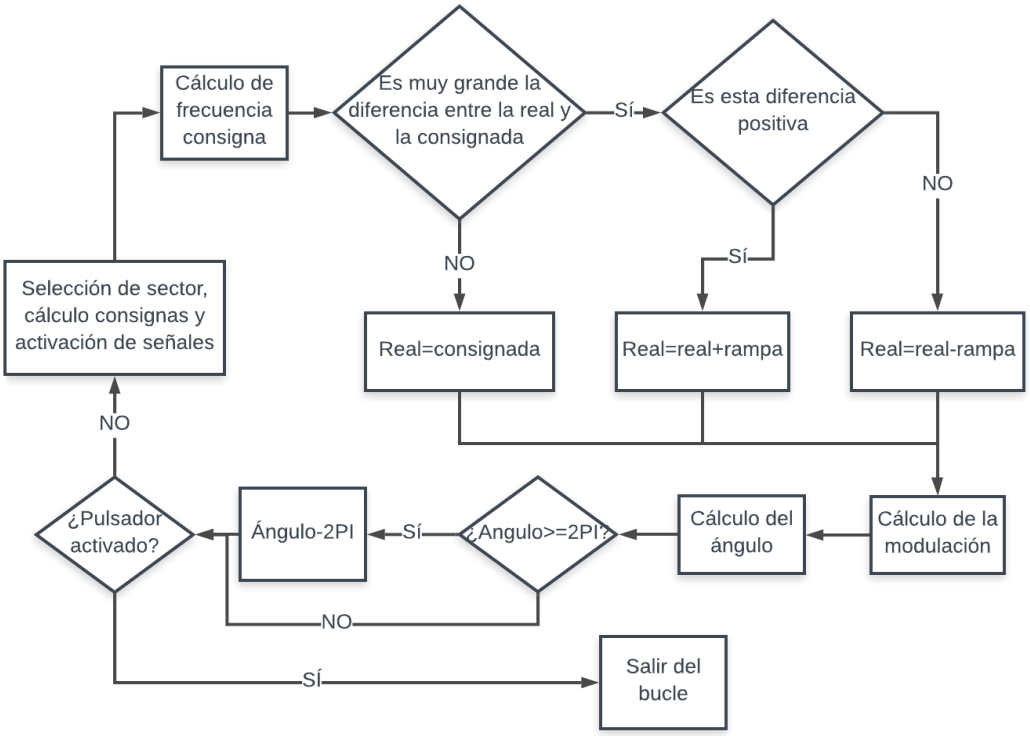


Figura 17. Esquema del código de actualización del ángulo del vector espacial

La última parte del código es la siguiente:

```
    digitalWrite(salidaencendido,LOW);
    ledcWrite(R,0);
    ledcWrite(S,0);
    ledcWrite(T,0);
    freq=0;
    mag=0;
    if(flag){
    delay(500);
    flag=0;
    }
}
```

En esta parte lo que se realiza es:

- Desactivar la señal de activación del inversor con la línea digitalWrite(pin,LOW).
- Poner los pines de las tres señales de los transistores en estado bajo con la función ledcWrite(X,0);
- Reiniciar tanto la frecuencia como la modulación de voltaje a 0, ya que el motor se ha parado y no se desea un arranque rápido.

- Si se acaba de salir del bucle (flag=1), se debe esperar medio segundo para dar tiempo a que se suelte el pulsador y reiniciar flag a 0.

El diagrama de flujo simplificado de toda la función loop es el siguiente:

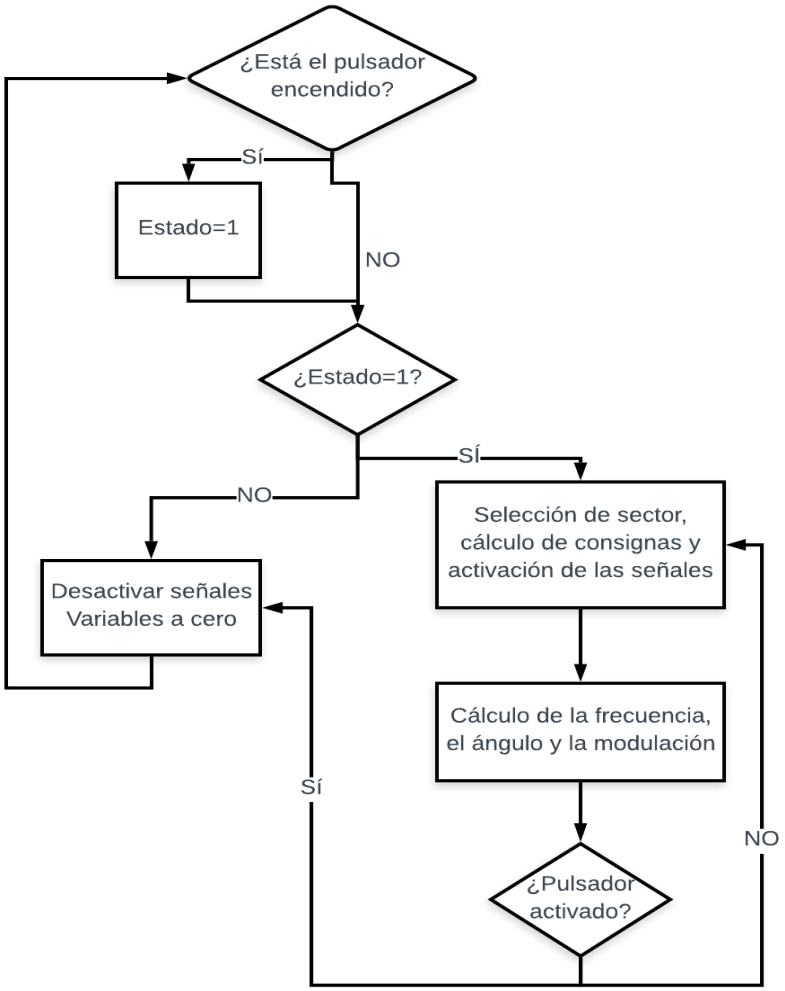


Figura 18. Esquema general del código de la generación de las señales PWM

6.2. Pruebas al código de control V-F

Se procede a realizar pruebas al código anterior para comprobar que no existe ningún error en el código que impida la correcta generación de los patrones de conmutación las 3 ramas del inversor / fases del motor, ya que si existiese se introduciría en el motor un cambio brusco en la tensión que aumentaría las corrientes estropeando el motor o el inversor.

6.2.1. Pruebas con el osciloscopio

Una vez realizado el código, procedemos a hacer medidas de las señales obtenidas mediante un osciloscopio de 2 canales. Como se quieren medir las 3 señales lo que hacemos es medir distintas de 2 en 2 para ver si en alguna de ellas hay un error y corregir así posibles defectos en el código, que provocarían una distorsión de la onda introducida al motor, pudiendo provocar fallos en el mismo o en el inversor.

Primero tomamos medidas de ondas de varios periodos para comprobar que la forma de onda se corresponde con una senoidal, comprobando así que las señales se encuentran desfasadas 120º. Esto se hace para las distintas combinaciones de las tres señales y para distintas frecuencias para tener la seguridad de que el código genera bien los patrones de aperturas de transistores en cualquier frecuencia.

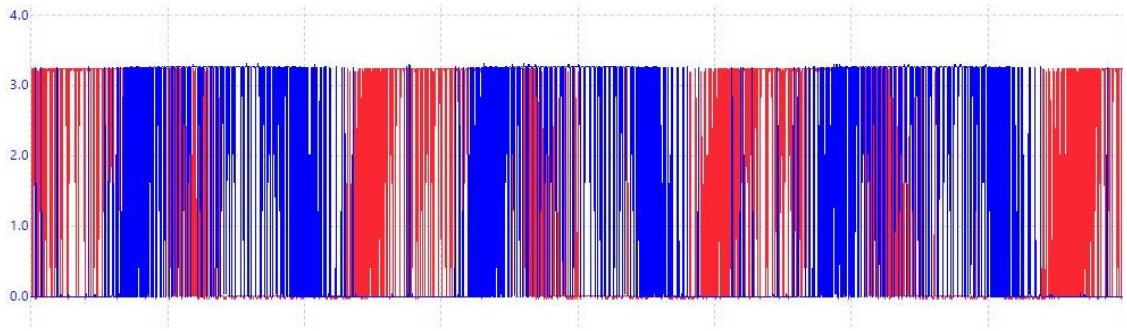


Figura 19. Señal PWM obtenida

Se obtienen gráficas de la señal PWM de la siguiente forma, que después de un filtrado de paso bajo quedan de la siguiente manera:

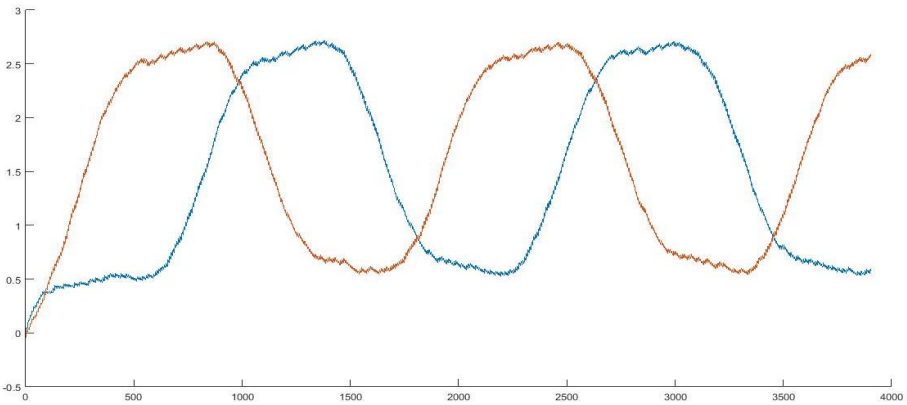


Figura 20. Señal PWM filtrada

Se produce una distorsión en los picos y las crestas de la onda producidas por los armónicos de tercer orden que aparecen en las señales generadas mediante la técnica de SVPWM, ya que aquí consideramos cada señal por separado y no tenemos en cuenta la interacción entre ellas. Al no tener las tres señales simultáneamente, no podemos calcular las interacciones y es por eso que las señales nos salen ligeramente distorsionadas en picos y crestas.

Una vez visto que las señales creadas tienen un contenido de armónicos aceptable, asociado al propio criterio de modulación elegido, y están desfasadas 120º, se procede a hacer un análisis detallado de fragmentos pequeños de las ondas para comprobar que no existen errores en el código que provoquen cambios bruscos de voltaje que puedan afectar al comportamiento del motor.

Para ello, se toman 50 muestras de una parte del ciclo por pareja de señales para cada frecuencia que queremos analizar. En este caso, se toman medidas a 4 frecuencias distintas y 3 combinaciones de fases (R-S, S-T y T-R), lo cual hace un total de 600 muestras tomadas. A estos datos tomados con el osciloscopio, lo que se hace es filtrarlos con un filtro paso bajo y obtener 600 gráficas con forma similar a las siguiente:

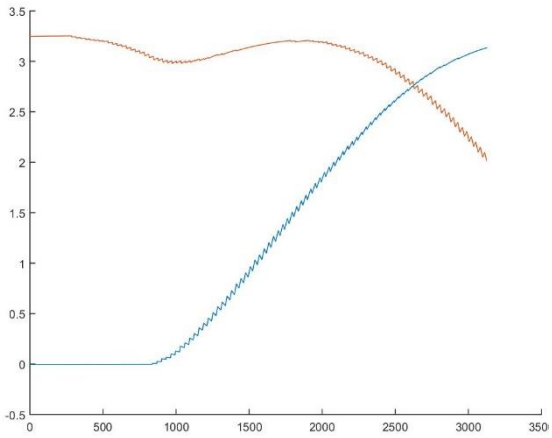


Figura 22. Parte de la PWM 1

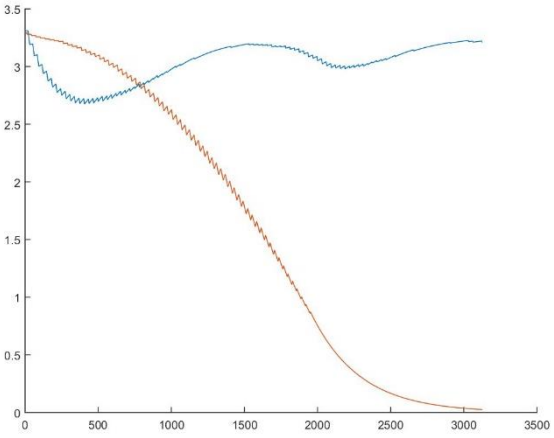


Figura 21. Parte de la PWM 2

En estas gráficas no es excesivamente importante la forma de la onda, ya que esta está distorsionada por el criterio de modulación PWM elegido y por la forma en la que se aplica el filtro, sino que lo que nos importa es que no se produzcan cambios bruscos en la forma de onda o picos que no deben ocurrir. Después de comprobar que en ninguna de las señales se produce ninguno de estos efectos, tenemos la seguridad de que el código funciona y por tanto podemos pasar a comprobarlo con el inversor y el motor.

6.2.2. Pruebas con el motor

Una vez hechas las pruebas del código analizando las señales generadas mediante el osciloscopio, se procede a unir el microcontrolador al inversor STEVAL-IHM028V1, que es una placa con el montaje de un inversor trifásico que permite introducir de forma externa las señales de apertura de los transistores, algo necesario para ser capaces de operar el motor mediante el control del microprocesador.

Para hacer las primeras pruebas, se utiliza un motor de menor potencia que el que se iba a utilizar de forma definitiva. Este tenía unas impedancias mucho mayores para evitar que se pudiera estropear el inversor en caso de que alguna parte del código tuviera algún error y la generación de las señales del motor produjera fallos, pudiéndose producir aumentos en la corriente consumida.

Para realizar estas pruebas se ha utilizado el siguiente montaje:

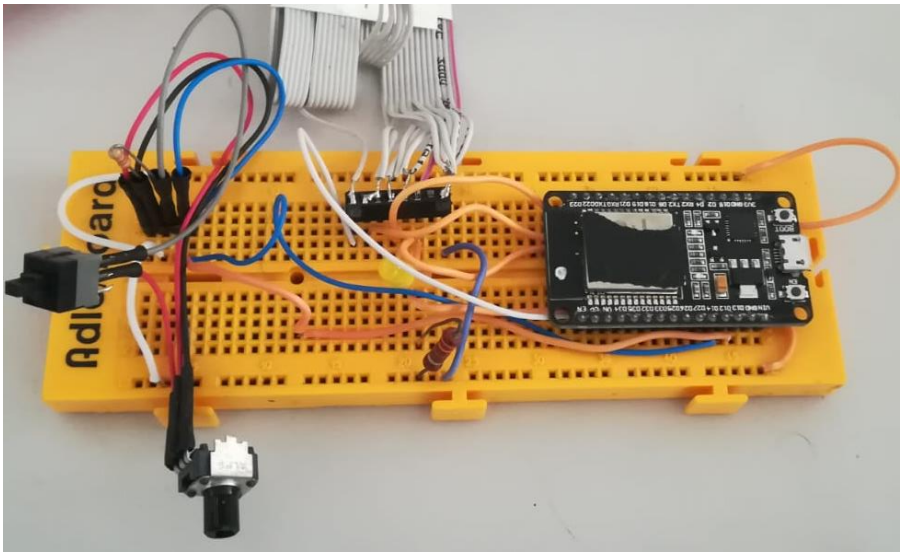


Figura 23. Montaje de pruebas

Con este montaje se comprueba que el motor gira sin ningún problema a las distintas frecuencias que se corresponden con distintas velocidades de giro y ya se puede dar el código de creación de las señales de apertura de transistores y de control escalar básico (control VF) en lazo abierto de la velocidad como válido.

Figura 24. Motor de pruebas

6.3. Implementación del control por Bluetooth

Para gobernar el microcontrolador desde el Smartphone realizando una conexión inalámbrica se necesita implementar en el ESP32 las líneas de código que aparecen a continuación, encargadas de gestionar la comunicación Bluetooth.

6.3.1. Declaración de los elementos

Para la implementación del control por Bluetooth lo primero es poner el siguiente código en la primera parte del código donde se hizo la declaración de las variables.

```
#include "BluetoothSerial.h"
BluetoothSerial ESP_BT;
int hercios;
int incoming;
char datos[2]="";
TaskHandle_t Task2;
```

Esta parte del código se compone de las siguientes partes:

- Incluir la librería “BluetoothSerial.h”, que será la encargada de gestionar todo lo relacionado con las funciones Bluetooth.

- Crear un objeto del tipo BluetoothSerial, que será por donde se mandarán y recibirán las distintas órdenes.

- Declarar las variables que se utilizarán para la recepción de datos por Bluetooth.

- Crear la tarea Task2 que posteriormente se mandará al núcleo 0 del Esp32 para que sea este el encargado de la comunicación bluetooth.

6.3.2. Función setup

Dentro de la función setup creada anteriormente se deberán incluir las siguientes líneas de código:

```
ESP_BT.begin("Microcontrolador_Motor"); //Name of your Bluetooth Signal
xTaskCreatePinnedToCore(
                  Task2code,   /* Task function. */
                  "Task2",     /* name of task. */
                  10000,       /* Stack size of task */
                  NULL,        /* parameter of the task */
                  1,           /* priority of the task */
                  &Task2,      /* Task handle to keep track of created task */
                  0);          /* pin task to core 1 */
```

En las líneas anteriores se desarrollan los siguientes aspectos:

- Primero se inicia el objeto bluetooth y se le da un nombre, en este caso Microcontrolador_Motor que es el que se verá al vincular el teléfono móvil con el microcontrolador.

- Se crea una tarea con la función xTaskCreatePinnedToCore y se asocia a la función Task2Code, además de asociarla a la tarea anteriormente declarada con el uso de &Task2 y que se asocia al núcleo 0 con la última línea de la tarea.

6.3.3. Definición de la tarea Task2

Para definir lo que se deberá hacer dentro de la tarea que ha sido creada anteriormente con la función Task2Code, ya que esta es la función asociada a la tarea creada, el código dentro de esta función es el siguiente:

```
void Task2code( void * pvParameters ){
    while(true){
    if (ESP_BT.available()){
      incoming = ESP_BT.read();
      datos[0]=(char)incoming;
      incoming = ESP_BT.read();
      datos[1]=(char)incoming;
      if(strcmp(datos,"ON")==0)estado=1;
        else if(strcmp(datos,"OF")==0)estado=0;
          else hercios=atoi(datos);
    }
    ESP_BT.print(String(mag));
    ESP_BT.print("#");
    yield();
    delay(10);
    ESP_BT.print(String(hercios));
    ESP_BT.print("&");
    yield();
    delay(10);
    ESP_BT.print(String(freq));
    ESP_BT.print("%");
    yield();
    delay(10);
    ESP_BT.print(String(estado));
    ESP_BT.print("$");
     yield();
     delay(10);
}
}
```

Esta función lo que hace es crear un bucle infinito dentro del cual se ejecutan las distintas partes dentro de la tarea ejecutada por el procesador dedicado para las comunicaciones. Lo que se realiza dentro del código es lo siguiente:

- Primero se comprueba si hay datos disponibles en la cola de entrada con la línea if(ESP_BT.available()) y si los hay se realiza lo siguiente:

- Dentro de la condición anterior se leen los dos datos recibidos y se almacenan en el vector datos[] para comprobar después que es lo que ha entrado.

- Después se comprueba si los datos recibidos son igual a ON o a OF, que son los dos datos que envía la aplicación para encender o apagar el inversor, con la función strcmp(cadena1, cadena2). También se observa si se cumple una de las dos condiciones y se pone la variable estado a 1 o a 0 respectivamente, para activar así el bucle que crea las señales de los transistores.

- Si no se cumple ninguna de las dos condiciones, quiere decir que lo recibido es la consigna de frecuencia, que será un número entre 5 y 50 correspondientes a los hercios de la señal que queremos generar. Estos datos se guardarán en la variable hercios utilizando la función atoi(cadena), que convierte el valor de los datos que se reciben como cadena en una variable de tipo entero.

- Lo siguiente que se hace es enviar los datos que se quieren mostrar por pantalla en la aplicación vía bluetooth. Para ello, se utilizará la función ESP_BT.print(String(variable a enviar)), que convierte la variable que se quiere en una cadena y la envía por bluetooth. Después, se envía un carácter, que es el que reconoce la aplicación para saber en qué cuadro de texto tiene que mostrar el mensaje.

- Entre el envío de cada mensaje se usa la función yield(), que permite que el procesador se dedique a otras tareas y un tiempo de espera para que no se mezclen los mensajes que van a cada uno de los cuadros de texto de la aplicación.

El diagrama de flujo de la tarea encargada de la gestión Bluetooth es el siguiente:

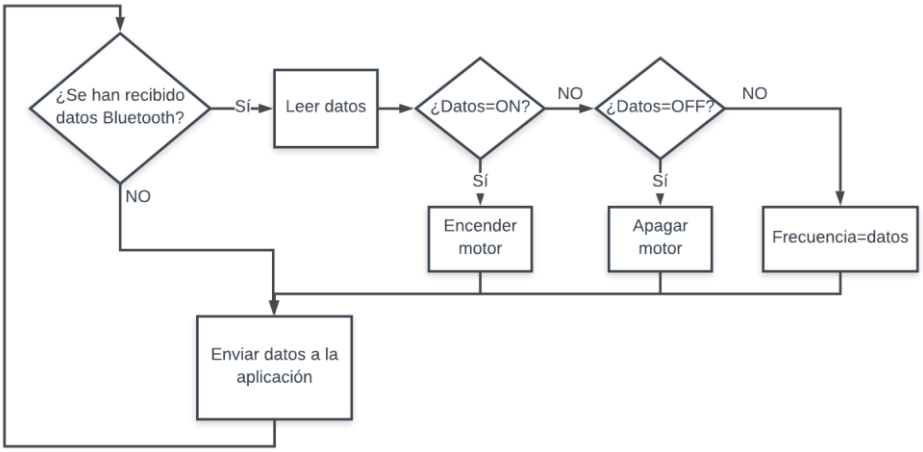


Figura 25. Esquema del código de la interfaz Bluetooth

6.3.4. Variaciones al código anterior

Además del código mostrado anteriormente, se deben hacer una serie de cambios dentro de la función loop, que son los siguientes:

Eliminar las siguientes partes del código de la función loop:

```
if(digitalRead(botonencendido)==HIGH){
  estado=1;
  delay(500);
}
```

```
if(flag){
delay(500);
flag=0;
}
```

```
if(digitalRead(botonencendido)==HIGH){
estado=0;
flag=1;
}
```

Sustituir la siguiente línea de código:

```
freqp=freqmin +analogRead(pinfreq)/4095.0 *(freqmax-freqmin);
```

Por esta otra línea:

```
freqp=hercios;
```

La acción que se persigue con estos cambios es eliminar las partes que se encargaban de comprobar si había habido cambios en el pulsador para encender/apagar el inversor y la señales PWM, ya que ahora se realiza mediante la aplicación Bluetooth.

Además, al cambiar las dos últimas líneas de código mostradas se modifica la forma de actualizar la frecuencia de consigna, que antes se calculaba midiendo el valor en el potenciómetro y ahora se realiza igualándola a la variable hercios que se modifica por bluetooth desde la aplicación Android.

6.3.5. Resultados de la aplicación

El resultado que se obtiene al ejecutar la aplicación en el Smartphone en conjunto con el código anteriormente explicado es el siguiente:

Figura 26. Pantalla de la aplicación Android

Con este código y haciendo uso de la aplicación Android se observa que la consigna de frecuencia puede ser modificada desde el teléfono, que es capaz de apagar y encender el motor bajo demanda, que la modulación de voltaje se realiza de forma correcta y que se aplica la rampa como corresponde para evitar cambios bruscos de frecuencia que puedan llevar asociadas intensidades elevadas.

6.4. Visualización de parámetros por la LCD

Para comprobar que los parámetros que se estaban calculando en el microcontrolador se hacían de forma correcta, se decide mostrar los valores relevantes por la pantalla LCD con la que cuenta el montaje, ya que de este modo se pueden modificar los parámetros que muestra la pantalla de forma muy sencilla cambiando una pequeña parte del código y sin necesidad de observar el Smartphone.

Para poder mostrar valores por la LCD, lo primero que se tiene que incluir en la primera parte del código anterior es:

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x3F, 16, 2);
```

El código anterior se encarga de incluir las librerías necesarias para la gestión de la pantalla LCD y de la creación una variable del tipo LiquidCrystal_I2C llamada lcd, que está en la dirección 0x3F (dirección de identificación en el protocolo I2C) y que tiene 16 columnas y 2 filas.

Dentro de la función setup() se incluirá lo siguiente:

```
lcd.begin(21, 22);
lcd.init();
lcd.backlight();
```

La primera línea se encarga de declarar que el objeto lcd debe iniciarse enviando los datos por los pines 21 y 22, la segunda inicia el objeto y la tercera es la encargada de encender la luz de la pantalla.

Dentro del bucle infinito (while(true)) de la tarea ya creada Task2code se debe incluir lo siguiente:

```
lcd.clear();
lcd.setCursor(0, 0);
lcd.print(freq);
lcd.setCursor(8, 0);
lcd.print(freqp);
lcd.setCursor(0, 1);
lcd.print(mag);
lcd.setCursor(8, 1);
lcd.print(estado);
```

Donde la primera línea se encarga de borrar todo lo que había anteriormente en la pantalla, las instrucciones del tipo lcd.setCursor(columna,fila) se encargan de organizar la información en la LCD, colocando cada dato en una posición determinada de la pantalla y las instrucciones del tipo lcd.print(dato) se encargan de mostrar los datos en la pantalla.

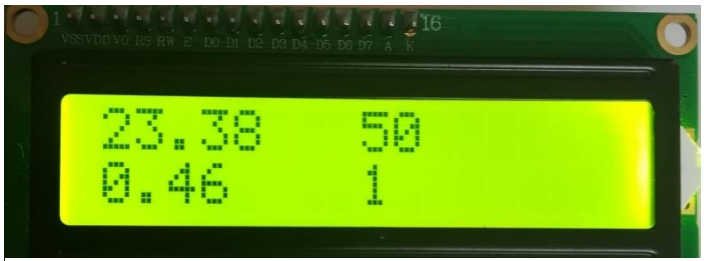


Figura 27. Pantalla LCD local

7. IMPLEMENTACIÓN DEL CONTROL EN CARGA

7.1. Introducción

Una vez vista la capacidad del código para generar las señales de apertura de los transistores, que el sistema de control básico VF es efectivo y que la aplicación Android es capaz de controlar los distintos parámetros que debemos recibir en el microcontrolador, se puede pasar ya a la implementación del control en carga.

La base del control en carga es controlar el deslizamiento entre rotor y estátor, ya que la intensidad que absorbe un motor asíncrono trifásico, así como el par motor, son dependientes del deslizamiento. Por tanto, controlando el deslizamiento se puede controlar el grado de carga de la máquina asíncrona, como muestra la siguiente gráfica.

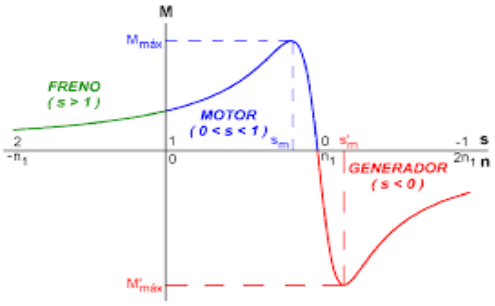


Figura 28. Gráfica deslizamiento-par

El motor que se va a ensayar siempre se encontrará en la zona de régimen motor (0<s<1) y además estará entre el punto de par nominal y el punto de deslizamiento nulo. Por tanto, al aumentar el deslizamiento se estará aumentando siempre el par generado de forma aproximadamente proporcional. Cuando el motor se alimenta a frecuencia ajustable, inferior a la nominal, es importante tener en cuenta que esa relación de proporcionalidad se mantiene con la frecuencia de las corrientes en el rotor ($f_{rot} = s \cdot f_{est}$) siempre que se trabaje con un criterio de regulación de flujo constante, como sucede con el control VF. En otras palabras, es factible integrar un ajuste efectivo de la carga con un control VF teniendo en cuenta la necesidad de forzar una frecuencia de las corrientes del rotor adecuada al nivel de carga deseado.

$$T = k \cdot f_{rot}$$

Por otra parte, en la máquina de inducción, la frecuencia de las corrientes del rotor marca la velocidad de giro del campo respecto del rotor y esta, junto con la velocidad mecánica, marca la velocidad de giro del campo respecto del estator.

$$\Omega_{campo}^{est} = \Omega_{rotor} + \Omega_{campo}^{rot} = \Omega_{rotor} + 2\pi f_{rot} / p$$

En resumen, con estas condiciones y, atendiendo al movimiento relativo de estator y rotor, se tiene que la velocidad de giro del campo respecto del estator debe ser igual a la suma de las velocidades del rotor respecto del estator y a la velocidad con la que el campo gira respecto del rotor. La misma relación se puede establecer para las posiciones del campo respecto del estator (asociada a la tensión de alimentación), del rotor respecto del estator (posición mecánica) y del campo respecto del estator (asociado también a la tensión del estator). Para obtener la orientación necesaria de la tensión

respecto del estator basta, por tanto, con sumar la posición mecánica a la orientación de la tensión respecto del rotor que, a su vez, gira con una velocidad asociada a la frecuencia de las corrientes del rotor.

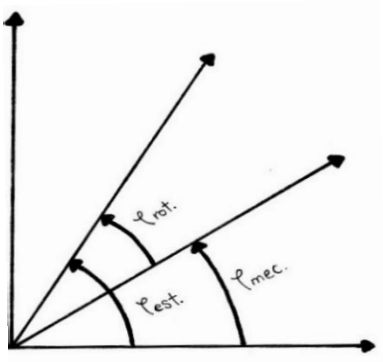


Figura 29. Ángulos del control en carga

Como se muestra en la figura anterior, primero se debe medir el ángulo mecánico para conocer la posición exacta del motor en cada instante. Además, a esta medida hay que sumarle la orientación de la tensión respecto del rotor para tener en cuenta así el deslizamiento del campo respecto del rotor, controlado por la frecuencia de las corrientes del rotor que es proporcional al nivel de carga que se fije por el usuario desde la aplicación y así controlar la carga del motor.

El ángulo mecánico se medirá mediante un encoder absoluto, que es capaz de proporcionar 1024 valores distintos de posición para una vuelta del eje del motor y el ángulo del rotor se calcula como integración de la velocidad de deslizamiento ajustable desde la aplicación para el control del grado de carga. Una vez calculados estos dos ángulos, los sumamos para obtener el ángulo de la corriente de estátor, que es el se utilizará en el código anterior para generar las señales PWM para operar el inversor.

7.2. Código del microcontrolador

En este apartado se va a mostrar el código utilizado para la medida de posición mecánica del rotor, el cálculo de la velocidad de giro del motor y la implementación del control en carga.

7.2.1. Medida de la posición

La medida de la posición del eje se realiza mediante un encoder absoluto, que funciona mediante el protocolo SPI. Las funciones empleadas en este trabajo para la medida de posición han sido desarrolladas en un TFG distinto en el marco del proyecto combinado en el que se integra este TFG. El resultado en este sentido se resumen en disponer de la función medirPos() que se basa en las especificaciones del protocolo SPI particulares del encoder utilizado para establecer la comunicación entre el ESP32 y el encoder y así recibir la señal de posición almacenada en un entero sin signo de 16 bits de los que los

6 bits más significativos son nulos, respetando la resolución de 10 bits del sensor utilizado.

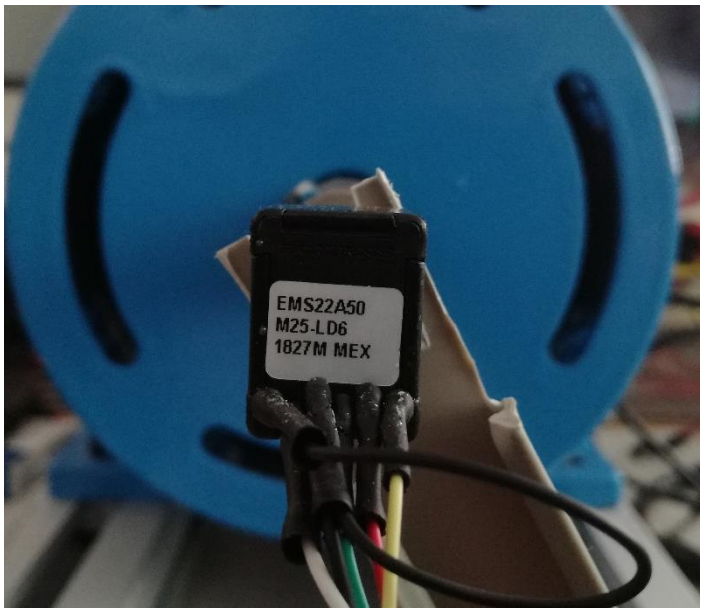


Figura 30. Encoder de posición acoplado al eje del motor

Se trata de un encoder absoluto de posición angular de 10 bits por tanto capaz de proporcionar 1024 estados distintos por cada vuelta del eje. Para su operación cuenta con 6 pines numerados del 1 al 6 de izquierda a derecha en el sentido que aparece en la foto. Los pines 1 y 3 se conectarán a la masa del ESP32 ya que el primero es el pin de entradas digitales y no se hará uso de él y el tercero es el asignado a la masa del encoder y el quinto será utilizado para su alimentación conectándose al pin Vcc. El segundo pin es utilizado por la señal de reloj necesaria para la sincronización entre el sensor y el microcontrolador, se conecta al pin 18 de este. El cuarto pin es el dedicado al envío de señales y que el ESP32 deberá emplear para obtener la lectura de posición en su pin 19. El sexto pin es una entrada encargada de activar la lectura del sensor cuando se encuentra a nivel bajo y se encuentra conectado al pin 5 del microcontrolador.

7.2.2. Medida de la velocidad mecánica del rotor

Al generar el ángulo como suma del mecánico que se mide del sensor y el del rotor, se pierde la noción de la frecuencia de onda que se está introduciendo al motor. Por tanto, no se sabe si se encuentra a frecuencias altas o bajas, ya que dependerá principalmente de la velocidad de giro a la que se encuentre el motor, pues la frecuencia asociada al deslizamiento será pequeña.

Por tanto, al perder la medida de la frecuencia para ajustar la modulación de voltaje, se tiene realizar un cálculo de la velocidad de giro, dado que esta estará ligada con la frecuencia de la onda introducida al motor, para mantener el criterio de operación

elegido de par máximo constante que equivale aproximadamente a relación constante entre tensión y frecuencia.

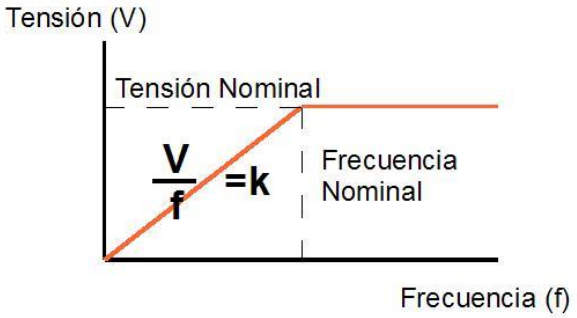


Figura 31. Gráfica tensión frecuencia de un motor

Para realizar la medida de velocidad se utiliza un observador: se estima la velocidad partiendo de un valor inicial (nulo, máquina parada) que se utiliza para estimar el cambio de la posición del rotor a lo largo del tiempo. Una vez obtenida la posición estimada, lo que se hace es calcular el error entre la posición estimada y la posición real medida con el sensor. Dicho error se introduce en un regulador que proporciona a la salida una nueva estimación de la velocidad, con la que a su vez actualizaremos la posición estimada, creando un bucle para la estimación de la velocidad. En nuestro caso, por el comportamiento integrador del estimador de posición ha sido suficiente utilizar un regulador de tipo P, más sencillo de implementar que un regulador tipo PI. La ventaja que hubiera supuesto utilizar un regulador PI, anular el error de seguimiento de la posición en regímenes de velocidad constante, no compensa la mayor complejidad de su implementación.

A continuación se muestra el funcionamiento esquemático elemental del funcionamiento de la estimación de velocidad:

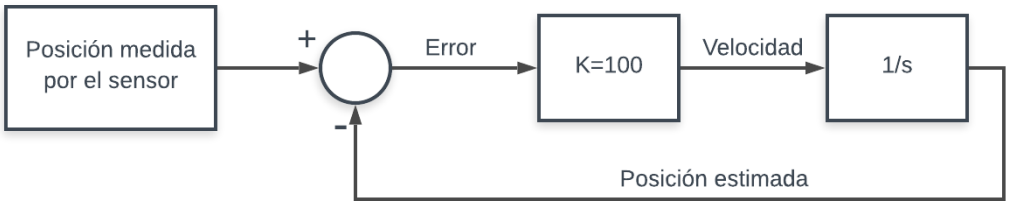


Figura 32. Esquema de funcionamiento del estimador de velocidad

El código que se utiliza con este fin es el siguiente:

Declarar las variables globales que se utilizarán, al principio del código, donde se declaran el resto de variables.

```
float velocidad,poser,error;
uint16_t posicion;
```

El código que realiza el cálculo de la velocidad propiamente es el siguiente, que se debe ubicar dentro de la función loop() para ejecutarse antes de la parte de actualización del ángulo del vector espacial :

```
posicion=medirPos();
error=posicion-poser;
if(error<-500)error+=1024;
if(error>500)error-=1024;
velocidad=100*error;
poser+=velocidad*tcom/1e6;
if(poser>=1024)poser-=1024;
if(poser<=-1024)poser+=1024;
if(poser<0)poser+=1024;
```

- Lo primero que se hace es igualar la posición al ángulo del sensor con la función medirPos() explicada anteriormente. Luego se calcula el error restando la posición y la posición estimada.

- En caso de que el error sea menor que -500 o mayor que 500, se le suma o resta 1024 (posición y poser van entre 0 y 1023) respectivamente, ya que quiere decir que una de las variables se ha reiniciado al dar una vuelta entera y la otra aún no.

- Después se estima la velocidad (en pulsos por segundo) como producto del error por la ganancia del regulador P, que en este caso se ha ajustado a 100. Un valor más elevado supone una mejor respuesta dinámica, una mayor variabilidad en la estimación de velocidad y un menor error de seguimiento de la posición, pero, dado que la posición real es conocida, la disminución de ese error no es un objetivo fundamental. Un valor inferior implica una dinámica más lenta y menor variabilidad de la estimación de velocidad pero, especialmente, un mayor error de seguimiento, que podría interferir con el código que se utiliza para discernir el error ente cambios del número de vueltas.

- Posteriormente, lo que se hace es actualizar el valor de la posición estimada integrando la velocidad estimada, es decir, a partir de la posición estimada anterior sumada con la velocidad multiplicada por el tiempo de paso (es el tiempo entre dos medidas consecutivas) dividido entre un millón para pasarlo a segundos.

- Por último, lo que se hace es colocar la posición estimada entre 0 y 1023 utilizando los 3 últimos if (poser<>x) para comprobar si está por encima de 1024

o por debajo de 1024 y restarle o sumarle 1024 para evitar que desborde. Además, si es menor que 0 se le suma también 1024.

7.2.3. Cálculo del ángulo respecto al estator

Lo primero que se debe hacer para calcular el ángulo es declarar las variables nuevas que se van a utilizar, que son el ángulo mecánico asociado a la posición del rotor y el ángulo asociado al deslizamiento del rotor.

```
float ap,as=0;
```

Para hacer este cálculo, además, se deben eliminar las líneas de código que antes se utilizaban para el cálculo del ángulo del vector espacial, es decir, las siguientes:

```
freqp=hercios;
 if(abs(freqp-freq)<=rampamax){
   freq=freqp;
 }else if(freqp-freq>rampamax){
   freq=freq+rampamax;
 }else freq=freq-rampamax;
 mag=freq/freqmax;
 a=a+freq*2*PI/(1000000/tcom);
  if (a>=2*PI)
   a=a-2*PI;
```

Y deben sustituirse por las siguientes:

```
 freq=hercios*3.33/100.0;
 mag=abs(velocidad)/26000.0;
 if(mag>1)mag=1;
 ap=2*posicion*2*PI/1024;
 if(ap>=2*PI) ap=ap-2*PI;
 as=as+freq*2*PI/(1000000/tcom);
 if(as>=2*PI) as=as-2*PI;
 a=as+ap;
 if (a>=2*PI)
    a=a-2*PI;
```

Las funciones de estas líneas son las siguientes:

- En primer lugar, se calcula la frecuencia del rotor como la variable hercios, que se envía desde la aplicación Android con un valor entre 0 y el 100%. Este valor se multiplica por 3.33 y se divide entre 100. El valor 3.33 se obtiene de los ensayos de carga que se han hecho previamente al motor y representa la frecuencia de las corrientes del rotor en régimen de carga nominal, es decir, el deslizamiento nominal multiplicado por la frecuencia nominal de 50 Hz.

- Lo siguiente que se hace es calcular la modulación (mag) como el módulo de la velocidad dividido entre 26000. Haciendo ensayos se observa que a 50Hz se dan valores de aproximadamente 25700 pulsos por segundo por el encoder , que se corresponden con los 1024 pulsos por revolución por la velocidad de sincronismo de la máquina de 25 revoluciones por segundo. De esta forma se obtiene la relación V/F constante (e igual a la relación V/F del régimen nominal de la máquina) característica de la operación en la región de par máximo constante o flujo constante de la máquina. Si el valor así calculado de mag es mayor que 1 se satura a 1 consiguiendo la operación efectiva en la zona de potencia máxima constante o debilitamiento del campo de forma sencilla.

- Se calcula el ángulo mecánico (ap) como 2 veces la posición obtenida por el sensor, multiplicada por 2 π y dividido entre 1024, para realizar la conversión de 0 a 1023 (que es el rango de medida de la posición obtenida por el sensor) a radianes. Se debe multiplicar por 2, ya que el motor que se utilizará es un motor de 2 pares de polos y, por tanto, el ángulo eléctrico es dos veces el ángulo del rotor porque para cada vuelta del rotor se dan dos vueltas del vector espacial. Si el ángulo mecánico es mayor que 2π se resta una vuelta.

- Después, lo que se realiza es el cálculo del ángulo de deslizamiento (as) como integración de la frecuencia del rotor. Para ello, se incrementa el valor del ángulo en función de la frecuencia multiplicándola por 2π para pasar a radianes y dividiendo por el número de veces que se ejecuta en 1 segundo (1e6/tcom). Si el ángulo es mayor de 2π se resta una vuelta.

- Por último, se suman ambos ángulos anteriormente calculados y si la suma supera los 2π se le resta también una vuelta, al igual que se hacía con los otros dos. Este ángulo será el que se utilizará para generar las consignas para la creación de las tres señales de apertura de los transistores.

7.3. Pruebas del control en carga

En este apartado se procede a realizar las primeras pruebas al código de implementación de control en carga, previas a la realización de los ensayos con la finalidad de encontrar errores en el funcionamiento del conjunto microcontrolador-inversor-motor. En ellas, se explicarán los resultados obtenidos analizando los datos y mostrando los errores descubiertos y la forma de solventarlos. Primeras pruebas a carga nula

Una vez escrito el código se realizan las primeras pruebas del mismo. Para ello, primero se prueba el código con carga nula, es decir con deslizamiento 0, por tanto únicamente se introduce una señal que gira síncrona con el motor.

En las primeras pruebas se observa que el motor gira con pequeños tirones y haciendo ruidos extraños, es por eso que se realiza un registro de las variables posición, posición estimada y velocidad de 2000 valores que se almacenan en un vector y se realizan gráficas obteniéndose los siguientes resultados:

Figura 33. Registro de la posición 1

En la figura anterior se muestra la gráfica de la medida de posición proporcionada por el sensor, donde se observan errores en la toma de datos que inducen a cambios bruscos en la señal, provocando los tirones y ruidos existentes.

Figura 34. Registro posición estimada 1

En esta figura se muestra la gráfica de posición estimada, donde no se aprecian alteraciones ni errores importantes.

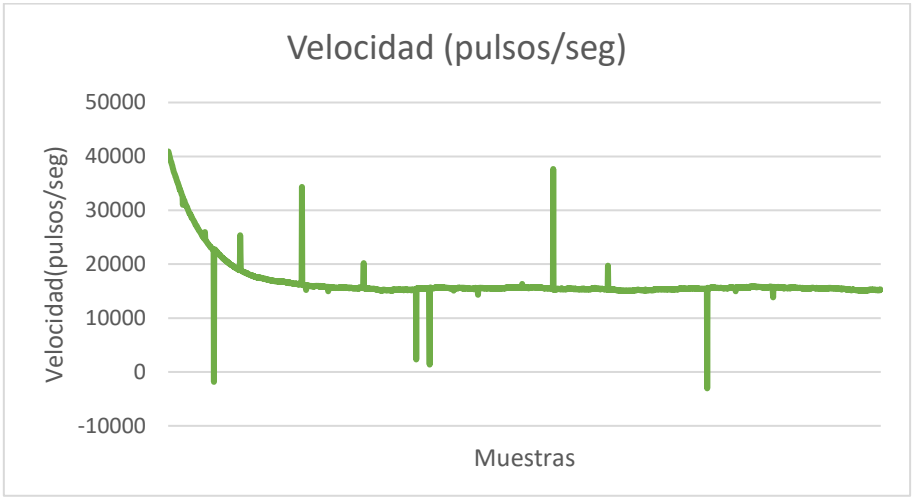


Figura 35. Registro de velocidad 1

En la anterior figura se muestra la gráfica de velocidad en pulsos por segundo, se observan cambios bruscos en la velocidad producidos por los incrementos repentinos observados la medida de la posición.

Una vez visto que el error estaba en la medida de posición, lo que se hace es aumentar los retrasos entre los flancos de señal de reloj y el instante de lectura para el protocolo de comunicación SPI. Los valores originales se habían establecido para obtener una medida estable en un tiempo de lectura mínimo en un entorno con menos ruido electromagnético (máquina conectada a la red de frecuencia constante en lugar de al

inversor) y este ensayo ha puesto de manifiesto que se requieren tiempos de establecimiento de la señal de digital de datos durante la operación en un entorno con mayor ruido, como es el caso del funcionamiento con el inversor. En cualquier caso, el tiempo de transmisión de la medida de la posición se ha mantenido en un rango aceptable de unos 22µs. Con esto hecho, se vuelve a hacer un registro de las mismas variables y se realizan gráficas con los siguientes resultados:

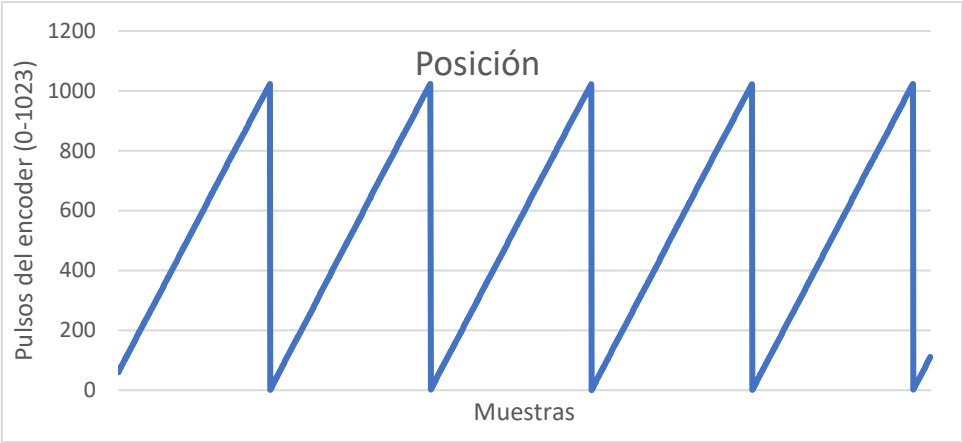


Figura 36. Registro de posición 2

En la gráfica anterior se puede apreciar una mejora significativa en la medida de posición, siendo solventados los errores apreciables antes de los cambios realizados al código.

Figura 37. Registro de la posición estimada 2

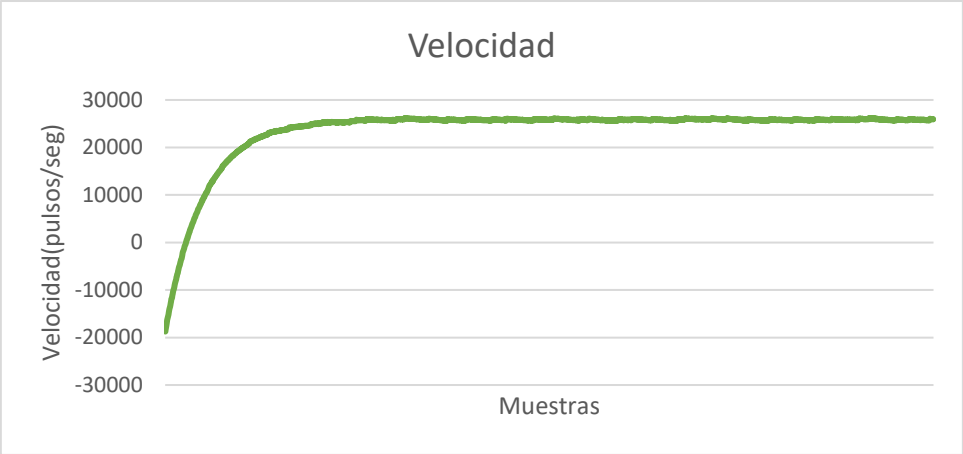


Figura 38. Registro de la velocidad 2

En esta gráfica se puede observar la ausencia de cambios bruscos en la medida de velocidad. Comprobado que en ninguna de las gráficas se observan valores anómalos que puedan alterar la generación de las consignas de las señales PWM se procede a realizar de nuevo las pruebas del código a carga nula y sin apreciarse tirones, ruidos ni ninguna anomalía que pudiese indicar la presencia de un error en el mismo.

7.3.1. Pruebas en carga

Una vez comprobado el correcto funcionamiento del código a carga nula, se procede a realizar las primeras pruebas utilizando deslizamiento no nulo. Para ello, se elige mediante la aplicación una carga baja y se procede a realizar la prueba.

En la primera prueba se observa el bloqueo del rotor durante una parte del giro y un funcionamiento totalmente anómalo y con tirones bruscos que provocan un ruido intenso en el motor. Por ello, se decide volver a hacer un registro de las variables implicadas para ver dónde está el fallo del código que provoca el mal funcionamiento del motor. Se decide hacer el registro de las variables ángulo de deslizamiento, ángulo mecánico y ángulo de generación de las consignas (suma de los otros dos) y los resultados obtenidos son los siguiente:

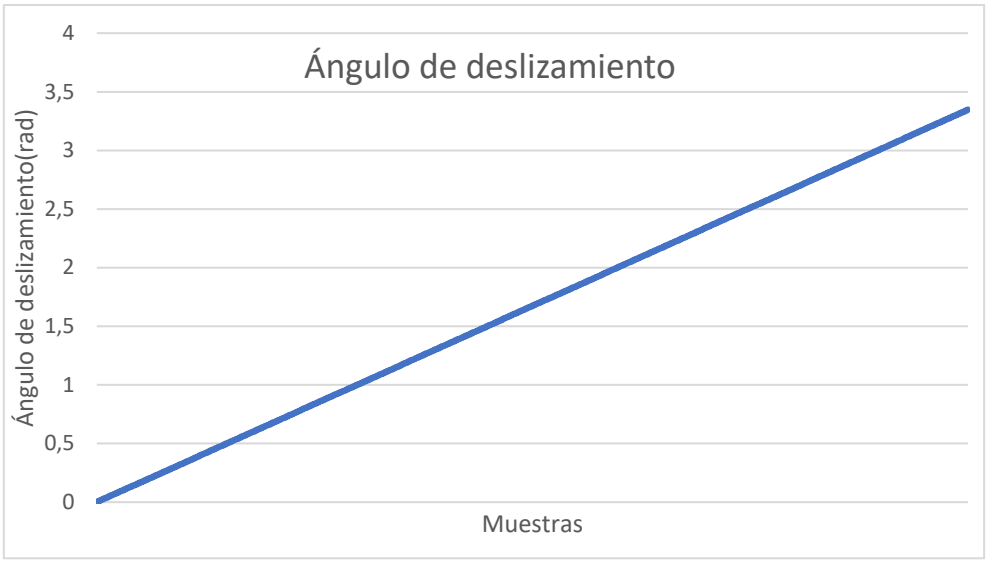


Figura 39. Registro del ángulo de deslizamiento 1

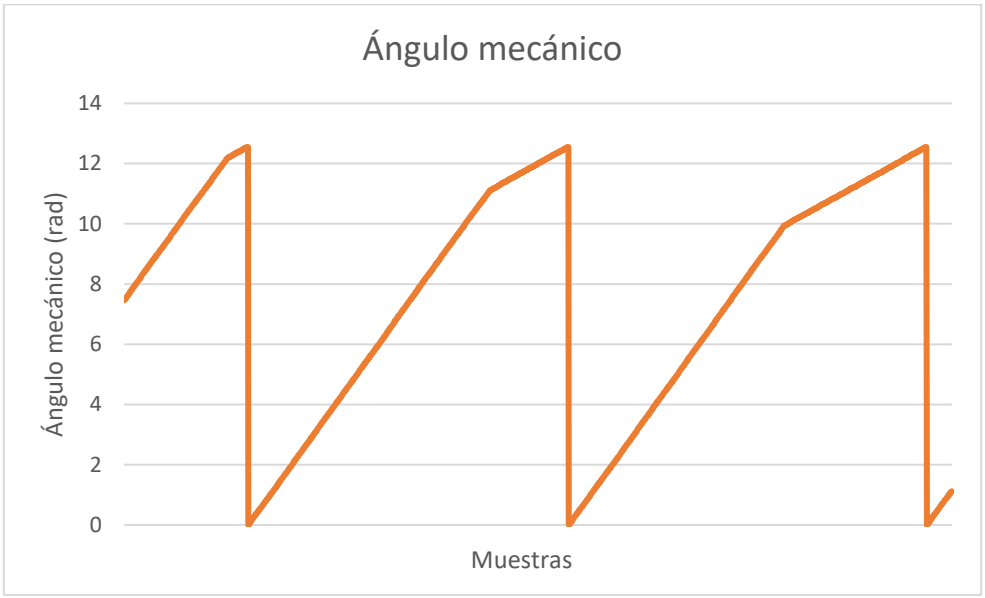


Figura 40. Registro del ángulo mecánico 1

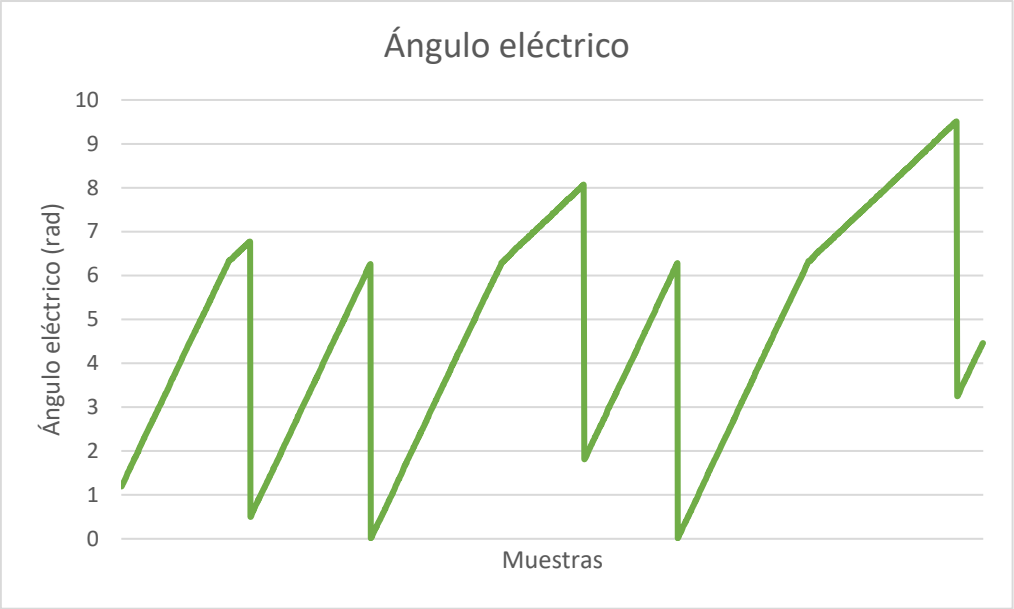


Figura 41. Registro del ángulo eléctrico 1

Analizando las gráficas se puede observar que el ángulo mecánico es, en algunos instantes superior a 2π, lo cual hace que el ángulo eléctrico también sea mayor que dicho valor y, por tanto, se introduzca un valor de ángulo al código fuera del rango de funcionamiento. Por ello, al no funcionar correctamente el código de selección de sextante para la modulación PWM durante el tiempo que el ángulo eléctrico superaba los 2π, se introducía una onda del mismo valor de ángulo, que equivale a corriente continua y bloqueaba el rotor haciendo que se dieran tirones.

Para solucionar este problema, lo que se hace es restringir el ángulo mecánico entre 0 y 2π con la siguiente línea, que en el código previamente mostrado ya se encontraba escrita ya que se ha decidido mostrar el código ya corregido.

```
if(ap>=2*PI) ap=ap-2*PI;
```

Una vez añadida esta línea, se decide volver a realizar un registro y graficar los datos de los tres ángulos de nuevo antes de realizar pruebas con el motor evitando así posibles consecuencias para el motor o el inversor y los resultados obtenidos son los siguientes:

Figura 42. Registro del ángulo de deslizamiento 2

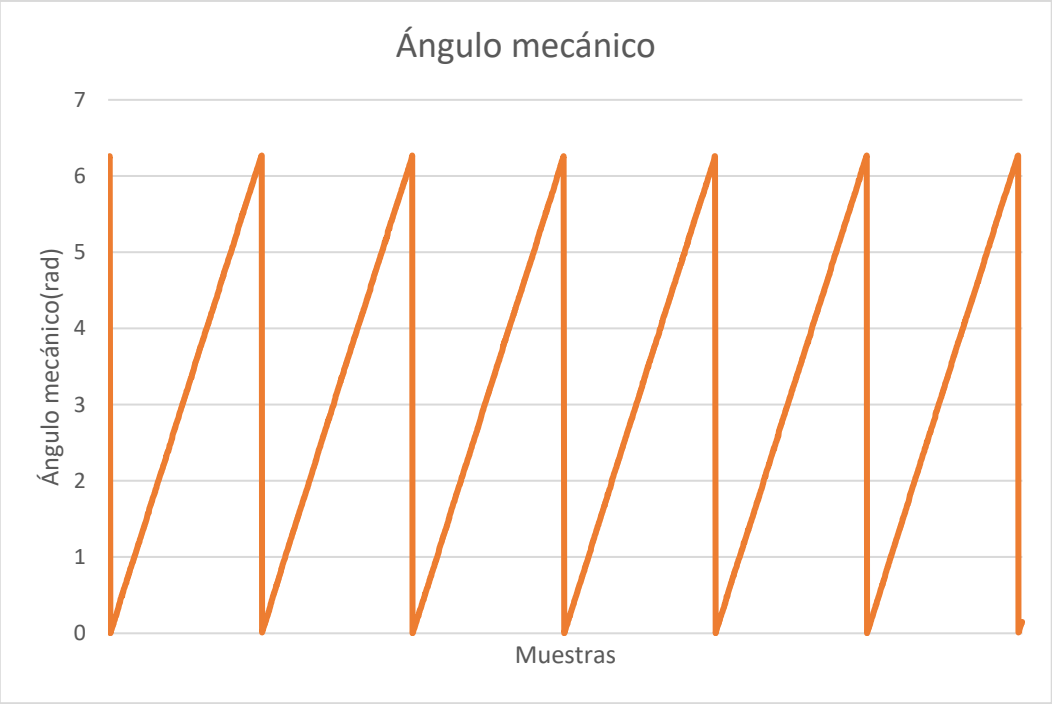


Figura 43. Registro del ángulo mecánico 2

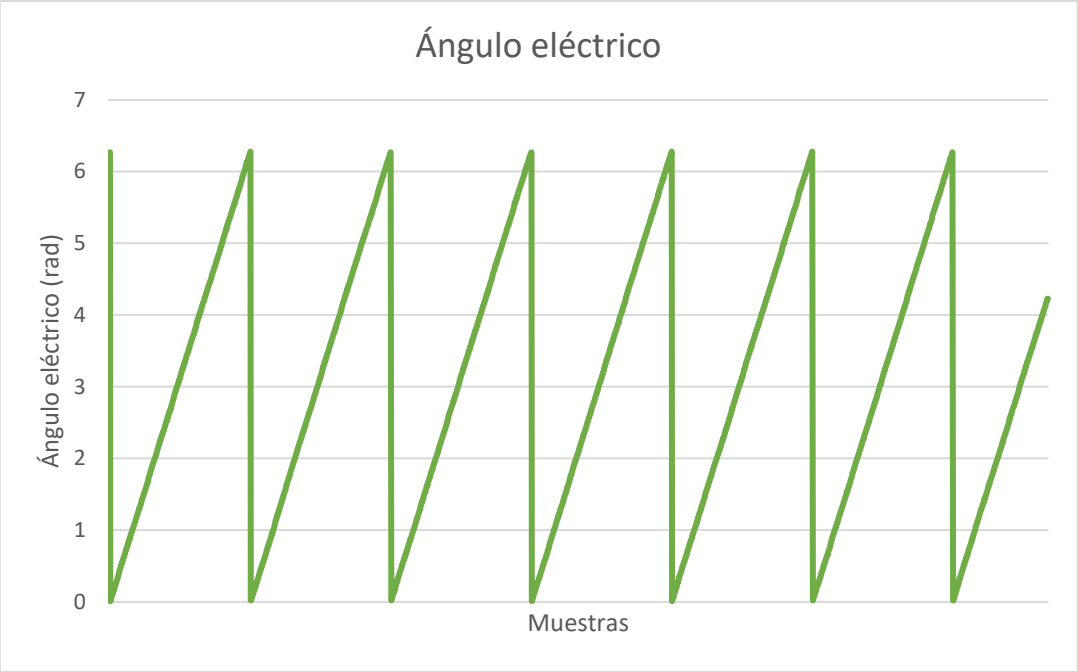


Figura 44. Registro del ángulo eléctrico 2

Como se puede observar en las tres gráficas mostradas realizadas con esta variación en el cálculo del ángulo mecánico, las gráfica del ángulo eléctrico y mecánico cambian totalmente su forma y evidenciándose la ausencia de cambios inesperados en la misma. Además, se observa que tanto la gráfica del ángulo eléctrico como la del mecánico están en el rango de 0 a 2π.

A la vista de los resultados, se decide volver a probar con el código corregido a baja carga para comprobar que no existen corrientes excesivamente elevadas o que se producen tirones indeseados en el motor. Como no se da ninguno de estos efectos, se decide probar el motor a cargas más elevadas sin apreciarse cambios ni errores de funcionamiento, por tanto se decide pasar a realizar las pruebas en carga del motor.

8. ENSAYOS EN CARGA

Una vez comprobado que el código funciona correctamente se puede proceder a la realización de los ensayos del motor en carga. Estos se realizan a distintos grados de carga y a distintas velocidades de giro. En este apartado se mostrarán los datos obtenidos en los distintos ensayos como son la tensión y corriente de línea, la potencia activa y reactiva y los resultados que aparecen en la pantalla y por último se realizará una comparativa analizando diversos parámetros de los ensayos para observar las diferencias entre ellos. Los resultados que se muestran a continuación sirven para demostrar que se ha alcanzado el objetivo fundamental del TFG, desarrollar un sistema de control de inversor basado en un microcontrolador para un banco de ensayos de accionamientos eléctricos gobernado mediante un Smartphone estableciendo comunicación Bluetooth para operar en control de velocidad basado en la variación de la frecuencia y el control de carga.

8.1. Ensayos a velocidad media

Primero se ensaya el motor a velocidad baja introduciendo una frecuencia de 15Hz en el variador del motor auxiliar para 3 grados de carga distintos:

8.1.1. Ensayo a carga nula

En primer lugar, se realiza una prueba con el motor en vacío y los resultados que se obtienen son los siguientes:

Gráfica de la tensión fase masa obtenida de dos fases del motor:

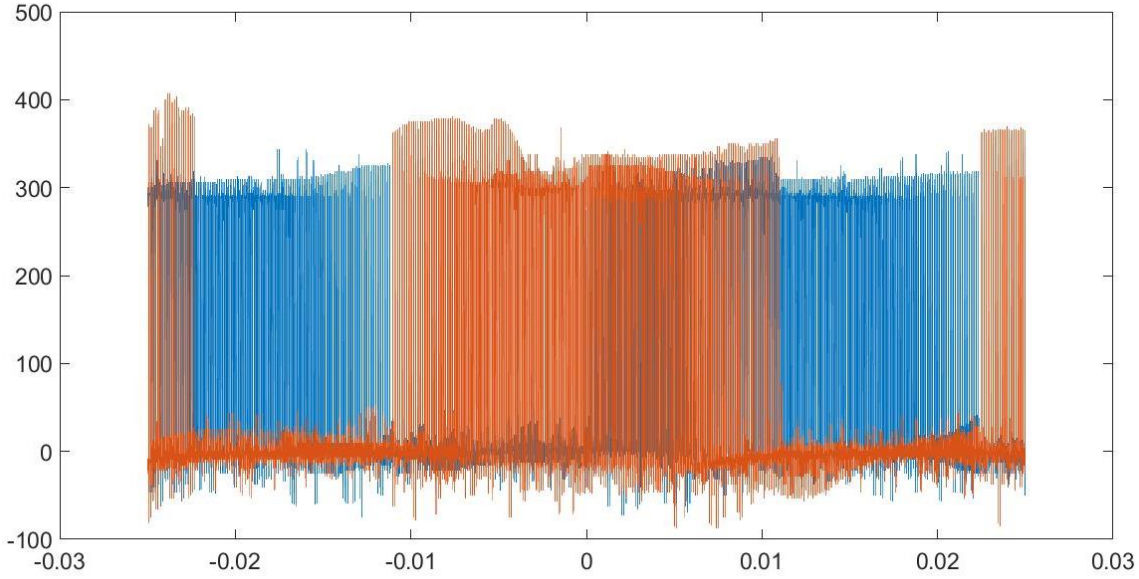


Figura 45. Tensión fase-masa del motor

De la gráfica anterior no se puede obtener demasiada información ya que se trata de una señal PWM de frecuencia 10kHz y, por tanto, se debe realizar un filtrado de paso bajo para ver la forma de la onda generada. La gráfica una vez aplicado el filtro se muestra a continuación.

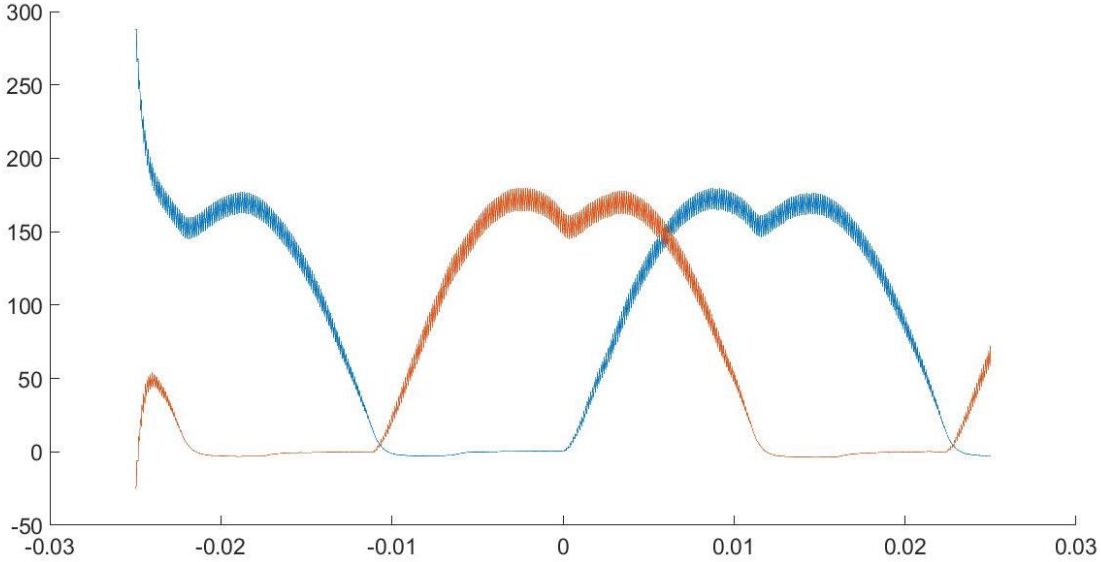


Figura 46. Tensión fase-masa filtrada

En esta gráfica ya se puede ver la forma característica de la modulación SVPWM asimétrica con dos sextantes de valor nulo. Se observa también la deformación en los picos positivos típica de la compensación automática con armónicos múltiplos de 3 que permite de forma natural a la modulación SVPM alcanzar todo el rango de tensiones de salida asociadas a la tensión de red y evitando la pérdida de tensión del 15% típica de la modulación PWM sin compensación homopolar.

Sin embargo, la tensión entre fase y masa no tiene tanta utilidad práctica para analizar el comportamiento del motor, como la tensión de línea, que es la que se aplicará a la fase por ser una conexión en triángulo, como resta de las dos PWM anteriores obteniendo tras el filtrado la siguiente gráfica:

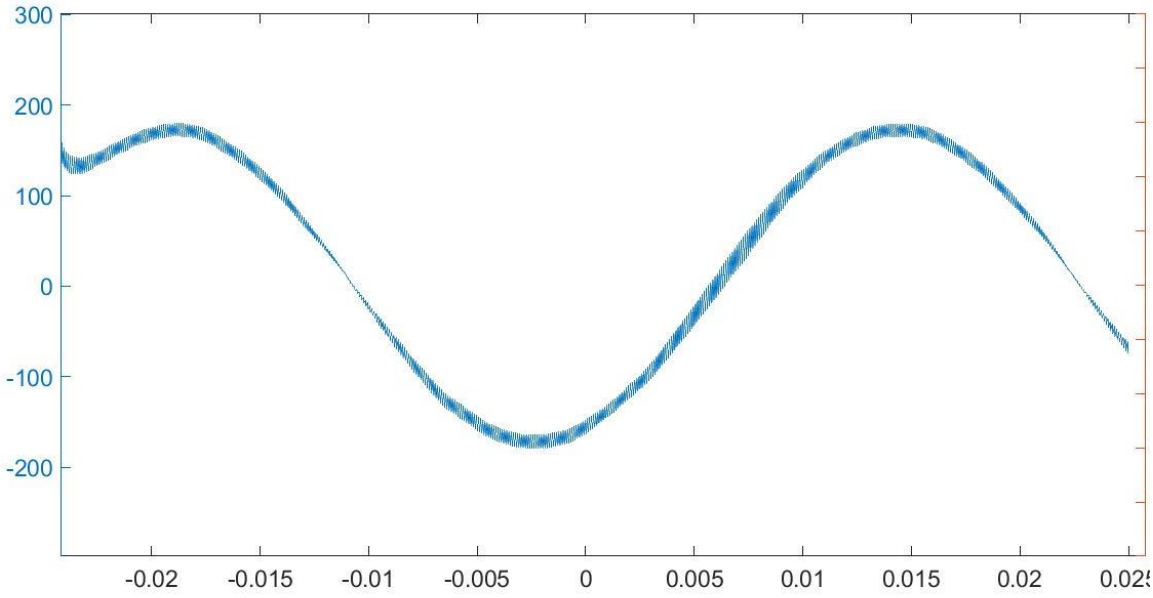


Figura 47. Tensión de línea del motor filtrada

En la anterior gráfica ya se puede observar claramente la tensión con forma senoidal, que es lo que se espera de la tensión de una de las fases del motor conectado en triangulo al utilizar un inversor trifásico.

Además de la medida de tensión, también se ha realizado la medida de corrientes del estátor y se han obtenido los siguientes resultados:

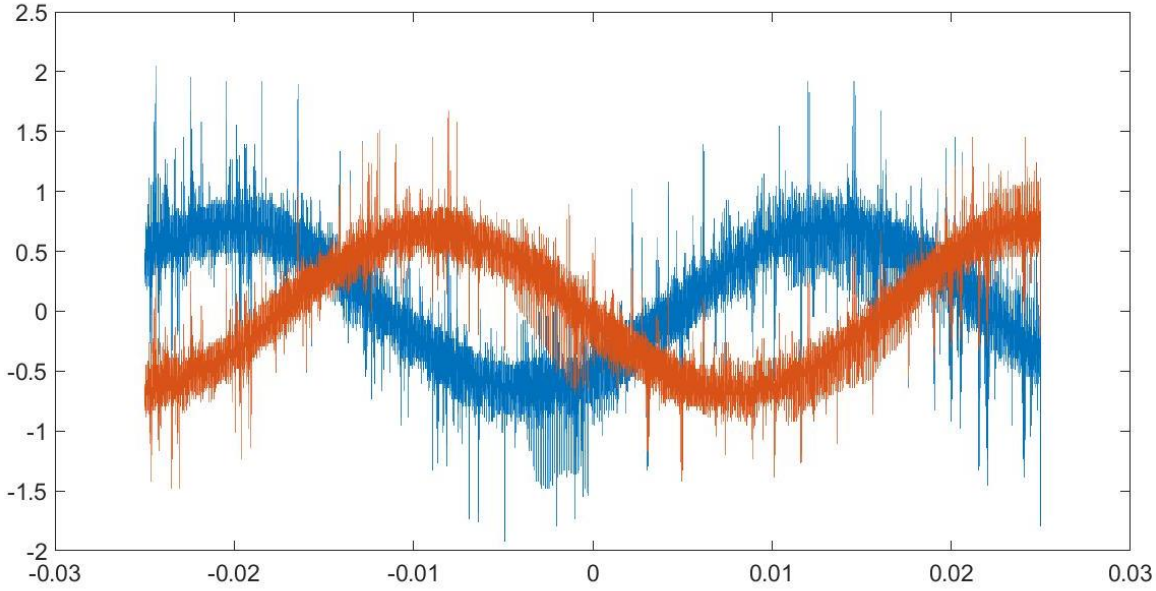


Figura 48. Corriente de línea

Las cuales después del filtrado quedan del siguiente modo:

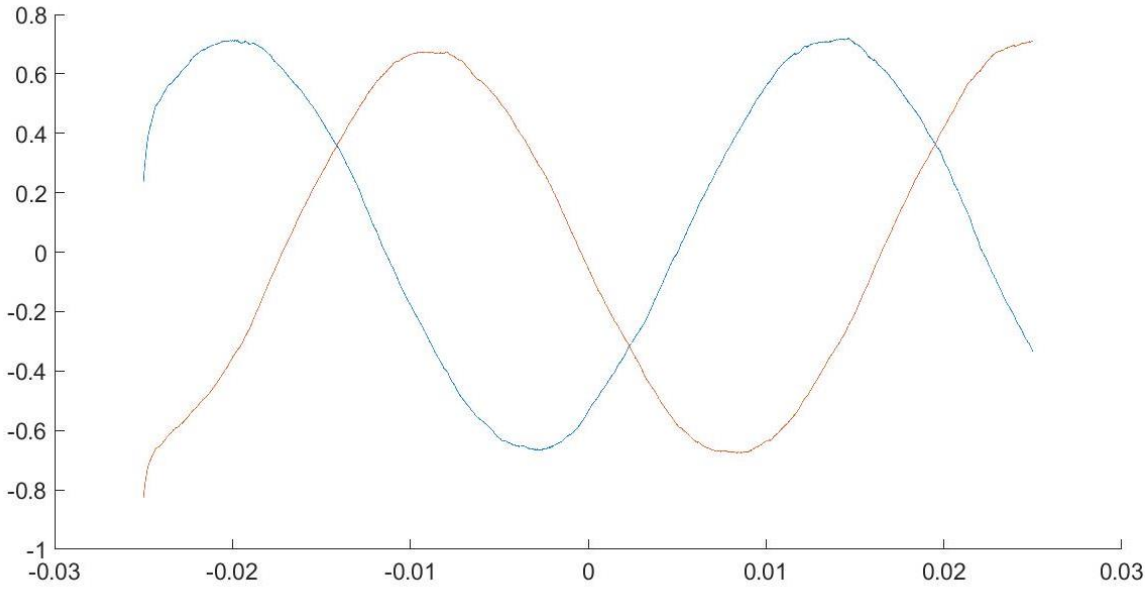


Figura 49. Corriente de línea filtrada

En la gráfica anterior se puede comprobar una forma cercana a la senoidal (bajo contenido de armónicos) en las dos señales y es posible observar que están desfasadas aproximadamente 120º.

Superponiendo las gráficas filtradas con distintos ejes para poder observar el desfase entre corriente y tensión se obtiene el siguiente resultado:

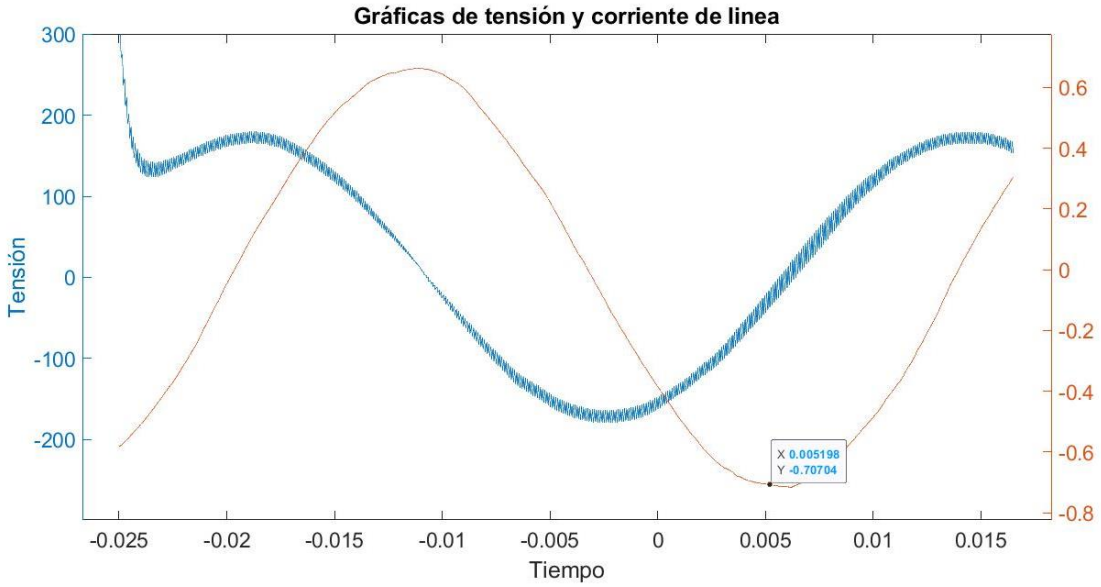


Figura 50. Tensión y corriente de línea, 15Hz, 0% carga

En esta gráfica ya se puede observar el desfase entre la tensión fase masa y la corriente de línea, que es de aproximadamente 90 grados al hallarse el máximo se la corriente ligeramente adelantado al paso por cero de la tensión.

Además de las medidas de corriente y tensión, se han tomado datos del analizador de energía que muestra lo siguiente:

Figura 51. Analizador de energía 1

En él se puede observar una potencia activa muy baja comparado con la potencia reactiva, ya que estamos a carga nula. Calculando el desfase como artg(421/63) se obtiene un valor de 81,49º ligeramente por debajo de los 90º, como se puede observar en la gráfica anterior.

Los datos que muestra la aplicación Android son los siguientes:

Figura 52. Pantalla de la aplicación Android 1

En ella se puede ver como la velocidad de deslizamiento es 0 y la velocidad de giro del motor es de 896 rpm o 15402 pulsos por segundo del sensor de posición y un índice de modulación del 58,88%.

8.1.2. Ensayo al 50% de carga

Para el resto de ensayos se van a mostrar únicamente las gráficas de voltaje y tensión superpuestas y filtradas de igual forma que en el caso anterior.

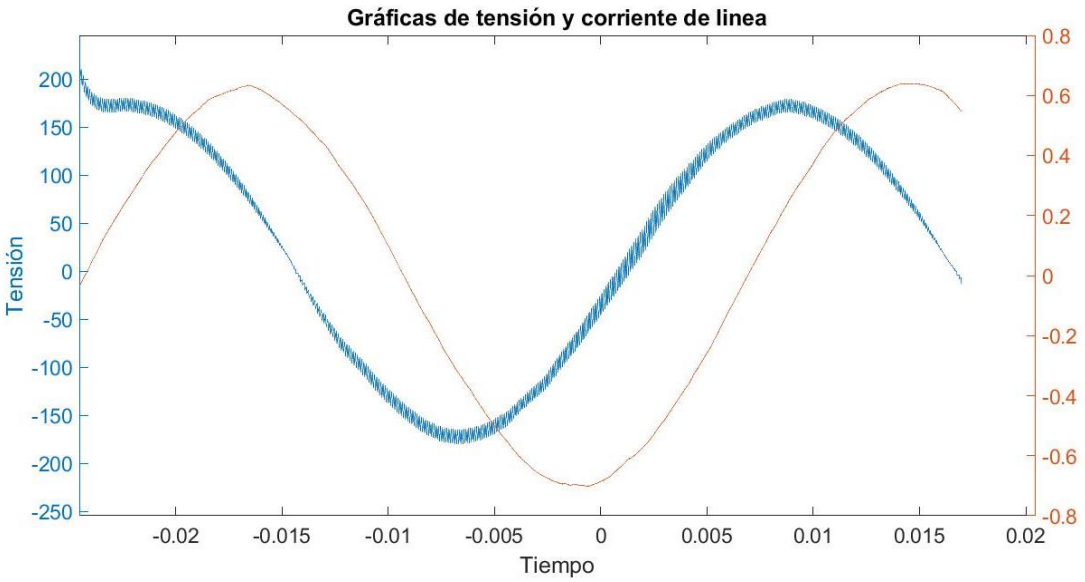


Figura 53. Tensión y corriente de línea, 15Hz, 50% carga

De esta gráfica se observa que el voltaje se ha mantenido prácticamente constante, y que el número de ciclos tampoco se ha modificado. Además, se puede ver que la corriente va ligeramente menos atrasada con respecto al voltaje que en la gráfica anterior, ya que al ser de mayor grado de carga existirá mayor porcentaje de corriente activa que en el caso anterior y, por tanto, la corriente irá menos retrasada. Este fenómeno se puede observar de forma más clara en el punto máximo de la corriente en relación con el voltaje.

El resultado del analizador de energía es el siguiente:

Figura 54. Analizador de energía 2

Aquí se puede observar que la potencia activa ha subido de forma considerable y que no ha habido grandes cambios en la potencia reactiva. Haciendo artg(448/192) se obtiene un desfase de 66,8º.

Los datos que muestra la aplicación Android son los siguientes:

Se puede observar que al aumentar el grado de carga aumenta ligeramente la velocidad y al aumentar esta aumenta también el índice de modulación, además de la velocidad de deslizamiento del rotor.

Figura 55. Pantalla de la aplicación Android 2

8.1.3. Ensayo al 100% de carga

De este ensayo se muestra la misma gráfica que en el anterior, obtenida del mismo modo que los dos casos anteriores.

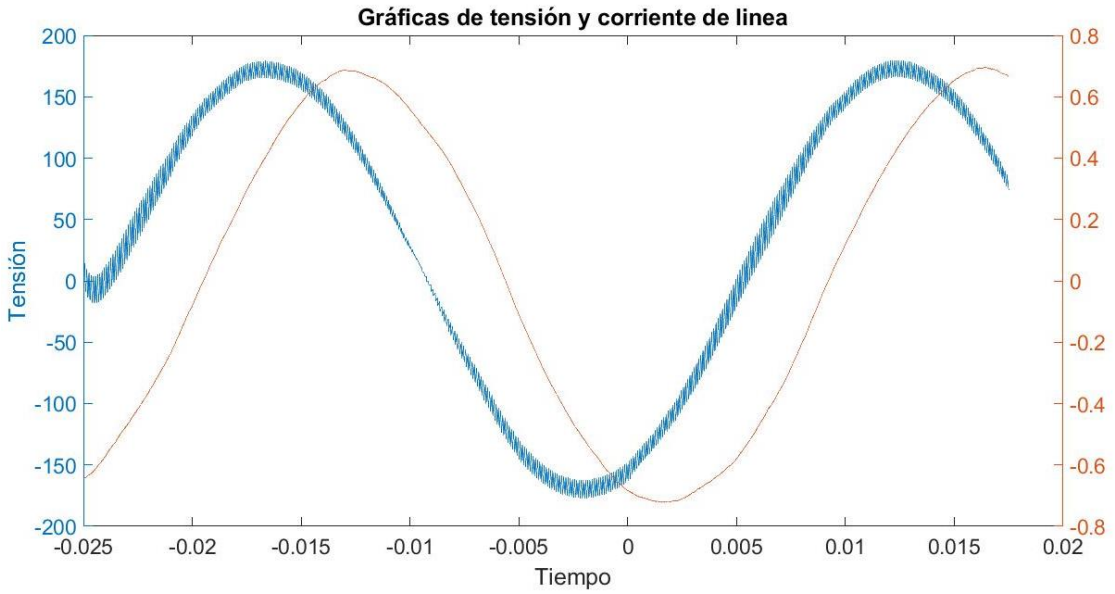


Figura 56. Tensión y corriente de línea, 15Hz, 100% carga

En ella se observa que las gráficas de voltaje no han cambiado y que la de corriente se encuentra menos retrasada respecto al voltaje que en los otros dos casos por lo explicado anteriormente.

El analizador de energía muestra lo siguiente:

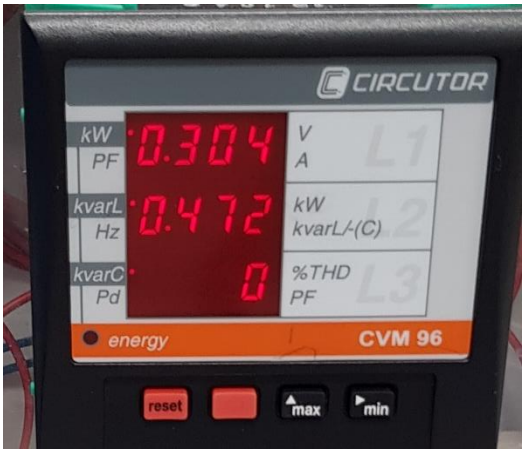


Figura 58. Analizador de energía 3

Figura 57. Pantalla de la aplicación Android 3

Se puede ver un aumento importante de la potencia activa y un pequeño aumento de la potencia reactiva.

En la pantalla de la aplicación se puede observar un aumento en la velocidad debido al mayor grado de carga y, por tanto, un aumento en el índice de modulación. En este sentido conviene aclarar que ambos motores trabajan acoplados, el motor auxiliar sometido a una consigna de frecuencia fija de 15 Hz y el principal (el alimentado por el inversor y controlador por el microcontrolador) trabaja en modo control de carga. En estas condiciones, al aumentar la consigna de grado de carga, aumenta la frecuencia de salida y el par motor aumenta. Como consecuencia de ello la velocidad del motor auxiliar, al estar ambos rotores acoplados, aumenta en la misma medida y entra en modo regenerativo, presentando un par de frenado mayor cuanto mayor es el deslizamiento (en valor absoluto, ya que es negativo para el motor auxiliar). De forma muy rápida se alcanza un equilibrio a una velocidad superior a la de sincronismo del motor auxiliar (900 r.p.m. en estos ensayos) en el que ambas máquinas trabajan en régimen permanente (motor la máquina principal, generador la auxiliar).

8.2. Ensayos a velocidad nominal

Después de realizar los ensayos a velocidad baja se procede a replicar los ensayos a la velocidad nominal del motor, para ello se alimenta el motor auxiliar a una frecuencia de 25Hz para que el conjunto de motores gire a una velocidad de aproximadamente 1500 rpm.

8.2.1. Ensayo a carga nula

Se muestra la gráfica de tensión y corrientes superpuestas para el mismo periodo de tiempo medido y con los mismos valores de escalado y filtrado anterior.

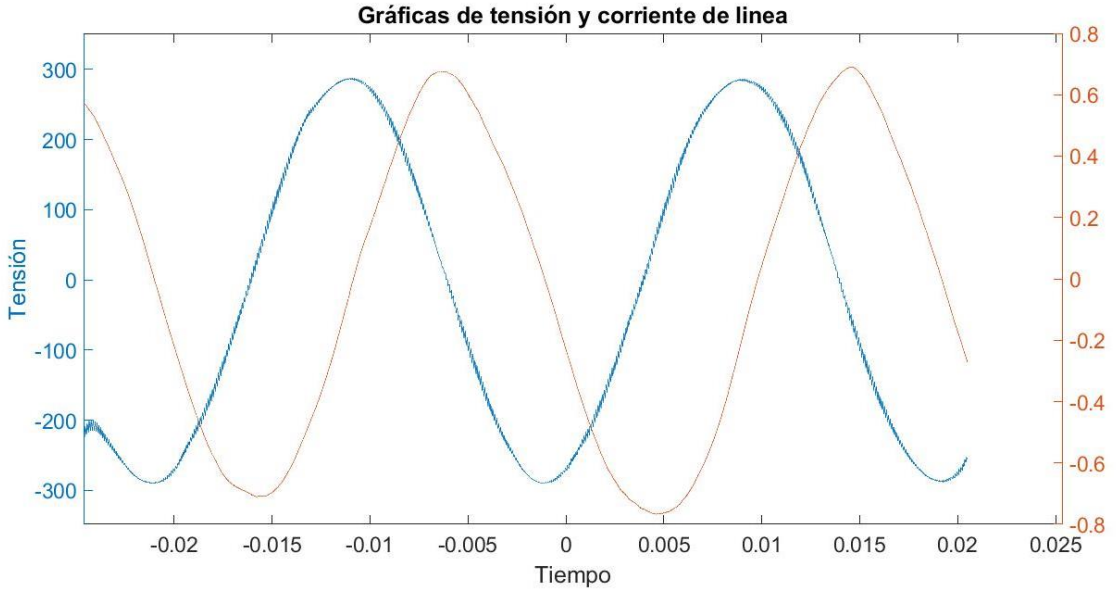


Figura 59. Tensión y corriente de línea, 25Hz, 0% carga

Por comparación con las anteriores, se puede ver la diferencia de frecuencia entre las señales anteriores, ya que para el mismo intervalo de tiempo aparecían antes únicamente un periodo y medio y, sin embargo, ahora aparecen dos periodos y medio, lo cual se podía esperar ya que a velocidades elevadas se debe introducir en el motor señales de mayor frecuencia.

Además, se puede ver la diferencia de magnitud en los voltajes ya que antes alcanzaban un valor máximo aproximado de 180 y en estos ensayos se alcanzan aproximadamente 290. Esto ocurre porque el voltaje es función de la velocidad de giro del motor ya que la modulación se calcula por relación directa de la velocidad.

El analizador de energía muestra lo siguiente:

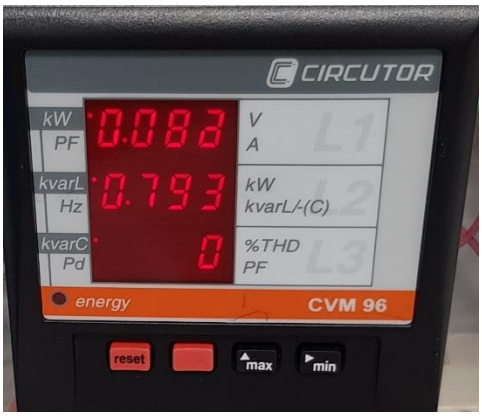


Figura 60. Analizador de energía 4

Se observa un aumento importante de la potencia reactiva con respecto al ensayo anterior a carga nula y una potencia activa baja y similar a la del otro ensayo. . El aumento de reactiva está

Figura 61. Detalle de la pantalla de la aplicación 4

asociado al gran incremento de la reactancia magnetizante al pasar de trabajar la máquina de 30 a 50 Hz aproximadamente con el mismo nivel de flujo en el circuito magnético principal. La potencia activa también aumenta ligeramente al aumentar las perdidas en el hierro de la máquina (frecuencia mayor de cambio del campo).

Puede observarse como el índice de modulación se ha visto incrementado con respecto a los ensayos anteriores al aumentar mucho la velocidad.

8.2.2. Ensayo al 50% de carga

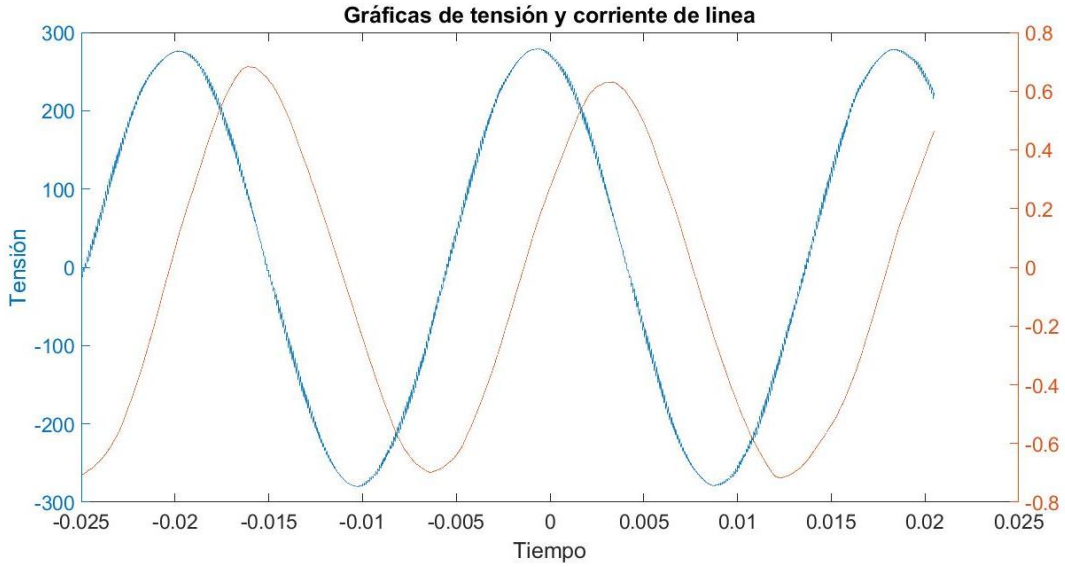


Figura 62. Tensión y corriente de línea, 25Hz, 50% carga

En esta gráfica se puede observar que la tensión es prácticamente igual a la anterior y la corriente es prácticamente la misma en módulo pero está ligeramente menos retrasada con respecto a la curva de voltaje que la anterior, provocada por la caída de tensión como se ha explicado en el primer ensayo.

Los resultados del analizador de energía y la pantalla de la interfaz de usuario son los siguientes:

Figura 64. Analizador de energía 5

Figura 63. Detalle de la pantalla de la aplicación 2

Se puede ver un aumento notable de la potencia activa con respecto al anterior, además de una bajada de la potencia reactiva debido a la bajada de tensión que se da en el bus a pesar de haber subido ligeramente el índice de modulación.

8.2.3. Ensayo al 100% de carga

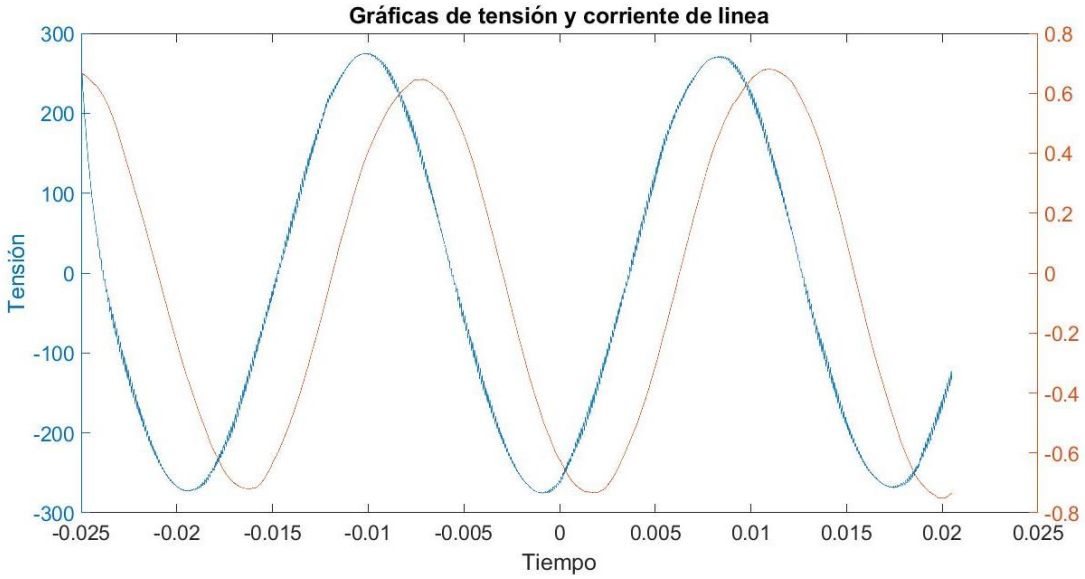


Figura 65. Tensión y corriente de línea, 25Hz, 100% carga

En esta gráfica se puede apreciar que el voltaje sigue siendo prácticamente el mismo pero la corriente se encuentra menos retrasada que en los dos casos anteriores.

El resultado del analizador de energía y la pantalla de la aplicación es el siguiente:

Figura 67. Analizador de energía 6

Figura 66. Detalle de la pantalla de la aplicación 3

En ellas se puede observar un aumento importante de la potencia activa y un pequeño descenso de la potencia reactiva, provocada por la caída de tensión, además de un pequeño aumento en la velocidad de giro del motor coherente con la explicación aportada en 8.2 relativa a las formas de operación del control principal (control de carga) y del auxiliar (frecuencia constante).

8.3. Comparativa de los resultados de ensayos realizados

Una vez hechas todas las medidas se procede a presentar todos los resultados de los distintos ensayos realizados al montaje:

8.3.1. Medida de corrientes

Para la medida de corrientes, lo que se hace es aislar un periodo de la señal y calcular el valor eficaz de cada uno de los ensayos anteriores mediante Matlab. El resultado obtenido es el siguiente:

n_s auxiliar	900rpm	1200 rpm	1500 rpm
Carga nula	1,15 A	1,61 A	1,92 A
50% de carga	1,34 A	1,57 A	1,86 A
100% carga	1,55 A	1,70 A	1,96 A

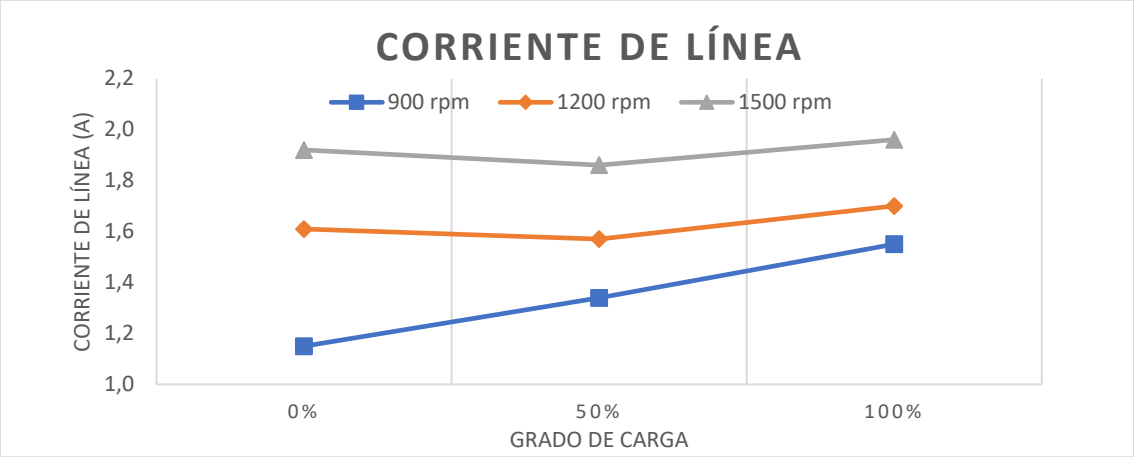


Figura 68. Comparación de las corrientes de línea

En la gráfica se puede ver como la corriente de vacío es mayor cuando sube la velocidad de giro del motor, ya que al aumentar la velocidad disminuye la importancia relativa de la caída de tensión del estator por comparación con la fem inducida y con la tensión de fase. Al utilizar un control VF, a frecuencias más elevadas la fem inducida es más cercana a la tensión de fase y se requiere una mayor corriente magnetizante, que se acerca más a la nominal conforme la frecuencia y la tensión se acercan más a la nominal. .

Al aumentar la carga manteniendo la velocidad de sincronismo de la máquina auxiliar se produce una caída de tensión en el bus de continua, especialmente grande a la velocidad más elevada que hace que la potencia reactiva caiga y, por tanto, la corriente baje. Al aumentar aún más la carga la corriente sube por el aumento de la potencia activa y de la componente activa de la corriente.

8.3.2. Medida de tensiones

La medida de tensión se ha realizado sumando las señales PWM fase-tierra de las dos señales de voltaje para a continuación aislar un periodo y calcular el valor eficaz de este periodo.

n_s auxiliar	900rpm	1200 rpm	1500 rpm
Carga nula	213,9334 V	230,5482 V	239,2824 V
50% de carga	209,4848 V	224,0403 V	229,6897 V
100% de carga	209,2575 V	219,6114 V	225,1322 V

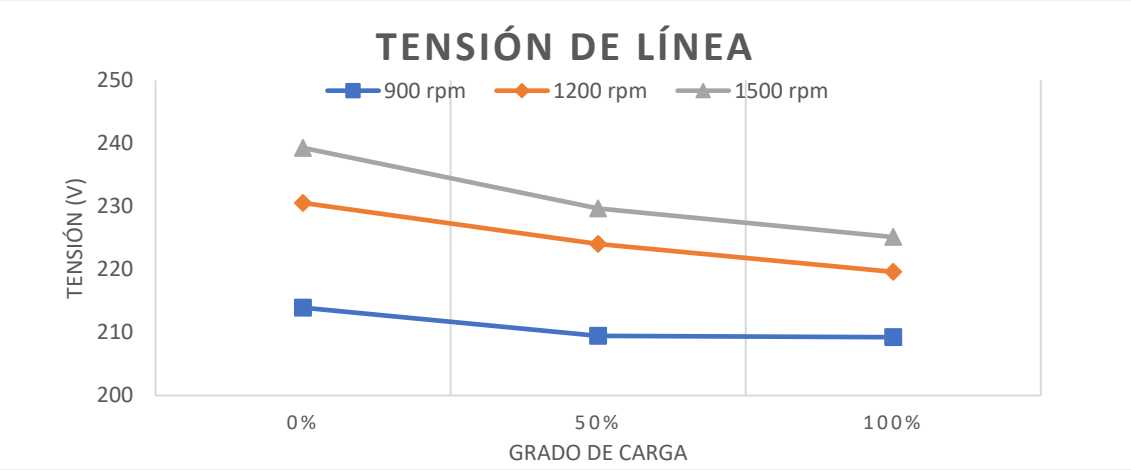


Figura 69. Comparación de las tensiones de línea

En esta gráfica se puede observar el salto de tensión que existe entre las distintas velocidades de sincronismo en el motor auxiliar debido a que la tensión que se introduce es proporcional a la velocidad del motor y, por tanto, es lógico que exista un salto al modificar la velocidad del motor. Además, se puede apreciar una bajada en el voltaje al aumentar la carga provocada por la caída de tensión el bus de continua producida por la caída de tensión en el transformador al operar a cargas elevadas, (al aumentar la potencia transformada provoca una caída de tensión en los devanados del mismo mayor). Es importante indicar que el transformador trabaja con un factor de potencia elevado que, justo para esta máquina, que se sabe que tiene una tensión de cortocircuito resistiva elevada, produce fuertes caídas de tensión.

8.3.3. Medida de velocidad

La velocidad se toma directamente de los resultados que muestra la aplicación y se muestra en revoluciones por minuto, ya que es un valor más comprensible que en pulsos del sensor por segundo.

n_s auxiliar	900rpm	1200 rpm	1500 rpm
Carga nula	896 rpm	1195 rpm	1508 rpm
50% de carga	918 rpm	1224 rpm	1533 rpm
100% de carga	937 rpm	1243 rpm	1542 rpm

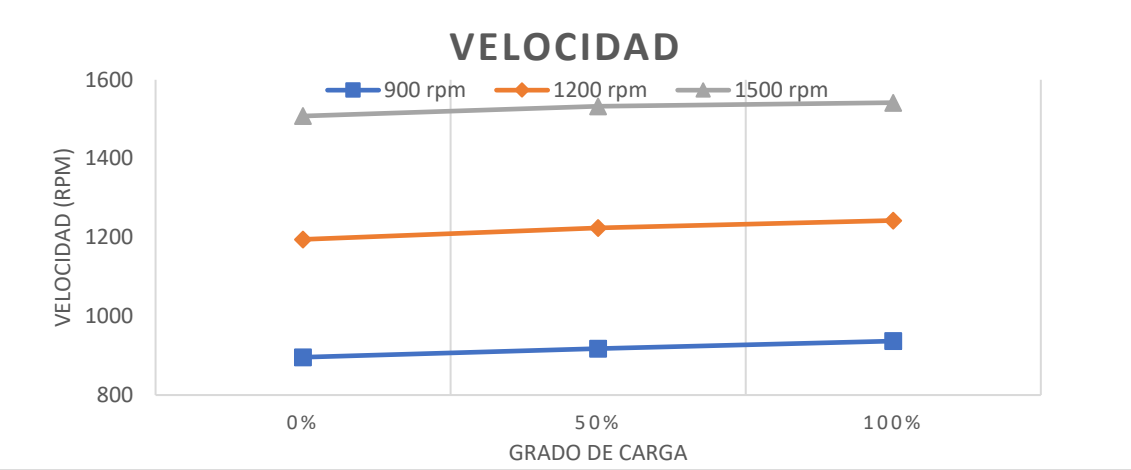


Figura 70. Comparación de las velocidades de los ensayos

En esta gráfica se puede observar el salto de velocidad medida que se da al incrementar la frecuencia en el motor auxiliar, además de un pequeño aumento de la velocidad de giro al aumentar la carga del motor principal para las tres velocidades distintas, fenómeno ya discutido en los epígrafes 8.1 y 8.2 y que obedece al modo de operación de las dos máquinas: principal en modo control de carga en régimen motor y auxiliar en modo frecuencia constante con control VF y modo regenerativo.

8.3.4. Medida del índice de modulación

El índice de modulación también se obtiene directamente de la pantalla de la aplicación.

n_s auxiliar	900rpm	1200 rpm	1500 rpm
Carga nula	58,88%	78,26%	98,72%
50% de carga	60,67%	80,14%	100%
100% de carga	62,38%	81,39%	100%

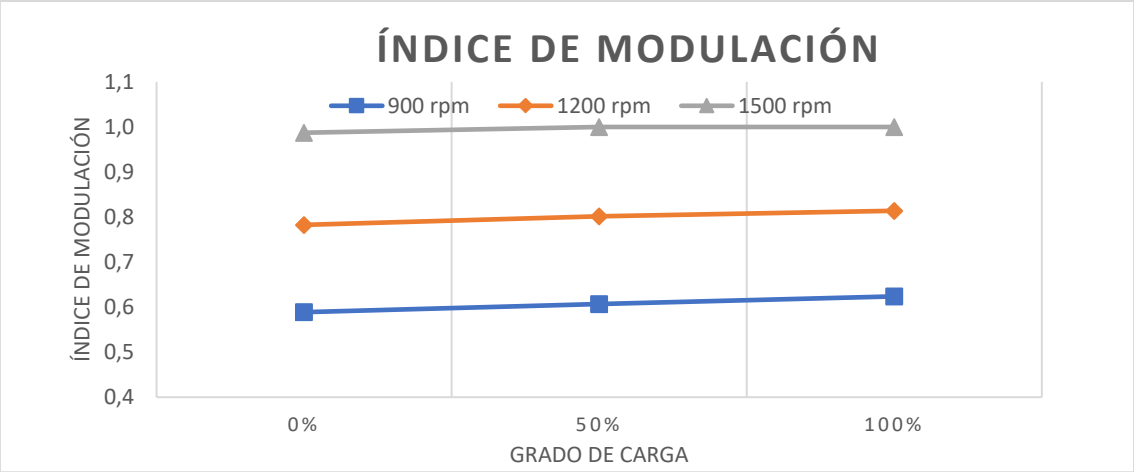


Figura 71. Comparación de los índices de modulación

El índice de modulación, al ser directamente proporcional a la velocidad, la gráfica tiene también el mismo aspecto que la anterior, aunque podemos resaltar que en este caso existe la saturación del índice de modulación para los ensayos de carga a ~ 1500 rpm.

8.3.5. Medida de potencia activa

La medida de la potencia activa se realiza con los datos extraídos del analizador de energía directamente.

n_s auxiliar	900rpm	1200 rpm	1500 rpm
Carga nula	63 W	67 W	83 W
50% de carga	192 W	244 W	285 W
100% de carga	304 W	395 W	446 W

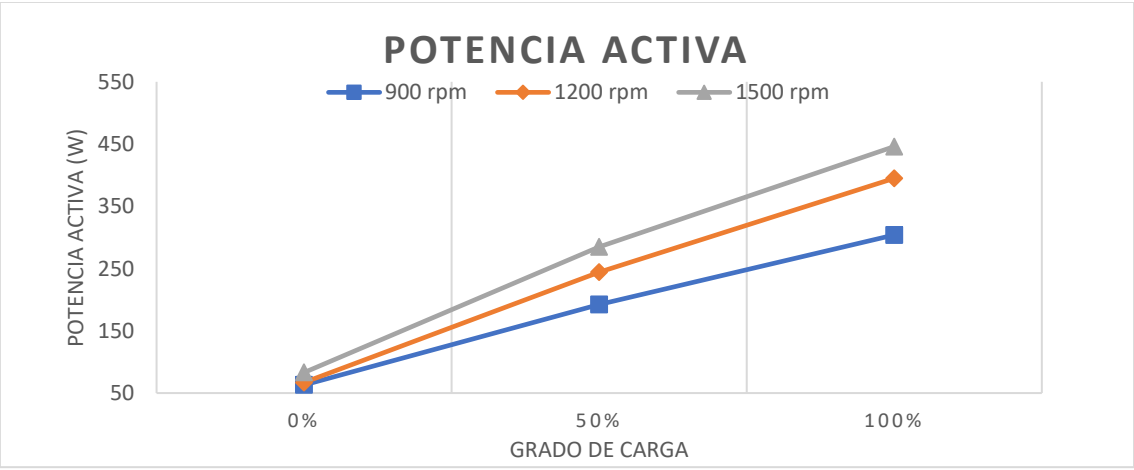


Figura 72. Comparación de la potencia activa

En esta gráfica se puede observar como la potencia activa parte de un valor cercano a 0 en todos los casos y se va viendo incrementada a medida que se incrementa el grado de carga, que era lo que se buscaba con este sistema de control. Además, se ve que a mayor velocidad mayor potencia para el mismo grado de carga solicitado, ya que este grado de carga está directamente relacionado con el par motor en relación al nominal.

8.3.6. Medida de potencia reactiva

Los datos de esta gráfica también se han obtenido directamente de los datos del analizador de energía.

n_s auxiliar	900rpm	1200 rpm	1500 rpm
Carga nula	421 VAr	639 VAr	793 VAr
50% de carga	448 VAr	558 VAr	681 VAr
100% de carga	472 VAr	510 VAr	619 VAr

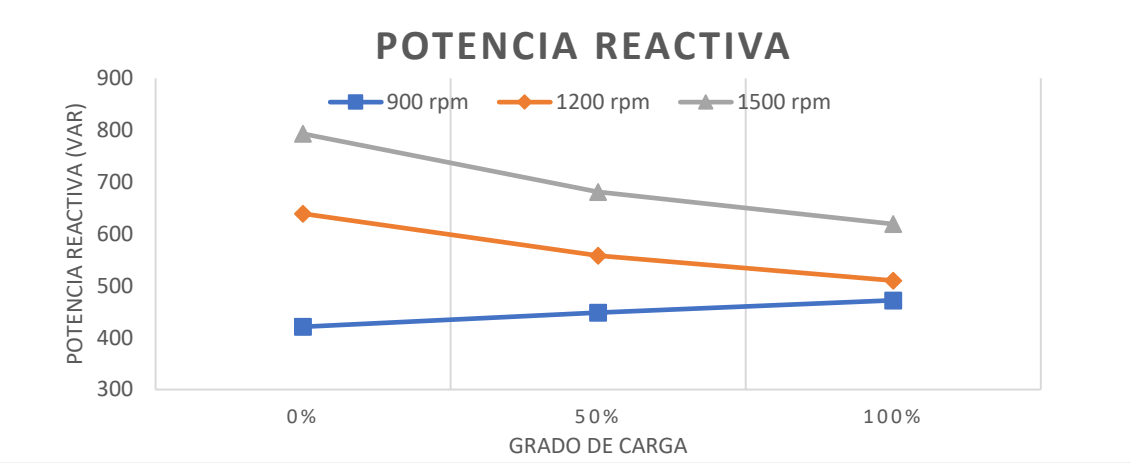


Figura 73. Comparación de la potencia reactiva

En la gráfica anterior se puede observar como la potencia reactiva aumenta con la velocidad, ya que el voltaje también sube al ser proporcional a la velocidad. Además, se observa que disminuye al aumentar el grado de carga a velocidades altas debido a la caída de potencial en el bus de continua explicada anteriormente. Sin embargo, a baja velocidad no se aprecia este fenómeno y el voltaje aumenta al variar ligeramente el índice de modulación haciéndolo también la potencia reactiva.

8.3.7. Medida de la potencia aparente

La potencia aparente se calcula como suma vectorial de la potencia activa y reactiva.

n_s auxiliar	900rpm	1200 rpm	1500 rpm
Carga nula	425,69 VA	642,50 VA	797,33 VA
50% de carga	487,41 VA	609,02 VA	738,23 VA
100% de carga	561,43 VA	645,08 VA	762,94 VA

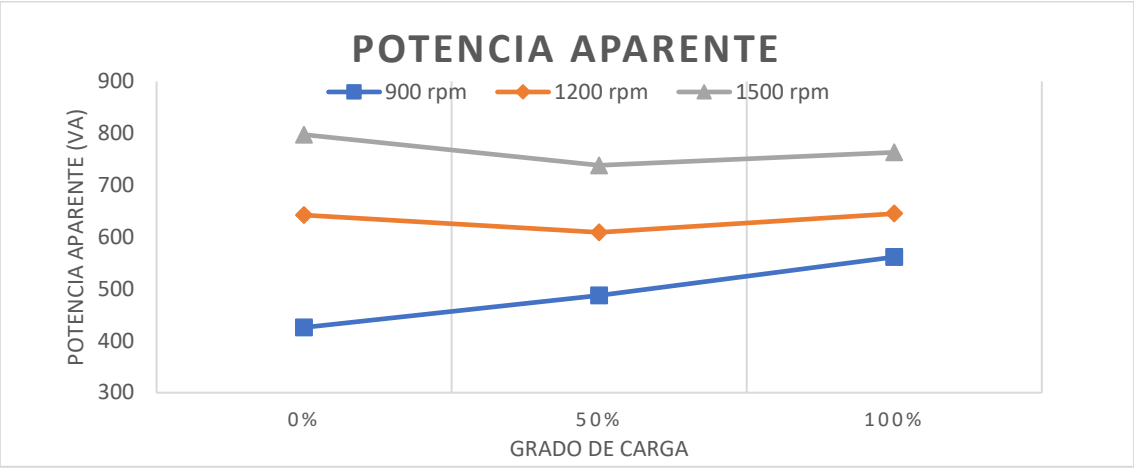


Figura 74. Comparación de la potencia aparente

De esta gráfica se puede extraer el aumento de potencia aparente al aumentar la velocidad y aumentar así el voltaje introducido al motor.

8.3.8. Desfase entre tensión y corriente

El desfase entre tensión y corriente se extrae de las medidas de potencia activa y reactiva del analizador de energía como atan(potencia reactiva/potencia activa).

n_s auxiliar	900rpm	1200 rpm	1500 rpm
Carga nula	81,49º	84,01º	84,02º
50% de carga	66,80º	66,38º	67,29º
100% de carga	57,22º	52,24º	54,23º

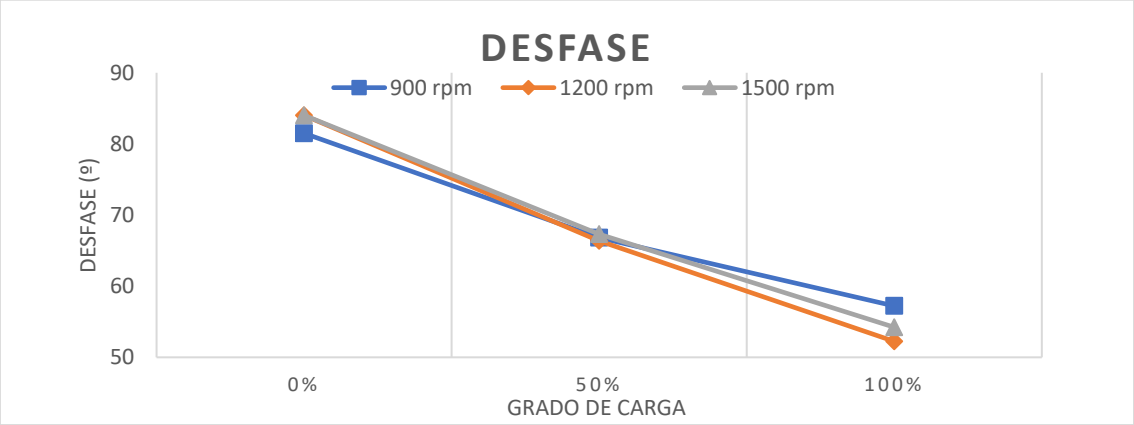


Figura 75. Comparación de los desfases en los ensayos

En esta gráfica se puede apreciar la relación que existe para las tres velocidades entre desfase y grado de carga ya que, al aumentar la carga, el motor tiene un comportamiento más activo y el desfase se hace menor.

9. APLICACIÓN ANDROID

9.1. Introducción

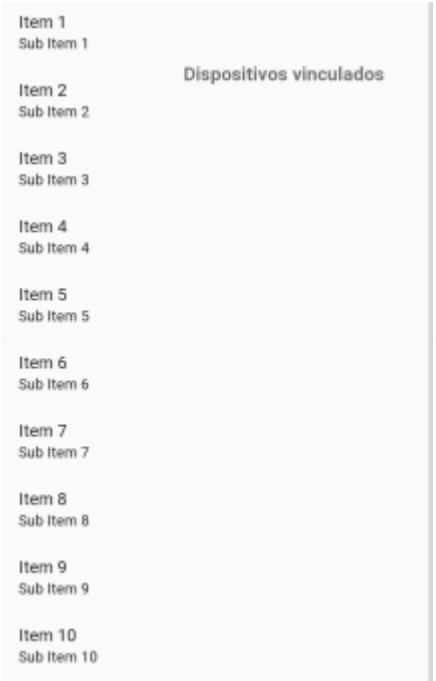
La necesidad de la creación de una aplicación Android surge de la idea de controlar un motor desde el teléfono móvil vía bluetooth. Para ello, se utiliza el entorno de desarrollo Android Studio para programar la aplicación con las necesidades requeridas para el proyecto.

En este trabajo solo se van a mostrar los aspectos clave del código de la aplicación Android por la amplia extensión el mismo. Por necesidades de espacio se ha descrito sólo la versión de la aplicación que permite el control del accionamiento, sin embargo se ha utilizado versiones distintas desarrolladas para cumplir funcionalidades específicas como el registro de datos de operación que se muestran en 7.3 y que se han utilizado para depurar el algoritmo de realimentación de posición y velocidad.

9.2. Interfaz de usuario

La aplicación consta de dos pantallas, la primera donde se muestran los dispositivos bluetooth que están conectados al dispositivo y la segunda que es la que realiza la interacción con el microcontrolador.

La primera pantalla se muestra a continuación:



Esta pantalla consta de los siguientes elementos:

-Un elemento del tipo ListView en el que se muestran los distintos dispositivos vinculados con el teléfono y al pulsar uno de ellos se intentará establecer la conexión bluetooth con él.
-Un elemento del tipo TextView donde se muestra el texto que aparece en la captura de pantalla “Dispositivos vinculados”.

Esta pantalla es la primera que saldrá al iniciar la aplicación y es la que servirá para seleccionar con que dispositivo se va a conectar.

Figura 76. Pantalla de selección de dispositivos

La segunda pantalla de la aplicación consta de los siguientes elementos:

- Tres botones, elementos del tipo Button: uno para encender el motor, otro para apagarlo y un último para desconectar el Bluetooth .
- Una barra para fijar el grado de carga o la frecuencia según en qué tipo de control estemos, es un elemento del tipo SeekBar y sirve para elegir el valor de consigna que se quiere que siga el microcontrolador.

- Diez elementos del tipo TextView, utilizados para mostrar texto. De ellos, cinco serán estáticos, no variará durante la ejecución de la aplicación su contenido y sirven para saber qué se muestra en cada TextView dinámico. Los otros cinco serán dinámicos y en ellos se mostrarán los datos recibidos por la aplicación y el valor de la SeekBar.

La pantalla de la aplicación tendrá el siguiente aspecto en funcionamiento:

Figura 77. Pantalla de control del motor

9.3. Código principal de la aplicación

En la primera pantalla, lo que se hará es iniciar el Bluetooth, mostrar en la ListView los dispositivos vinculados y cuando se pulse uno de ellos se intentará establecer conexión con él. Si se da de forma satisfactoria, se ejecutará la actividad UserInterfaz vinculada a la segunda pantalla y se podrá interactuar con el motor.

```
private AdapterView.OnItemClickListener mDeviceClickListener = new
AdapterView.OnItemClickListener() {
    public void onItemClick(AdapterView av, View v, int arg2, long
arg3) {
        String info = ((TextView) v).getText().toString();
        String address = info.substring(info.length() - 17);
        Intent i = new Intent(DispositivosBT.this,
UserInterfaz.class);
        i.putExtra(EXTRA_DEVICE_ADDRESS, address);
        startActivity(i);
    }
};
```

Esta es la parte de código que cuando se pulsa uno de los elementos de la lista obtiene la dirección Bluetooth del dispositivo e inicia la actividad UserInterfaz.

Dentro del archivo UserInterfaz.java lo que se hace es inicializar primero las variables de todos los elementos que se quieren modificar desde la aplicación, como son los paneles donde se muestra el texto que proviene del microcontrolador como TextView, la barra de selección de la carga como SeekBar y los tres botones como elementos del tipo Button.

Después, se relaciona cada elemento con el elemento que se quiera de la interfaz gráfica que se ha mostrado anteriormente con las líneas:

```
IdEncender = (Button) findViewById(R.id.IdEncender);
IdApagar = (Button) findViewById(R.id.Idapagar);
IdDesconectar = (Button) findViewById(R.id.IdDesconectar);
IdBufferIn = (TextView) findViewById(R.id.IdBufferIn);
IdBufferIn2 = (TextView) findViewById(R.id.IdBufferIn2);
IdBufferIn3 = (TextView) findViewById(R.id.IdBufferIn3);
IdBufferIn4 = (TextView) findViewById(R.id.IdBufferIn4);
Hercios = (TextView) findViewById(R.id.hercios);
seekBar = (SeekBar) findViewById(R.id.seekbar);
```

Para poder enviar información al dispositivo por Bluetooth, se debe crear la siguiente función, que será la encargada de enviar los datos al microcontrolador.

```
public void write(String input)
{
    try {
        mmOutStream.write(input.getBytes());
    }
    catch (IOException e)
    {
        Toast.makeText(getBaseContext(), "La Conexión fallo",
Toast.LENGTH_LONG).show();
        finish();
    }
}
```

Esta función recibe los datos como entrada y los envía por bluetooth. Si la transmisión falla muestra un mensaje y cierra la conexión.

Para recibir datos se utilizan las siguientes líneas de código:

```
bluetoothIn = new Handler() {
    public void handleMessage(android.os.Message msg) {
        if (msg.what == handlerState) {
            String readMessage = (String) msg.obj;
            DataStringIN.append(readMessage);

            int endOfLineIndex = DataStringIN.indexOf("#");

            if (endOfLineIndex > 0) {
                String dataInPrint = DataStringIN.substring(0,
endOfLineIndex);
                IdBufferIn.setText(dataInPrint+" %");
                DataStringIN.delete(0, DataStringIN.length());
            }
           int endOfLineIndex2 = DataStringIN.indexOf("&");
            if (endOfLineIndex2 > 0) {
                String dataInPrint = DataStringIN.substring(0,
endOfLineIndex2);
                IdBufferIn2.setText(dataInPrint);//
                DataStringIN.delete(0, DataStringIN.length());
            }
            int endOfLineIndex3 = DataStringIN.indexOf("%");
            if (endOfLineIndex3 > 0) {
                String dataInPrint = DataStringIN.substring(0,
endOfLineIndex3);
                IdBufferIn3.setText(dataInPrint+" rad/s");
                DataStringIN.delete(0, DataStringIN.length());
            }
            int endOfLineIndex4 = DataStringIN.indexOf("$");
            if (endOfLineIndex4> 0) {
                String dataInPrint = DataStringIN.substring(0,
endOfLineIndex4);
                IdBufferIn4.setText(dataInPrint+" rpm");
                DataStringIN.delete(0, DataStringIN.length());
            }
        }
    }
};
```

Las primeras 5 líneas son las encargadas de comprobar si ha llegado algún dato y lo guarda en la variable DataStringIn. Las otras líneas lo que hacen es comprobar si existe un carácter de cuatro distintos que se han elegido para codificar los cuatro datos distintos a recibir desde el microcontrolador vía Bluetooth y, si lo hace, quiere decir que el dato pertenece al campo al que se ha asociado ese carácter. Entonces, dentro de cada bucle primero borra el carácter de identificación y luego pone los datos en el cuadro de texto correspondiente. Por último, borra los datos almacenados para poder recibir los nuevos datos y este código se repite 4 veces para los cuatro cuadros de texto cambiando el carácter de identificación de panel.

Para determinar que harán los tres botones distintos se utilizan las siguientes líneas :

```
IdEncender.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v)
    {
        MyConexionBT.write("ON");
    }
});

IdApagar.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {
        MyConexionBT.write("OF");
    }
});

IdDesconectar.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {
        MyConexionBT.write("OF");
        if (btSocket!=null)
        {
            try {btSocket.close();}
            catch (IOException e)
            { Toast.makeText(getBaseContext(), "Error",
Toast.LENGTH_SHORT).show();;}
        }
        finish();
    }
});
```

Para cada botón se aplica el método setOnClickListener asociado a cada uno de los tres botones y con la línea MyConexionBT.write(“texto a enviar”) se envía el texto que se quiere, en este caso ON para encender y OF para apagar.

El botón desconectar lo que hace es, primero, enviar la orden de paro al motor y luego cerrar la conexión con el comando btSocket.close().

En las siguientes líneas se mostrará lo necesario para operar la SeekBar para seleccionar el grado de carga:

```
seekBar.setOnSeekBarChangeListener(new
SeekBar.OnSeekBarChangeListener() {
    @Override
    public void onProgressChanged(SeekBar seekBar, int progress,
boolean fromUser) {
        proceso=progress*99/100;
       Hercios.setText("" + progress +"%");

    }


    @Override
    public void onStartTrackingTouch(SeekBar seekBar) {

    }

    @Override
    public void onStopTrackingTouch(SeekBar seekBar) {
        MyConexionBT.write(""+proceso);
    }
});
```

En primer lugar, hay que llamar al método setOnSeekBarChangeListener que detecta cuando se produce un cambio en la SeekBar dentro de la función interna del método onProgressChanged mientras mueves la barra. Sin soltar el dedo de la pantalla, se calcula el valor que se enviará al microcontrolador escalando la variable progress que va de 0 a 100 a una que vaya de 0 a 99, ya que el código del Esp32 solo permite 2 números. Además, se muestra el valor seleccionado en un cuadro de texto acompañado del símbolo de % con la línea Hercios.setText("" + progress +"%").

Al soltar el dedo se activa la función onStopTrackingTouch y se envía el valor de progreso entre 0 y 99 al microcontrolador para su lectura.

Se ha decidido no mostrar la parte de código que conecta la aplicación por Bluetooth ya que de esa parte hay infinidad de ejemplos por internet y no aporta nada adicional a lo que se está haciendo en este proyecto.

10. CONCLUSIONES

La finalidad de este proyecto ha sido, desde que se empezó su realización, conseguir gobernar un motor de inducción desde el teléfono móvil utilizando un microcontrolador para controlar un inversor trifásico de un banco de ensayos. Para ello, se han tenido que realizar multitud de tareas de distinta dificultad.

El TFG contempla el desarrollo de un sistema de control de inversor basado en microcontrolador para un banco de ensayos de accionamientos eléctricos. El sistema debe estar basado en un microcontrolador tipo ESP32 o similar y el desarrollo debe contemplar:

- Desarrollo y pruebas de un algoritmo de modulación por ancho de pulso por vector espacial, tal y como se explica en los capítulos 6.1 y 6.2.
- Desarrollo de la interfaz de transmisión de datos para operación remota (tablet-PC), realizado como se muestra en el punto 6.3.
- Desarrollo de la interfaz de visualización local (LCD) explicada en el capítulo 6.4.
- Desarrollo y pruebas de un algoritmo de control con realimentación de velocidad/posición/corriente para la implementación del control en carga, tal y como se explica en el capítulo 7.
- Desarrollo de una App Android para operación remota del banco de ensayos, realizada de la forma que se muestra en el punto 9 de la memoria.

Para la realización de este trabajo se han requerido conocimientos multidisciplinares de distintos ámbitos de aplicación como son la parte eléctrica, la electrónica, la informática y la automatización. Al tratarse de un trabajo con tal variedad de conocimientos necesarios, uno se da cuenta de la importancia de conocer distintas áreas de trabajo y la necesidad de ver los proyectos como algo global donde está todo relacionado.

Además de esto, la realización de este trabajo ha servido para darse cuenta de las dificultades que presenta la investigación cuando uno se enfrenta a problemas nuevos y tiene la necesidad de solucionarlos por caminos que no sabe a dónde le van a llevar y donde hay que buscar información e intentar saber en qué lugar pueden existir errores y solventarlos.

Por eso, creo que el hecho de operar un motor desde un microcontrolador, tener que aprender el lenguaje para programarlo, ser capaz de generar con él tres señales trifásicas y operar un motor, así como empezar con la creación de aplicaciones para Android y ser capaz de operar el motor vía Bluetooth son habilidades importantes para un ingeniero y que han sido desarrolladas gracias a la realización de este trabajo.

11. BIBLIOGRAFÍA

[1] Yu, Zhenyu (1999). Texas Instruments. http://www.ti.com/lit/an/spra524/spra524.pdf

[2] Marwan A.A. Badran, Ahmad M. Tahir y Waleed F. Faris (2013). Semantic Scholar. International Islamic University Malaysia. https://pdfs.semanticscholar.org/7907/f61426b020ad3bbf258b66c38b958612d65c.pdf

[3] Belkacem Belkacem, Lahouari Abdelhakem-Koridak, Mostefa Rahli (2017). Journal of Power Technologies. Oran-Algeria. http://papers.itc.pw.edu.pl/index.php/JPT/article/download/898/778

[4] Santhi M (2015).Shodganga a reservoir of Indian Theses. Anna University. https://shodhganga.inflibnet.ac.in/bitstream/10603/49367/12/12_appendix.pdf

[5] Blum, J. (2013) , *EXPLORING ARDUINO,* Indianápolis, Wiley.

[6] Motion Control (2010). Info PLC. http://www.infoplc.net/blog4/2010/02/27/el-convertidor-de-frecuencia-ii-control-escalar/

[7] Rodríguez Pozueta,M. A. (2017). Frenado de Máquinas Asíncronas o de Inducción. Cantabria. https://personales.unican.es/rodrigma/PDFs/Frenado%20asincronas_Web.pdf

[8] Mjrovai (2018). Playing With the ESP32 on Arduino IDE https://www.instructables.com/id/IOT-Made-Simple-Playing-With-the-ESP32-on-Arduino-/

[9] Android Developers. Bluetooth for Android Studio. https://developer.android.com/guide/topics/connectivity/bluetooth?hl=es-419

[10] Innova Domotics (2013). Comunicación Bluetooth. Ecuador. http://www.innovadomotics.com/mn-tuto/mn-android/proyectos/23-and-cnm-bt.html

[11] Android Developers. SeekBar. https://developer.android.com/reference/android/widget/SeekBar

[12] Martínez Román, J. A. (2018) , Apuntes y Transparencias de Ampliación de Equipos e Instalaciones Eléctricas, 1º de Master en Ingenierías Industriales, Universidad Politécnica de Valencia.

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIEROS INDUSTRIALES VALENCIA

DESARROLLO DE UN SISTEMA DE CONTROL DE INVERSOR BASADO EN MICROCONTROLADOR PARA UN BANCO DE ENSAYOS DE ACCIONAMIENTOS ELÉCTRICOS

PRESUPUESTO

AUTOR: JORGE MIRALLES GUIMERÁ

TUTOR: JAVIER ANDRÉS MARTÍNEZ ROMÁN

Curso Académico: 2018-19

En la realización de un proyecto es necesario conocer el coste asociado al mismo, para ello se redacta un presupuesto detallado para justificar los costes implicados en la realización.

Para ello se ha divido la integridad del proyecto en cinco unidades diferenciadas con la finalidad de detallar el coste de cada una de las partes que integran este Trabajo Final de Grado.

UNIDAD DE OBRA 1. CIRCUITO DE CONTROL				
TIPO DE RECURSO	**Concepto**	**Precio unitario**	**Cantidad**	**Coste**
MATERIALES	Microcontrolador ESP32	6,32 €/u	1 u	6,32 €
	Sensor de corriente de efecto Hall	12,46 €/u	4 u	49,84 €
	Sensor absoluto de posición	33,20 €/u	1 u	33,20 €
	Pantalla LCD	1,37 €/u	1 u	1,37 €
	Manguera de cables de cobre 6x0,2mm2	0,37 €/m	1 m	0,19 €
	Cable flexible de cobre (1,5)mm2	0,53 €/m	4 m	2,12 €
	Conectores de seguridad tipo banana	3,68 €/u	7 u	25,76 €
	Cable de cobre esmaltado (1,5mm2)	0,67 €/m	0,2 m	0,13 €
	Cable apantallado 0,35 mm2	1,32 €/m	0,3 m	0,40 €
	Condensadores cerámicos de 47pF	0,07 €/u	4 u	0,28 €
	Resistencias de 330 Ω	0,10 €/u	4 u	0,40 €
	Conectores pin hembra	0,01 €/u	54 u	0,54 €
	Conectores pin macho	0,01 €/u	52 u	0,52 €
	Placa de montaje de 15x10cm	3,21 €/u	1 u	3,21 €
	Caja de PVC de 40x25cm	10,37 €/u	1 u	10,37 €
	Cable plano 34 cables	2,63 €/u	1 u	2,63 €
HUMANOS	Estudios previos	17,00 €/h	15 h	255,00 €
	Montaje y soldaduras	17,00 €/h	5 h	85,00 €
	Pruebas de la circuitería	17,00 €/h	4 h	68,00 €
COSTE UNIDAD DE OBRA 1				**545,28 €**

UNIDAD DE OBRA 2. MONTAJE DEL CIRCUITO DE POTENCIA				
TIPO DE RECURSO	**Concepto**	**Precio unitario**	**Cantidad**	**Coste**
MATERIALES	Motor Alecop AL 306	407,30 €/u	1 u	407,30 €
	Motor trifásico Siemens	102,30 €/u	1 u	102,30 €
	Inversor trifásico Siemens Micromaster	257,00 €/u	1 u	257,00 €
	Inversor para prototipos IHM028-V1	298,30 €/u	1 u	298,30 €
	Analizador de energía	221,00 €/u	1 u	221,00 €
	Caja de resistencias	67,45 €/u	1 u	67,45 €
	Manguera cable cobre 3x(1,5)mm2	1,20 €/m	3 m	3,60 €
	Manguera cable cobre 2x(1,5)mm2	1,06 €/m	2 m	2,12 €
	Cables 20cm 1,5mm2 conectores banana	3,27 €/u	6 u	19,62 €
	Conectores tipo banana macho	3,68 €/u	13 u	47,84 €
	Clavija de enchufe macho	2,65 €/u	1 u	2,65 €
	Interruptor automático 50A	35,15 €/u	1 u	35,15 €
	Transformador 3KVA	226,00 €/u	1 u	226,00 €
	Barra de aluminio para bancada	13,20 €/u	2 u	26,40 €
	Acoplador de los motores	35,40 €/u	1 u	35,40 €
HUMANOS	Montaje de los motores en la bancada	17,00 €/h	6 h	102,00 €
	Estudios previos	17,00 €/h	5 h	85,00 €
	Montaje del todos los elementos	17,00 €/h	7 h	119,00 €
	Pruebas al montaje	17,00 €/h	4 h	68,00 €
COSTE UNIDAD DE OBRA 2				**2.126,13 €**

UNIDAD DE OBRA 3. DESARROLLO DEL CÓDIGO DEL MICROCONTROLADOR				
TIPO DE RECURSO	**Concepto**	**Precio unitario**	**Cantidad**	**Coste**
MATERIALES	Ordenador personal	12,00 €/mes	5 meses	60,00 €
	Software Arduino IDE	0,00 €/u	1 u	0,00 €
HUMANOS	Estudios previos	16,00 €/h	15 h	240,00 €
	Implementación y pruebas del SVPWM	16,00 €/h	20 h	320,00 €
	Implementación del control Bluetooth	16,00 €/h	7 h	112,00 €
	Implementación del control en carga	16,00 €/h	6 h	96,00 €
COSTE UNIDAD DE OBRA 3				**828,00 €**

UNIDAD DE OBRA 4. DESARROLLO DE LA APLICACIÓN ANDROID				
TIPO DE RECURSO	**Concepto**	**Precio unitario**	**Cantidad**	**Coste**
MATERIALES	Ordenador personal	12,00 €/mes	2 meses	24,00 €
	Teléfono Huawei P10 Lite	4,00 €/mes	2 meses	8,00 €
	Software Android Studio	0,00 €/u	1 u	0,00 €
HUMANOS	Estudios previos	16,00 €/h	20 h	320,00 €
	Desarrollo de la aplicación	16,00 €/h	15 h	240,00 €
	Pruebas a la aplicación	16,00 €/h	5 h	80,00 €
COSTE UNIDAD DE OBRA 4				**672,00 €**

UNIDAD DE OBRA 5. REDACCIÓN DE LA MEMORIA				
TIPO DE RECURSO	**Concepto**	**Precio unitario**	**Cantidad**	**Coste**
MATERIALES	Ordenador personal	12,00 €/mes	2 meses	24,00 €
HUMANOS	Estudios previos	16,00 €/h	80 h	1.280,00 €
COSTE UNIDAD DE OBRA 5				**1.304,00 €**

Una vez detallado el coste de cada una de las unidades implicadas en la realización de este trabajo se procede a realizar el presupuesto de la ejecución del Trabajo Final de Grado, quedando del siguiente modo:

PRESUPUESTO	
CONCEPTO	COSTE
UNIDAD DE OBRA 1	545,00 €
UNIDAD DE OBRA 2	2.126,13 €
UNIDAD DE OBRA 3	828,00 €
UNIDAD DE OBRA 4	672,00 €
UNIDAD DE OBRA 5	1.304,00 €
Presupuesto de ejecución material	**5.475,13 €**
Gastos generales (16%)	876,02 €
Beneficio industrial (6%)	328,51 €
Presupuesto de ejecución por contrata	**6.679,65 €**
IVA (21%)	1.402,73 €
Presupuesto base licitación	**8.082,38 €**

El presupuesto asciende a un total de **OCHO MIL OCHENTA Y DOS EUROS CON TREINTA Y OCHO CÉNTIMOS**.