Proyecto Fin de Carrera

ANÁLISIS E IMPLEMENTACIÓN EN FPGA DE UN SISTEMA DE IDENTIFICACIÓN DE CARGAS DE INDUCCIÓN EN TIEMPO REAL BASADO EN EL ALGORITMO DE MÍNIMOS CUADRADOS

Autor

Raúl Zaborras Sanz

Directores

Mr. Oscar Jiménez Navascués

Dr. Oscar Lucía Gil

Departamento de Ingeniería Electrónica y Comunicaciones

Escuela de Ingeniería y Arquitectura

Universidad de Zaragoza

2012

ANÁLISIS E IMPLEMENTACIÓN EN FPGA DE UN SISTEMA DE IDENTIFICACIÓN DE CARGAS DE INDUCCIÓN EN TIEMPO REAL BASADO EN EL ALGORITMO DE MÍNIMOS CUADRADOS

RESUMEN

En la actualidad, la tecnología de calentamiento por inducción doméstico está desplazando a otras tecnologías de calentamiento clásicas como la resistiva o el gas debido a sus ventajas en cuanto a rapidez, eficiencia y limpieza. Esto ha sido posible gracias a los avances realizados en el área de la electrónica de potencia, que han permitido el diseño de etapas electrónicas compactas, versátiles y robustas.

Con el objetivo de alcanzar un elevado rendimiento, generalmente se utilizan etapas de potencia resonantes. Una característica vital de dichas etapas es que el tanque resonante determina el punto de operación. La impedancia de la carga de inducción, formada por el sistema inductor-recipiente, presenta una gran variabilidad debido a que depende de múltiples factores como la frecuencia de trabajo, la temperatura del recipiente, el alineamiento del recipiente con el inductor o el material del recipiente. Dado que la impedancia puede cambiar durante el funcionamiento del sistema, y que ello repercute tanto en el punto de operación de la etapa inversora como en su área de operación segura, se hace deseable obtener una identificación en tiempo real de dicha impedancia.

En el presente proyecto fin de carrera se ha analizado el algoritmo mínimos cuadrados aplicado a la identificación de la carga de inducción. Dicho análisis se ha realizado mediante el uso de la herramienta MATLAB, analizándose diversos modos de implementación del algoritmo con el objetivo de seleccionar tanto la forma adecuada del mismo como los parámetros a utilizar en su implementación. En esta fase se ha discretizado la carga, se han desarrollado las distintas formas del algoritmo y se han realizado las simulaciones pertinentes.

La forma elegida se ha implementado en Vivado HLS, para lo cual el algoritmo se ha desarrollado en lenguaje C. Posteriormente la implementación del mismo se ha optimizado mediante dicha herramienta de software y los resultados obtenidos con medidas experimentales se han representado en MATLAB para un último análisis.

AGRADECIMIENTOS

Quisiera aprovechar para agradecer a todas las personas que se han visto involucradas, directa o indirectamente en la realización del presente proyecto.

En primer lugar agradecer a José Miguel Burdío y a BSH la posibilidad de incorporarme a su grupo de investigación para el desarrollo de este proyecto. A Luis Ángel Barragán por introducirme en el proyecto y por aportar su experiencia para la resolución de los obstáculos que han surgido en el camino. A mis compañeros de laboratorio por el buen ambiente y disposición en el día a día. A mis amigos y compañeros de carrera por su apoyo incondicional en los momentos difíciles.

Me gustaría agradecer de forma especial la dedicación de mis tutores Oscar Jiménez y Oscar Lucía, ya que gracias a sus consejos, conocimientos y paciencia este proyecto ha sido posible. Por último a mis padres por su apoyo a lo largo de toda la carrera, que me ha permitido llegar hasta aquí.

A todos ellos gracias.

ÍNDICE

LISTA DE SÍMBOLOS

R	Resistencia equivalente del modelo inductor-recipiente (Ω)
L	Inductancia equivalente del modelo inductor-recipiente (µH)
C	Condensador de resonancia (nF)
D	Ciclo de servicio
f_{sw}	Frecuencia de conmutación del sistema (kHz)
T_{sw}	Periodo de conmutación de los transistores (ms)
F_S	Frecuencia de muestreo (MHz)
T_S	Periodo de muestreo (s)
T_{STOP}	Tiempo de simulación (ms)
b	Número de bits
V_{BUS}	Tensión de bus (V)
V_0	Tensión obtenida del semipuente (V)
I_L	Intensidad que circula por el inductor (A)
V_C	Tensión en el condensador de resonancia (V)
T_C	Tiempo de cómputo del algoritmo (µs)
II	Intervalo de inicio (ciclos de reloj)
PDM	*Pulse Density Modulation*
SW	*Square Wave*
ADC	*Analog-to-Digital Converter*
ADC	*Asymmetrical Duty Cycle*
ZVS	*Zero-Voltage Switching*
FPGA	*Field Programmable Gate Array*
VHDL	*Very High Speed Integrated Circuit Hardware Description Language*
HLL	*High Level Languages*

RTL	*Register-Transfer Level*
ASIC	*Application-Specific Integrated Circuit*
DSP48E	*Digital Signal Processing logic Element*
RAM	*Random-Access Memory*
GUI	*Graphical User Interface*

1. INTRODUCCIÓN

El presente proyecto se ha realizado dentro del marco de colaboración en proyectos de I+D+i entre la empresa BSH Electrodomésticos España y la Universidad de Zaragoza. Es de especial interés el desarrollar la tecnología para poder competir en el mercado actual, por lo que este tipo de proyectos de investigación son de gran utilidad para la empresa. Se da gracias a ambas partes y a su estrecha colaboración, la oportunidad de realizar proyectos con aplicaciones reales en el ámbito de una empresa antes de acabar la carrera. Esto supone una primera toma de contacto con el mundo real y una preparación previa a la salida al mercado laboral, muy recomendable e importante desde el punto de vista del estudiante.

El calentamiento por inducción en el ámbito doméstico es una tecnología que se ha extendido rápidamente en estos últimos años debido principalmente a su mayor eficiencia, seguridad y rapidez en comparación con tecnologías anteriores como la resistiva o el gas [1]. Esto ha sido posible gracias a los avances realizados en el área de la electrónica de potencia, que han permitido el diseño de etapas electrónicas compactas, versátiles y robustas. Con el objetivo de alcanzar un elevado rendimiento, generalmente los fabricantes optan por utilizar etapas de potencia resonantes. Una característica vital de dichas etapas es que el tanque resonante determina el punto de operación, además del área de operación segura. La impedancia de la carga de inducción presenta una gran variabilidad debido a que depende de múltiples factores como la frecuencia de trabajo, la temperatura y el material del recipiente o el alineamiento del recipiente con el inductor. Dado que la impedancia puede cambiar durante el funcionamiento del sistema, y que ello repercute en el punto de operación de la etapa inversora, se hace deseable obtener una identificación en tiempo real de dicha impedancia. Debido a esto, durante los últimos años el grupo de investigación de Electrónica de Potencia y Microelectrónica de la Universidad de Zaragoza ha realizado un esfuerzo investigador en el ámbito de la caracterización en tiempo real de la carga de inducción [2-5].

Con la realización del presente proyecto se pretende estudiar el comportamiento de un sistema de identificación de cargas de inducción en tiempo real basado en el algoritmo mínimos cuadrados. Se analizaran diversas formas del mismo para implementar aquella con la que se obtengan mejores resultados.

1.1 Funcionamiento de la cocina de inducción

Un sistema de calentamiento por inducción doméstico se divide en cuatro etapas, tal y como muestra la Figura 1.1.

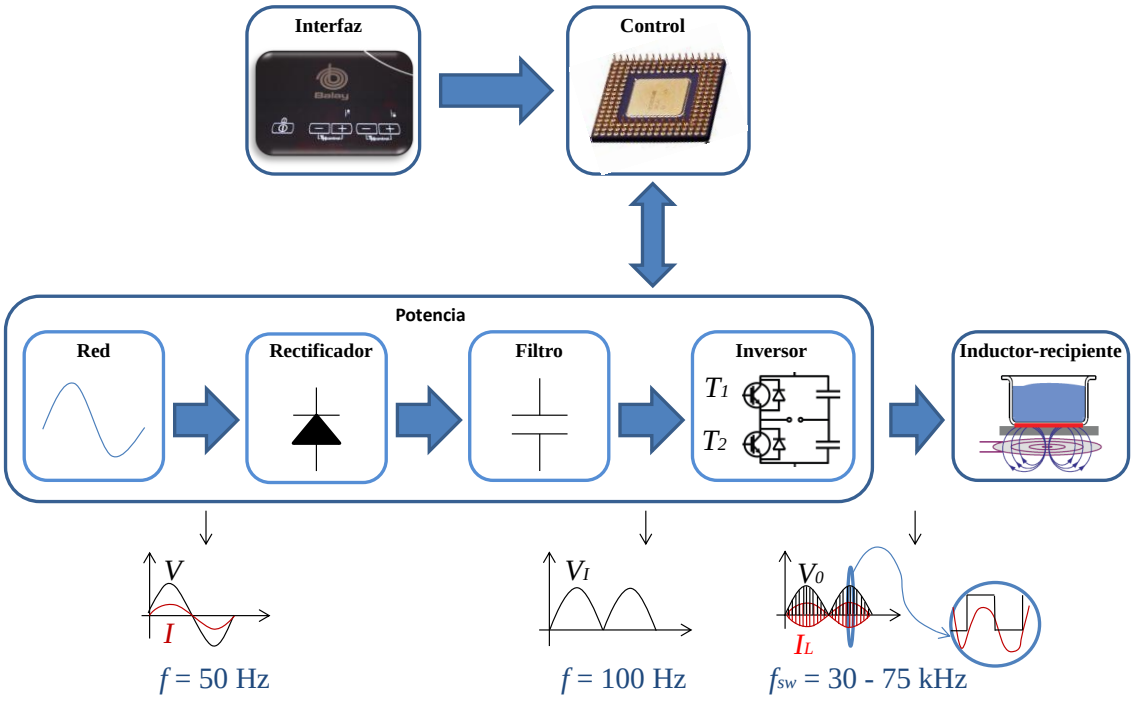


Figura 1.1. Esquema básico de una cocina de inducción.

- Interfaz: Permite al usuario elegir la potencia a la que trabajará la cocina de inducción. También sirve para visualizar alertas o la potencia que utiliza actualmente el sistema.

- Etapa de control: Consistente en un microprocesador y un ASIC (*Application-Specific Integrated Circuit*) para la comunicación de la interfaz con la cocina de inducción, el control de su funcionamiento y la toma de medidas.

- Etapa de potencia: Adapta la potencia obtenida de la red a baja frecuencia para suministrarla a la carga. En una primera etapa rectifica y filtra la tensión, duplicando la frecuencia de la misma. Se utiliza un valor de condensador pequeño, obteniendo un gran rizado en la tensión de bus V_{BUS}. La consecuencia es un comportamiento similar al de una resistencia, mejorando así el factor de potencia.

  El inversor DC-AC se encarga de convertir la tensión de corriente continua a una de corriente alterna de media frecuencia f_{sw}, la cual es inyectada al inductor. En la Figura 1.2 se observa como está modulada dicha intensidad I_L y la tensión en la carga V_0, además de conmutaciones de los transistores T_1 y T_2 con un periodo de conmutación T_{sw}. De los tipos de modulación posible, SW (*Square Wave*), ADC (*Asymmetrical Duty Cycle*) y PDM (*Pulse Density Modulation*); se escogió la primera de ellas. Esta se basa en modificar la frecuencia de conmutación del inversor para obtener la potencia objetivo marcada por el usuario.

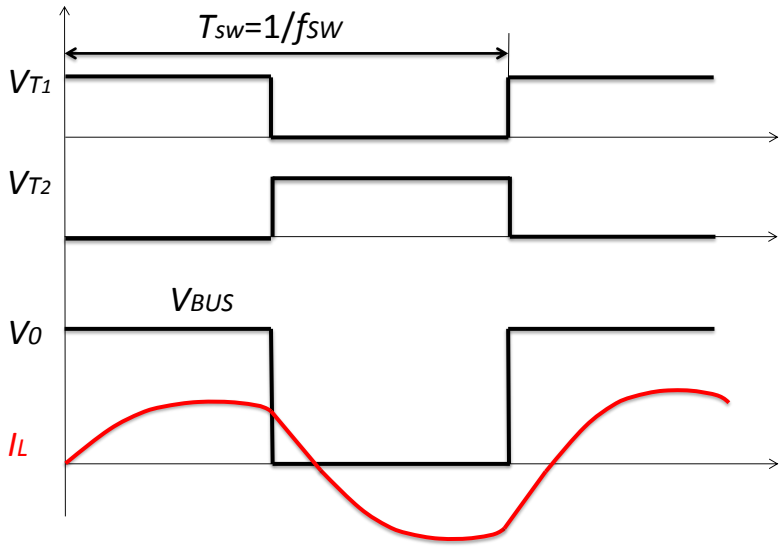


Figura 1.2. Formas de onda del semipuente.

En el caso de este proyecto se ha elegido la etapa semipuente resonante serie, mostrada en la Figura 1.3, de entre las diversas topologías existentes, ya que se consigue un buen compromiso entre coste y prestaciones. Dicha etapa trabaja siempre por encima de resonancia para conseguir un tipo de conmutación denominada ZVS (*Zero-Voltage Switching*). De esta forma se consigue que la activación de los transistores se produzca cuando la tensión es cero para reducir las pérdidas. Los transistores disponen de condensadores de snubber C_S para reducir las pérdidas en el paso a corte.

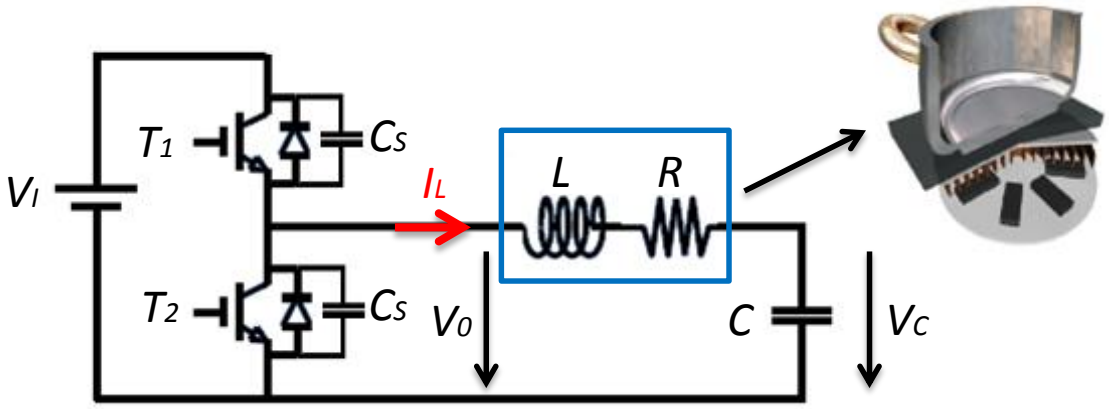


Figura 1.3. Modelo equivalente del sistema inductor-recipiente.

- Sistema inductor-recipiente: Comprende al inductor y al recipiente. El modelo más comúnmente utilizado de la carga de inducción está compuesto por una resistencia *R* y una inductancia *L* conectadas en serie. Además, se dispone con condensador de resonancia *C* en serie para formar el tanque resonante. En la Figura 1.3 anteriormente mencionada aparece el modelo equivalente.

  En esta etapa se transforma la potencia eléctrica en potencia calorífica al crearse un campo magnético en el inductor sobre el que se coloca el recipiente. Dicho campo se forma al excitar el inductor mediante el inversor semipuente resonante.

La corriente alterna que circula por él genera un campo magnético en sus alrededores y al introducir el recipiente aparecen corrientes inducidas que se oponen al campo. El calentamiento se produce debido a dos fenómenos [6]:

- Efecto Joule: Consiste en la aparición de corrientes inducidas en un material conductor debido al campo magnético por el que se ve afectado. Es el mecanismo de disipación predominante en el calentamiento por inducción doméstico.

- Histéresis magnética: Debido a la condición ferromagnética de un material se produce disipación de energía a causa de procesos de imanación y desimanación.

1.2 FPGA

Una FPGA (*Field Programmable Gate Array*) es un dispositivo semiconductor que contiene bloques de lógica cuya interconexión y funcionalidad puede ser configurada *'in situ'* mediante un lenguaje de descripción especializado. En el caso de este proyecto el lenguaje elegido ha sido VHDL (*Very High Speed Integrated Circuit Hardware Description Language*). Para ello se ha utilizado la herramienta de síntesis Vivado HLS desarrollada por Xilinx, que a partir de un código en lenguaje C, genera una descripción en VHDL.

Aunque antes las FPGAs solían seleccionarse para diseños de bajo volumen, velocidad y complejidad, en la actualidad pueden alcanzar los 500 MHz y algunas incorporan dispositivos como procesadores o DSPs (*Digital Signal Processing*) [7]. La principal ventaja de las FPGAs es la capacidad de reprogramación, que unida al bajo coste para pequeños volúmenes la hace ideal para el prototipado de casi cualquier tipo de sistema. Es por estos factores que el dispositivo que se ha elegido es una FPGA, más concretamente la Spartan-6 XC6SLX45 FPGA de Xilinx sobre la placa Atlys de DIGILENT [8].

1.3 Motivación

La motivación del presente proyecto reside en la influencia en el punto de operación del tanque resonante de un sistema de calentamiento por inducción doméstico. También influye en la determinación del área segura de funcionamiento, por lo que el interés deja de ser meramente energético.

Dado que la variabilidad del tanque es alta, puesto que depende de factores como la geometría, la temperatura de los materiales, etc.; el desarrollo de un sistema de identificación de cargas en tiempo real reportaría beneficios tanto económicos como medioambientales al optimizar el funcionamiento de la etapa de potencia.

1.4 Objetivo

El objetivo principal es el análisis e implementación del algoritmo de mínimos cuadrados en FPGA aplicado a la identificación en tiempo real de la impedancia de un sistema de calentamiento por inducción.

Para ello se compararán distintas formas de implementar y discretizar el algoritmo de mínimos cuadrados, así como variaciones para ajustar las simulaciones al sistema real y sus restricciones, tales como el número de muestras por conmutación o los errores de discretización al pasar de un sistema continuo a uno discreto.

1.5 Alcance

Para la elaboración del proyecto se ha partido del trabajo previamente realizado en [5]. De éste se han tomado las bases para desarrollar el algoritmo mínimos cuadrados en sus distintos métodos, así como la comparación de resultados para certificar su correcto funcionamiento. También fueron de gran utilidad varios artículos [2-4, 9, 10] para la toma de contacto con el tema que ocupa el proyecto y para conocer y entender distintos conceptos en él utilizados. En la Figura 1.4 se muestra un diagrama de Gantt con el tiempo de dedicación de cada etapa del proyecto.

	Nombre de tarea	*Comienzo*	*Fin*	*Duración*
1	Introducción al proyecto	09/01/2012	20/01/2012	2s
2	Análisis del algoritmo	23/01/2012	18/05/2012	17s
3	Implementación en Vivado HLS	21/05/2012	28/09/2012	19s
4	Redacción de la memoria y correciones	01/10/2012	09/11/2012	6s

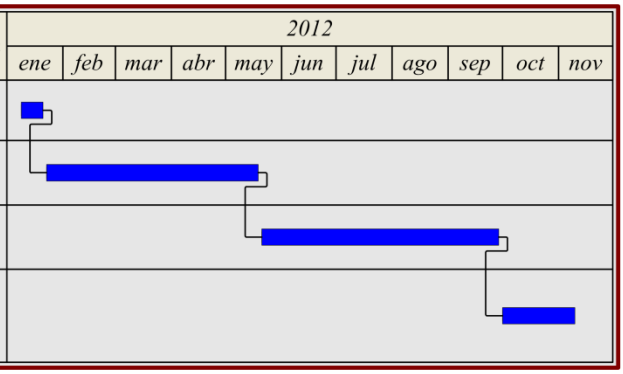


Figura 1.4. Diagrama de Gantt de las etapas del proyecto.

La fase número dos, “Análisis del algoritmo”, ha sido realizada en MATLAB y mediante ella se seleccionó tanto la forma adecuada del mismo como los parámetros a utilizar en su implementación. Las distintas sub-fases son:

- Discretización de la carga.
- Desarrollo del algoritmo en MATLAB.
- Realización de estudios sobre distintos parámetros.
- Comparación de resultados.
- Elección de parámetros para la posterior implementación.

Los estudios se han realizado siguiendo las distintas formas del algoritmo y discretización e incluyendo variaciones como el ruido que presenta el sistema real.

Para la tercera fase, "Implementación en Vivado HLS", las etapas han sido:

- Desarrollo de código auxiliar en lenguaje C.
- Desarrollo del algoritmo en lenguaje C.
- Optimización del diseño.
- Implementación con señales tomadas experimentalmente.
- Análisis de resultados en MATLAB.

1.6 Contenido

Los capítulos que siguen a continuación explican los pasos seguidos en la realización del proyecto, así como la comparación de los resultados y las conclusiones que de ella se extrae.

En este primer capítulo se han expuesto las bases de las que se ha partido al comienzo del proyecto y se han introducido brevemente algunos conceptos necesarios para la comprensión del mismo. También se han justificado los motivos de su realización y los objetivos a alcanzar.

En el segundo capítulo se tratan las etapas de las que consta la parte de análisis de los diversos modos de implementación del algoritmo. El tercero hace referencia a la implementación de la forma escogida. En ambos se exponen los resultados obtenidos y los parámetros utilizados para su obtención.

La comparativa de resultados final se muestra en el capítulo cuatro, además de su justificación y extracción de conclusiones.

Adjuntos al documento principal se incluyen varios anexos en los que se desarrollan las distintas formas del algoritmo o se presentan los resultados menos representativos que no aparecen en los capítulos anteriores.

2. ANÁLISIS DEL ALGORITMO MÍNIMOS CUADRADOS

En este capítulo se exponen las distintas etapas y resultados de simulación de las diversas implementaciones del algoritmo mínimos cuadrados analizados en el presente proyecto. La herramienta escogida para el análisis es MATLAB debido a su potencia de cálculo y facilidad a la hora de programar sistemas, así como por la familiaridad con el programa debido a su uso habitual en las carreras de ingeniería.

El esquema que se ha seguido para dicho análisis según las distintas formas del algoritmo se muestra en la TABLA 2.1. En el caso de la forma con señales de entrada V_0 e I_L, el sistema necesita únicamente de dos conversores analógico-digital. Además la digitalización de ambas señales ya está previamente disponible en el sistema debido a que son utilizadas por la etapa de control. En el caso de señales de entrada V_0, I_L y V_C, es necesario un conversor más, aumentando la complejidad del sistema. Por el contrario se espera que las prestaciones del algoritmo aumenten considerablemente. En cuanto a las formas de discretización Euler hacia adelante y trapezoidal, la primera es más sencilla respecto a las operaciones necesarias. A costa del aumento de complejidad, la precisión que se espera conseguir con la Trapezoidal es mayor. Por último las formas matricial y recursiva difieren en que en la primera se realizan en el mismo instante de tiempo las operaciones con todas las medidas tomadas. Esto hace imposible la identificación de la carga de inducción en tiempo real. La recursiva en cambio realiza las operaciones con las señales de entrada en un instante de tiempo determinado.

TABLA 2.1.
ESQUEMA DEL ANÁLISIS REALIZADO

Parámetros de entrada V_0 e I_L				Parámetros de entrada V_0, I_L y V_C			
Euler hacia delante		Trapezoidal		Euler hacia delante		Trapezoidal	
Matricial	Recursivo	Matricial	Recursivo	Matricial	Recursivo	Matricial	Recursivo

2.1 Algoritmo mínimos cuadrados

En este apartado se introduce el algoritmo mínimos cuadrados. En él se explica su funcionamiento y se exponen las ecuaciones que forman los métodos matricial y recursivo. Se analizan ambos porque aunque para la forma matricial se tiene más precisión, no es viable su implementación en FPGA. Su análisis sirve únicamente para la comparación de los resultados obtenidos con la recursiva. El desarrollo de ambos métodos se presenta en el ANEXO I.

Este algoritmo se basa en una técnica de análisis numérico que, dada una función *f* que modela el sistema, encuentra el vector de parámetros de dicha función que

minimiza el error cuadrático de las estimaciones a partir de un número *k* de puntos. Dicho error viene dado por (2-1).

$$E(f) = \frac{1}{k} \cdot \sum_{i=1}^{k} \varepsilon^2(i), \tag{2-1}$$

donde $\varepsilon(i)$ es el error de la función en el punto *i*.

El desarrollo de dichas ecuaciones a partir de la ecuación en diferencias que modela el sistema se encuentra en el ANEXO I. La forma matricial viene dada por la expresión (2-2):

$$\hat{\theta} = \left[\Phi^T(k) \cdot \Phi(k) \right]^{-1} \cdot \left(\Phi^T(k) \cdot Y \right), \tag{2-2}$$

donde $\hat{\theta}$ es el vector con la solución, *Y* es el vector de observaciones y $\Phi(k)$ es la matriz de las *k* mediciones. La ejecución de la misma supone operar con todos los términos en el mismo instante de tiempo, lo que hace imposible conocer los resultados en tiempo real.

Al desarrollarla obtenemos la forma recursiva que se implementa mediante las ecuaciones (2-3), (2-4) y (2-5):

$$L(k) = \frac{P(k-1) \cdot x(k)}{\varphi + x^T(k) \cdot P(k-1) \cdot x(k)}, \tag{2-3}$$

$$\hat{\theta} = \hat{\theta}(k-1) + L(k) \cdot \left[y(k) - x^T(k) \cdot \hat{\theta}(k-1) \right], \tag{2-4}$$

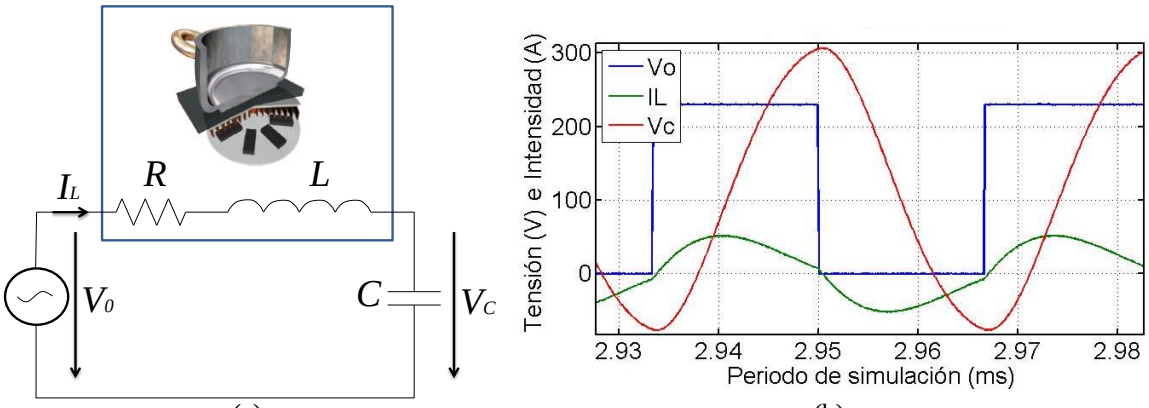
$$P(k) = \left[I - L(k) \cdot x^T(k) \right] \cdot \frac{P(k-1)}{\varphi}, \tag{2-5}$$

en las cuales $x(k)$ es el vector con los parámetros del modelo y φ es el factor de olvido utilizado para restar importancia a las muestras más antiguas. En el caso de este proyecto se eligió un valor de φ de uno para dotar de la misma importancia a todas las muestras.

Se analizaron a su vez dos formas del algoritmo según los parámetros de entrada y según la discretización fuera Euler hacia delante o Trapezoidal. El desarrollo de dichos métodos se encuentra también en el ANEXO I.

2.2 Discretización de la carga

Como se ha explicado en el capítulo 1, la carga se modela como un circuito RL al que se añade el condensador de resonancia. El circuito equivalente de la etapa de potencia y sus principales formas de onda se representan en la Figura 2.1.



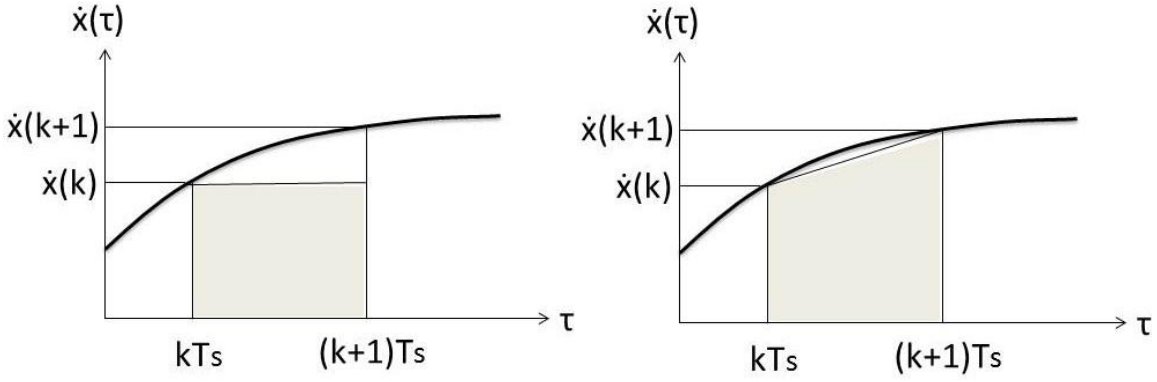
(a) (b)

Figura 2.1. (a) Sistema a simular. (b) Señales de entrada del algoritmo.

Debido a que el algoritmo ha sido previamente discretizado, es necesario discretizar también el modelo de la carga para obtener las señales de entrada. De otro modo, el error introducido es excesivamente alto y los resultados dependerían del número de muestras empleado siendo entonces muy poco funcional. Se han utilizado dos tipos de discretizaciones, trapezoidal y Euler hacia adelante, en función de con cuál se había discretizado el algoritmo, con periodo de muestreo T_S.

El proceso de discretización consiste en transformar las ecuaciones de estado en tiempo continuo que modelan el sistema en las ecuaciones en tiempo discreto tal y como se muestra en el ANEXO I.6.

En función de si la discretización es Euler hacia adelante o trapezoidal el fundamento es el mostrado en la Figura 2.2. (a) y (b) respectivamente. Como puede apreciarse en ellas el método trapezoidal es más preciso.



(a) (b)

Figura 2.2. Representación gráfica de la aproximación según el método (a) Euler hacia adelante. (b) trapezoidal.

En el primer caso la expresión que corresponde a la integral de la función es la mostrada en (2-6), mientras que la trapezoidal hace referencia a (2-7):

$$\int_{k\cdot Ts}^{(k+1)\cdot Ts} \dot{x}(\tau)d\tau = Ts\cdot\dot{x}(k), \tag{2-6}$$

$$\int_{k\cdot Ts}^{(k+1)\cdot Ts} \dot{x}(\tau)d\tau = Ts\cdot\frac{\left(\dot{x}(k)+\dot{x}(k+1)\right)}{2}, \tag{2-7}$$

donde $\dot{x}(\tau)$ es la función a discretizar, T_s el periodo de muestreo y τ el instante de tiempo.

2.3 Parámetros de simulación

A continuación se presentan los parámetros fijados para la simulación en MATLAB del sistema. En la TABLA 2.2 se reflejan los rangos de los parámetros que se han variado durante el análisis del algoritmo. Donde f_{sw} es la frecuencia de conmutación, F_S es la frecuencia de muestreo, b es el número de bits del conversor y T_C es el tiempo de cómputo cuyo significado se explica al final de esta sección.

TABLA 2.2.
RANGOS DE LOS PARÁMETROS DE SIMULACIÓN

f_{sw} (kHz)	30-75
F_S (MHz)	10 o 20
b (bits)	8, 10 y formato real
T_C (µs)	0.1-5

Para el estudio de f_{sw} se ha supuesto una carga de R = 2.85 Ω y L = 18.65 µH, condensador de resonancia de C = 1440 nF y ciclo de servicio D = 0.5. El rango de estudio ha sido de 30 a 75 kHz ya que se corresponde con el utilizado en las cocinas de inducción. Para la obtención de las señales de entrada V_0, I_L y V_C, se ha discretizado el sistema mostrado en la Figura 2.1. (a) según los dos métodos de discretización utilizados. Se ha introducido ruido blanco de potencia 10^{-14} W y se han tomado frecuencias de muestreo F_S de 10 y 20 MHz. Se ha simulado también considerando una captura de las señales de entrada mediante conversores de 8 y 10 bits de precisión para una F_S de 10 MHz.

Un parámetro importante es el tiempo que tiene el algoritmo para converger. Este tiempo está marcado por el cruce por cero de la red eléctrica, que es el momento en que es necesaria la obtención de la estimación de la impedancia. En la Figura 2.3 se observa como dicho paso por cero ocurre cada 10 ms, ya que es el periodo de tensión de bus V_{BUS}.

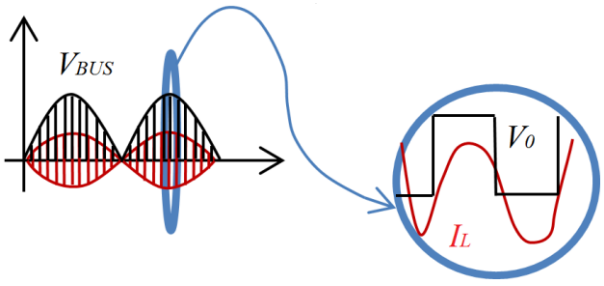


Figura 2.3. Formas de onda I_L y V_0 conformadas con V_{BUS}.

Como consecuencia de implementar el algoritmo en una FPGA es necesario realizar un estudio del tiempo de cómputo T_C, debido a que la FPGA necesita un tiempo para realizar las operaciones asociadas al algoritmo. Pasado T_C vuelven a muestrearse las señales de entrada en parejas o tríos en función de si se trata de la forma con parámetros de entrada V_0, I_L y V_C o la de V_0 e I_L respectivamente.

La Figura 2.4 muestra gráficamente el significado de T_C, para evitar confundirlo con el periodo de muestreo T_S. En ella aparece un trío de medidas representadas por cruces rojas y otro trío posterior representado por triángulos. En cada caso las medidas están separadas entre ellas un tiempo T_S, puesto hace referencia al tiempo de muestreo. Con las cruces se obtendría el vector $\hat{\theta}(k)$ que contiene la solución en el instante k. Tras un tiempo T_C se volvería a ejecutar el algoritmo con las medidas simbolizadas por los triángulos, obteniéndose el vector $\hat{\theta}(k+1)$ con las soluciones en el instante $k+1$.

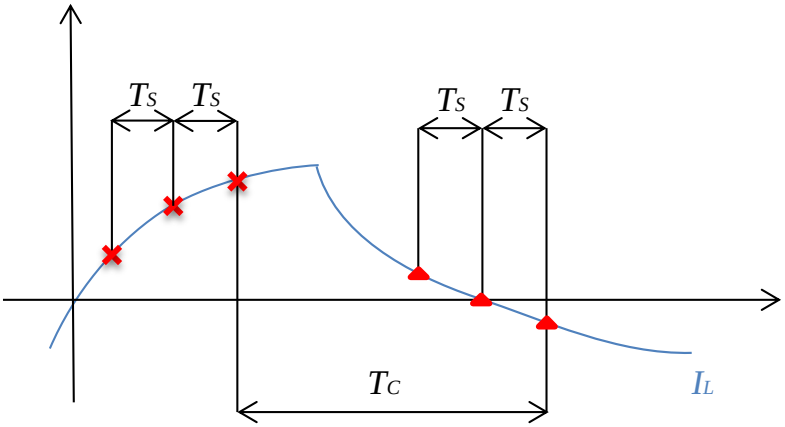


Figura 2.4. Esquema de tiempos del tiempo de cómputo del algoritmo T_C con parámetros de entrada V_0 e I_L.

El estudio de T_C se ha realizado en un rango de 0.1 a 5 µs para una f_{sw} fija de 75kHz y F_S de 10 MHz. A diferencia del estudio de f_{sw}, el realizado en función del tiempo de cómputo se efectuó únicamente sobre la forma recursiva, ya que no tiene sentido hacerlo sobre la matricial. Esto se debe a que en la matricial se utilizan todas las medidas en el mismo instante, así que el tiempo de ejecución del algoritmo no es necesario.

Por último aclarar que los errores mostrados en las gráficas han sido obtenidos mediante la ecuación (2-8):

$$Error(\%) = \frac{Parámetro\_estimado - Parámetro\_real}{Parámetro\_real} \cdot 100, \qquad (2\text{-}8)$$

donde *Parámetro_estimado* es la estimación obtenida tras la ejecución del algoritmo y *Parámetro_real* es el valor supuesto para R, L o C y con los que se obtienen las señales de entrada.

2.4 Resultados de simulación

En esta etapa se ha implementado el algoritmo según los parámetros de entrada, el primero con las medidas de la intensidad que circula por la inductancia I_L y la tensión obtenida del semipuente V_0 y el segundo añadiendo V_C. Además se han comparado las dos formas de discretizar la carga, trapezoidal y Euler hacia adelante.

Debido a que el objetivo final es la implementación en FPGA del algoritmo, el objetivo es el estudio del mismo en su forma recursiva. El único propósito de simularlo también en su forma matricial ha sido el de comparar los resultados y comprobar que el algoritmo funciona correctamente, ya que con este otro método se obtienen buenas estimaciones, aunque no es implementable en FPGA.

En los apartados presentados a continuación se exponen los resultados más significativos para las dos formas del algoritmo según los parámetros de entrada. Estos son los relativos al algoritmo con discretización trapezoidal, frecuencia de muestreo F_S de 10 MHz y señales de entrada representadas en formato 10 bits y reales. Dentro de cada apartado se muestran los errores de estimación obtenidos tanto para el estudio de T_C como para el de f_{sw}. Los resultados del análisis de T_C reflejan solo los errores de las estimaciones relativos a la forma recursiva, tal y como se explicó en el apartado 2.3.

El capítulo se cierra con las conclusiones de comparar los métodos según las señales de entrada. En el ANEXO II se presentan todos los resultados que se obtuvieron del estudio siguiendo el esquema mostrado en la TABLA 2.1.

2.4.1 Parámetros de entrada V_0 e I_L sin ruido

A continuación se exponen los resultados de las estimaciones según el algoritmo trapezoidal con señales de entrada V_0 e I_L sin previa adición de ruido blanco.

El estudio de T_C mostrado en la Figura 2.5 revela que los errores pueden considerarse nulos hasta al menos los 5 µs, que es un tiempo razonable para que la FPGA realice las operaciones. La frecuencia de conmutación utilizada para la simulación ha sido de 75 kHz puesto que es el caso más desfavorable dentro del rango de estudio.

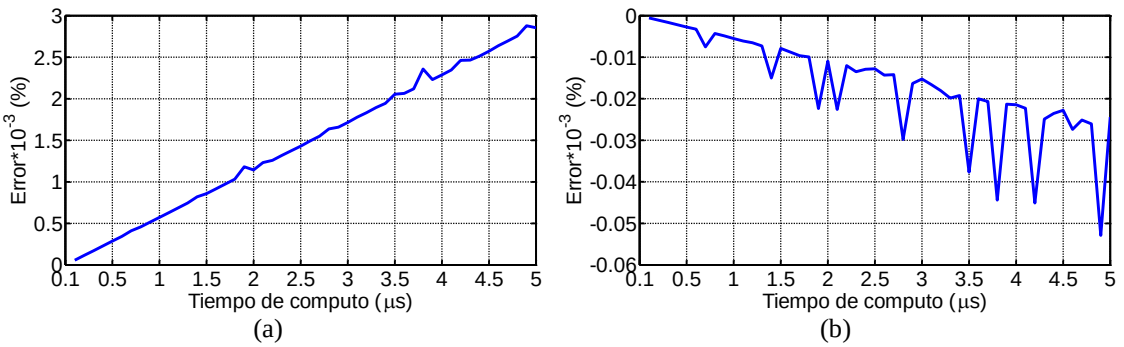


Figura 2.5. (a) Error en la estimación de la resistencia R (b) Error en la estimación de la inductancia L en función del tiempo de cómputo T_C, frecuencia de muestreo de 10 MHz y discretización trapezoidal.

En cuanto al estudio de f_{sw}, según muestra la Figura 2.6, los errores en las estimaciones de R y L pueden considerarse nulos. Dichas estimaciones han sido realizadas implementando el algoritmo con las señales de entrada en reales. Se trata por tanto de una situación ideal mediante la cual se ha comprobado que el algoritmo funciona correctamente. En relación a la forma de discretización, los resultados obtenidos con el método Euler hacia adelante son siempre peores que los del trapezoidal.

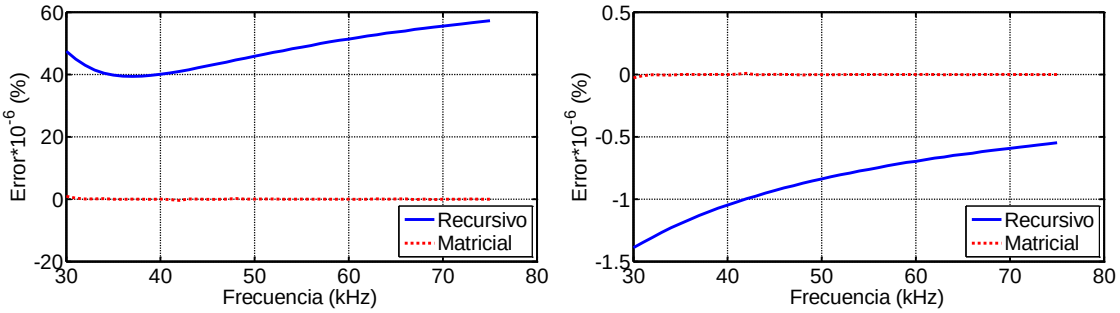


Figura 2.6. (a) Error en la estimación de la resistencia R (b) Error en la estimación de la inductancia L en función de la frecuencia de conmutación, frecuencia de muestreo 10 MHz y discretización trapezoidal.

Cuando se implementa con un número de bits menor los errores aumentan considerablemente. En la Figura 2.7 se observa como con 10 bits el error en la estimación de R alcanza los dos órdenes de magnitud. A la vista de estos resultados se descartó continuar con la implementación de esta forma del algoritmo, puesto que el error obtenido sin llegar a introducir ruido en las señales de entrada ya lo hace inviable.

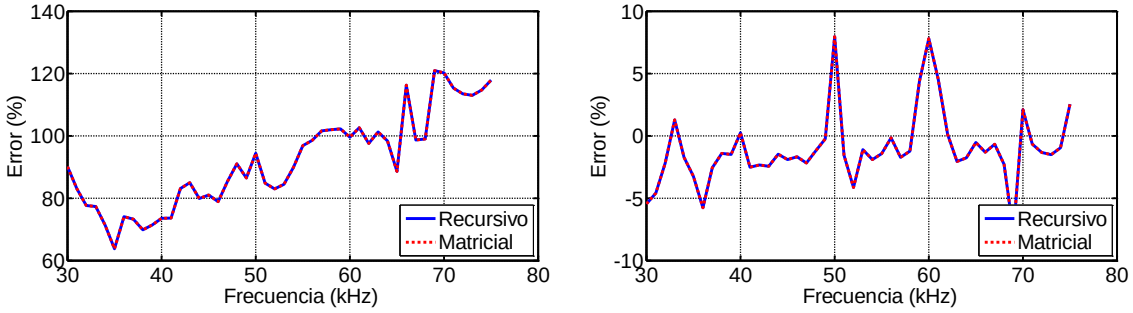


Figura 2.7. (a) Error en la estimación de la resistencia R (b) Error en la estimación de la inductancia L en función de la frecuencia de conmutación, frecuencia de muestreo de 10 MHz, señales de entrada en 10 bits y discretización trapezoidal.

2.4.2 Parámetros de entrada V_0, I_L y V_C sin ruido

En este caso se ha comenzado también con el estudio de T_C. En la Figura 2.8 se observa que se obtienen errores nulos hasta al menos los 5 µs para las estimaciones de R y L. Por tanto se ha procedido a estudiar el comportamiento del algoritmo al variar la frecuencia de muestreo y decidir si es posible continuar con el análisis.

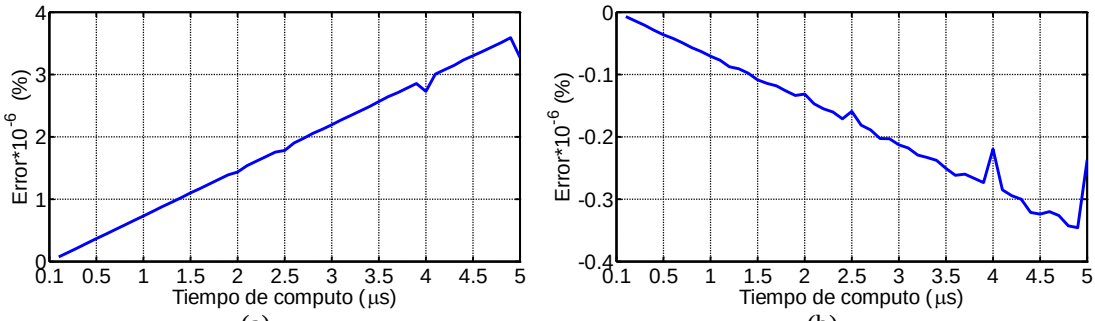


Figura 2.8. (a) Error en la estimación de la resistencia *R* (b) Error en la estimación de la inductancia *L* en función del tiempo de cómputo T_C, frecuencia de muestreo de 10 MHz y discretización trapezoidal.

La Figura 2.9 muestra errores nulos para las señales de entrada al variar f_{sw}. Aunque con la forma recursiva no se alcanza la misma precisión que con la matricial, son resultados tan bajos que pueden considerarse nulos. Dichos resultados han sido obtenidos con señales en reales, así que para acercar el sistema a la situación real el siguiente paso ha sido reducir el número de bits con el que se representan.

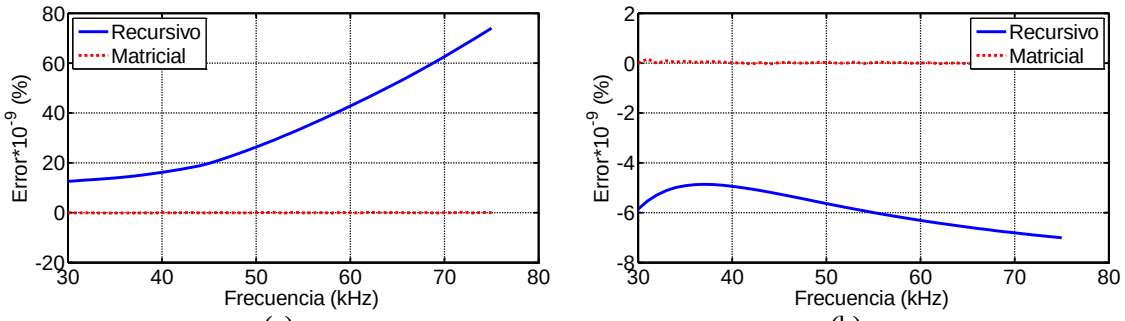


Figura 2.9. (a) Error en la estimación de la resistencia *R* (b) Error en la estimación de la inductancia *L* en función de la frecuencia de conmutación, frecuencia de muestreo 10 MHz y discretización trapezoidal.

La Figura 2.10 hace referencia a los resultados obtenidos con 10 bits, aunque se implementó también con señales de entrada representadas con 8 bits obteniéndose errores de hasta un orden mayor. Se observa también que las estimaciones conseguidas mediante la forma recursiva son prácticamente iguales que los de la matricial, por lo que a priori el algoritmo funcionaría correctamente.

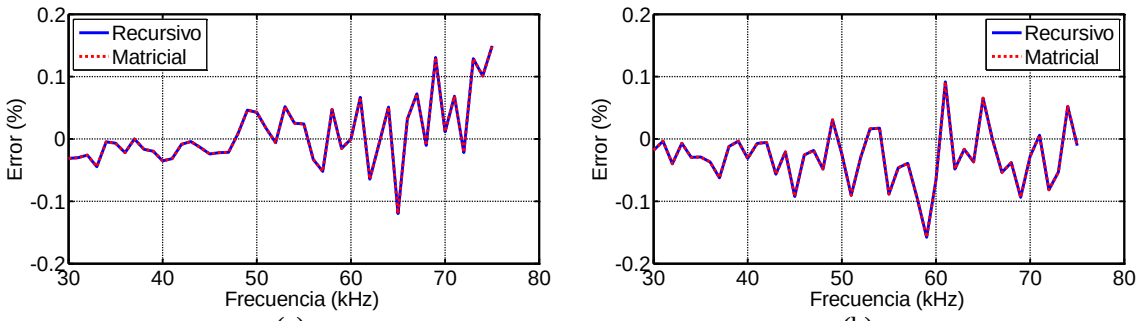


Figura 2.10. (a) Error en la estimación de la resistencia *R* (b) Error en la estimación de la inductancia *L* en función de la frecuencia de conmutación, frecuencia de muestreo de 10 MHz, señales de entrada en 10 bits y discretización trapezoidal.

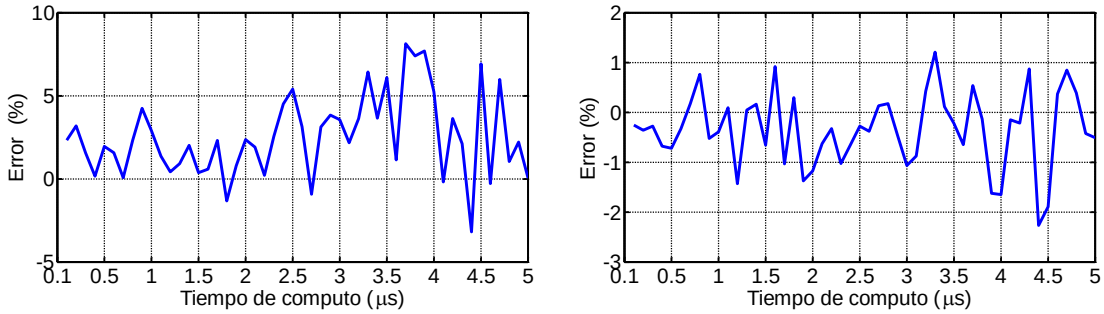
Los errores se incrementan considerablemente al disminuir el número de bits aunque, a pesar de dicho aumento, se mantienen dentro de un rango razonable, cuyos valores máximos se sitúan en torno al 0.1%. Además son menores que los obtenidos para la forma del algoritmo con señales de entrada V_0 e I_L, lo que denota una menor sensibilidad al ruido. En cuanto a la comparación entre métodos de discretización los resultados obtenidos son similares en ambos casos.

Puesto que los errores en las estimaciones alcanzan valores pequeños, el siguiente paso ha sido introducir ruido a las medidas para estudiar un sistema más cercano a la realidad.

2.4.3 Parámetros de entrada V_0, I_L y V_C con ruido

Para el análisis final del algoritmo se ha introducido ruido en las tres medidas simultáneamente, ya que es la situación que más se acerca a la realidad, siendo el resto de parámetros los mismos que en las situaciones anteriores.

Con respecto al análisis para distintos T_C se han obtenido los resultados mostrados en la Figura 2.11. En ella se observa que para 2.5 µs el error en la estimación de R alcanza como mucho un 5%. Ambos valores son razonables puesto que hay que llegar a un compromiso entre dichos parámetros, error y T_C. Para la estimación de L se obtienen errores inferiores por lo que el factor que restringe T_C es la estimación de R.



(a) (b)

Figura 2.11. (a) Error en la estimación de la resistencia R (b) Error en la estimación de la inductancia L en función del tiempo de cómputo T_C, frecuencia de muestreo de 10 MHz y discretización trapezoidal.

Para el caso del algoritmo con discretización Euler hacia adelante, cuando el tiempo de cómputo supera los 0.6 µs se incurre en errores de estimación de R mayores del 5%. La comparación gráfica de los resultados de ambos métodos se muestra en el ANEXO II.

Tras la implementación del algoritmo con señales de entrada en reales y con ruido, se obtienen errores muy superiores en comparación al utilizar señales sin ruido. Aun así en la Figura 2.12 se observa que no supera el 2.5% para la estimación de R, que es un valor razonable una vez se ha introducido el ruido. Lo mismo ocurre con la estimación de L, aunque la Figura 2.13 muestra un error de un orden menos que para la de R. Como en ambas estimaciones los errores cometidos son asumibles se ha procedido al estudio con señales con número de bits de 8 y 10.

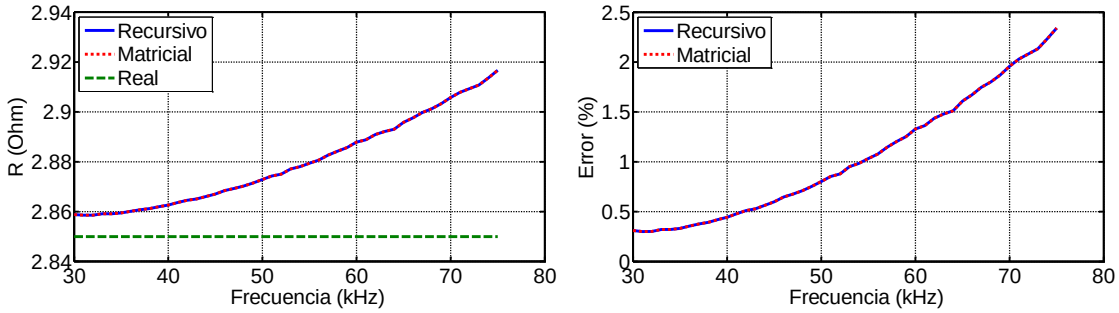


Figura 2.12. (a) Estimación de la resistencia *R* (b) Error en la estimación de la resistencia *R* en función de la frecuencia de conmutación, frecuencia de muestreo 10 MHz y discretización trapezoidal.

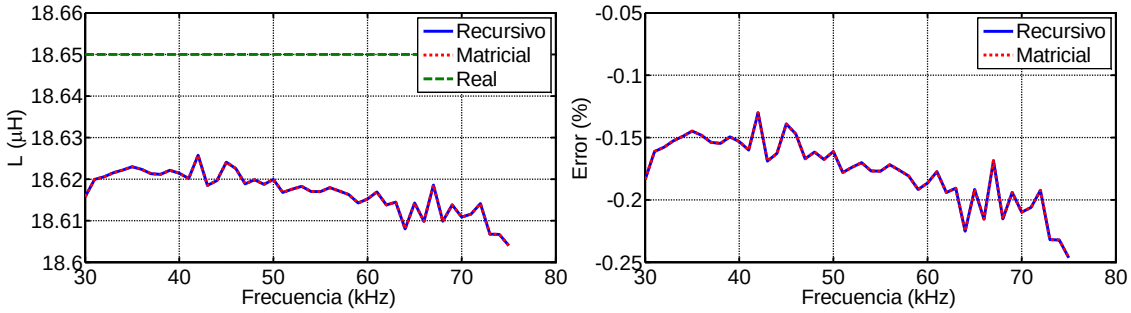


Figura 2.13. (a) Estimación de la inductancia *L* (b) Error en la estimación de la inductancia *L* en función de la frecuencia de conmutación, frecuencia de muestreo 10 MHz y discretización trapezoidal.

La Figura 2.14 refleja unos resultados con 10 bits muy similares a los que presenta el algoritmo con señales de entrada en reales. La reducción del número de bits a 8 conlleva un ligero aumento del error en las estimaciones, obteniéndose unos valores máximos de error de 4 y 0.35% para las estimaciones de *R* y *L* respectivamente.

Los errores cometidos por el algoritmo discretizado con Euler hacia adelante son similares a los del trapezoidal, tanto al usar señales en reales como en 8 y 10 bits. A pesar de ello la dispersión del error del método trapezoidal es menor, así como su tiempo de convergencia.

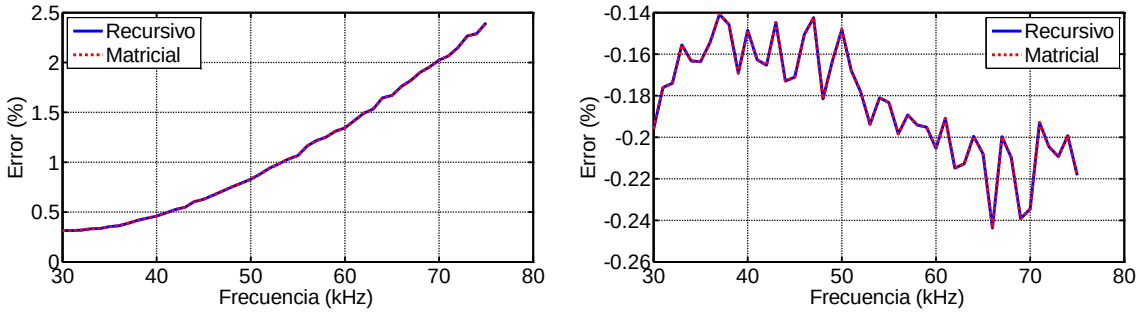


Figura 2.14. (a) Error en la estimación de la resistencia *R* (b) Error en la estimación de la inductancia *L* en función de la frecuencia de conmutación, frecuencia de muestreo de 10 MHz, señales de entrada en 10 bits y discretización trapezoidal.

2.5 Comparación de resultados

En esta sección se procede a recopilar las conclusiones del capítulo, referentes a la comparación de las dos formas del algoritmo según los parámetros de entrada. La TABLA 2.3. refleja de manera más visual los resultados del análisis en función de la frecuencia de conmutación f_{sw} en cuanto a valores máximos de error.

TABLA 2.3.
COMPARACIÓN DE RESULTADOS CON F_S DE 10 MHZ

Parámetros de entrada	**Discretización**	***b***	**Error en *R* (%)**	**Error en *L* (%)**
V_0 e I_L sin ruido	**Euler hacia adelante**	**reales**	1.5×10^{-2}	10^{-4}
		8 bits	2500	70
		10 bits	120	15
	Trapezoidal	**reales**	6×10^{-5}	1.5×10^{-6}
		8 bits	1750	45
		10 bits	120	7.5
V_0, I_L y V_C sin ruido	**Euler hacia adelante**	**reales**	7.5×10^{-8}	7.5×10^{-9}
		8 bits	2.25	1
		10 bits	0.15	0.2
	Trapezoidal	**reales**	7.5×10^{-8}	7×10^{-9}
		8 bits	2.25	0.5
		10 bits	0.15	0.15
V_0, I_L y V_C con ruido	**Euler hacia adelante**	**reales**	2.25	0.22
		8 bits	3.9	0.31
		10 bits	2.3	0.23
	Trapezoidal	**reales**	2.3	0.24
		8 bits	3.9	0.32
		10 bits	2.3	0.24

A la vista de los resultados con el método trapezoidal se obtienen menos picos de error en las estimaciones a lo largo de la zona de estudio, además de un menor tiempo de convergencia. Los valores de dicho error son similares para ambos métodos considerando un tiempo de cómputo igual al periodo de muestreo, siendo siempre la estimación de *L* más precisa que la de *R*.

Con respecto al número de bits de las señales de entrada se llegó a la conclusión de que el algoritmo con parámetros de entrada V_0 e I_L presenta una gran sensibilidad, lo que conlleva errores de hasta tres órdenes de magnitud. En base a estos resultados se descartó continuar con el análisis de esta forma.

Para la implementación con señales de entrada V_0, I_L y V_C el análisis se realizó tanto sin ruido en las medidas como con él. Las estimaciones conseguidas con señales en 10 bits sin ruido tenían errores máximos de 0.15% tanto para R como para L. Al añadir ruido a las señales de entrada en reales los errores aumentaban en la estimación de R hasta un 2.3% y en L hasta el 0.24%. Los resultados con 10 bits son similares mientras que en 8 bits los errores aumentan ligeramente.

Se observó que el algoritmo en general era muy sensible al ruido introducido en I_L, aumentando el error de las estimaciones hasta ocho órdenes de magnitud para las simulaciones con señales de entrada en reales. El mismo efecto produce representar dicha señal con un número de bits menor, aunque como puede verse en los resultados, la contribución de ambos no se suma.

Los resultados de variar la frecuencia de muestreo presentan el mismo comportamiento en todos los casos. Se obtiene un error menor para F_S de 10 MHz que para 20 MHz. Esto se debe a los errores por cancelación introducidos en gran parte del periodo de simulación, ya que uno de los parámetros de entrada es la resta de valores de V_0 en instantes consecutivos de tiempo y al aumentar el número de muestras el algoritmo no es capaz de detectar cambios.

Se estudió también el comportamiento del algoritmo en función del tiempo de cómputo T_C para el caso más desfavorable con f_{sw} de 75 kHz. Los resultados se muestran en la TABLA 2.4 teniendo en cuenta que no se ha querido superar el 5% de error máximo en la estimación de R o L.

TABLA 2.4.
RESULTADOS DEL ESTUDIO DE T_C

Parámetros de entrada	**Discretización**	**T_C (µs)**
V_0 e I_L sin ruido	**Euler hacia adelante**	> 5
	Trapezoidal	> 5
V_0, I_L y V_C sin ruido	**Euler hacia adelante**	> 5
	Trapezoidal	> 5
V_0, I_L y V_C con ruido	**Euler hacia adelante**	0.6
	Trapezoidal	2.5

A pesar de que los errores para ambos algoritmos de discretización son los mismos al considerar el tiempo de cómputo igual al periodo de muestreo, como se observa en el ANEXO II el análisis en función de T_C ha desvelado que para obtener una precisión del 5% en la estimación de la resistencia el algoritmo ha de ejecutarse en menos de 0.6 µs en el caso de Euler hacia adelante y 2.5 µs en el caso del trapezoidal. De los resultados de este estudio se ha optado por el algoritmo trapezoidal para la implementación final. Aunque para 2.5 µs los errores eran asumibles, se eligió un T_C algo menor para contar con un margen de seguridad.

La conclusión anterior hace referencia únicamente a la forma del algoritmo con entradas V_0, I_L y V_C. Para la otra forma solo se realizó el estudio con las señales de entrada sin ruido, ya que se descartó ese método debido a su alta sensibilidad.

El método que se ha elegido para implementarse en Vivado HLS ha sido el de parámetros de entrada V_0, I_L y V_C, con discretización trapezoidal, F_S de 10 MHz y señales representadas en 10 bits. Esta forma del algoritmo es la que presenta menor sensibilidad de las dos y se obtienen errores en las estimaciones aceptables. De las F_S se optó por 10 MHz por ser con la que menor error se obtenía. El número de bits elegido es 10 ya que los resultados son similares a los de utilizar las señales en reales.

Por último ha de tenerse en cuenta que el tiempo del que se dispone para ejecutar el algoritmo es de 2 µs tal y como reveló el estudio de T_C.

3. IMPLEMENTACIÓN EN VIVADO HLS

Debido a la complejidad de implementación del algoritmo, se ha utilizado una novedosa herramienta de síntesis de alto nivel. Esta permite sintetizar un diseño RTL (*Register-Transfer Level*) a partir de un código en lenguaje C. Con ella se obtienen distintas soluciones en función de si se prioriza la rapidez del sistema, el rendimiento, minimizar el área que ocupan los componentes, el tipo de componentes utilizados, etc. [11, 12]. En el ANEXO III se presenta una breve introducción al entorno, así como alguna de las características utilizadas para la realización del presente proyecto. La FPGA elegida para la implementación ha sido la Spartan-6 XC6SLX45 con el encapsulado CSG324C de Xilinx, trabajando a una frecuencia de reloj de 100 MHz.

El capítulo se divide en apartados según las distintas soluciones planteadas para la forma del algoritmo elegida. En nuestro caso la mayor restricción es la del tiempo de cómputo T_C, puesto que necesitamos que se realicen todas las operaciones en menos de 2 µs, tal y como se comprobó en el capítulo anterior. Para ello el software dispone de la posibilidad de desarrollar distintas soluciones sin variar el código, simplemente incluyendo directrices, lo que facilita enormemente la tarea. En la Figura 3.1 se muestra el diagrama de bloques de parte del sistema tras su implementación. Con él es posible visualizar la conexión entre los diversos elementos del sistema y los recursos necesarios para ejecutar las operaciones, así como los ciclos de reloj necesarios para ello.

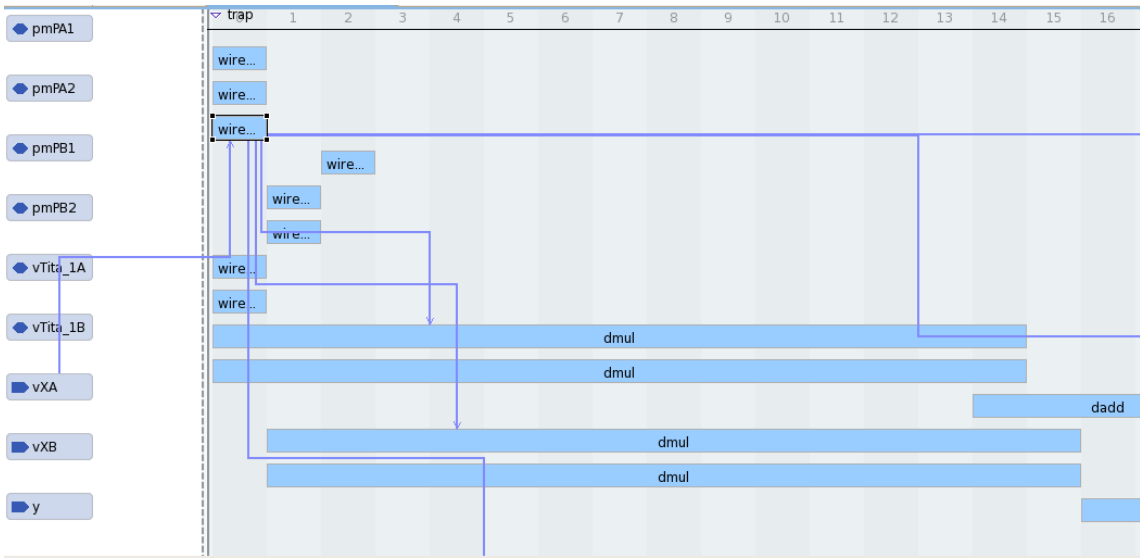


Figura 3.1. Diagrama de bloques de parte del sistema.

Al final del capítulo se exponen los resultados en cuanto a las estimaciones de la resistencia R y la inductancia L para algunas de las frecuencias de conmutación estudiadas además de las conclusiones obtenidas tras la implementación. Los resultados restantes se pueden encontrar en el ANEXO IV.

3.1 Resultados iniciales de síntesis

En esta sección se presentan los resultados de la síntesis del algoritmo sin aplicar ninguna directriz. Como se observa en la TABLA 3.1, se cumple con la restricción del tiempo de cómputo, pero el porcentaje de la utilización de DSP48Es (D*igital Signal Processing logic Element*) es superior al 100% por lo que será necesario optimizar la implementación digital realizada. La potencia no presenta unidades puesto que el valor no es una estimación real, sino que sirve únicamente de comparación entre distintas soluciones.

TABLA 3.1.
RESULTADOS

Mínimo periodo de reloj (ns)	8.63				
Ciclos de reloj utilizados	116				
T_C (µs)	1				
Potencia	3220				
Utilización de componentes (%)	BRAM	DSP48E	FF	LUT	SLICE
	0	234	28	60	0

3.2 Optimización del área ocupada

Para cumplir con los requisitos de recursos de la FPGA utilizada se ha intentado reducir el número de componentes usados para ejecutar el algoritmo. La base de dicha optimización es la reutilización de hardware en la implementación de los distintos bucles y en la utilización de funciones cuando se repiten conjuntos de operaciones.

Para ello se ha empleado la directiva PIPELINE, cuyo comportamiento es el representado en la Figura 3.2. Por defecto la directiva busca un intervalo de inicio *II* igual a uno. Esto significa que intentará procesar una nueva señal de entrada cada ciclo de reloj o el mínimo posible que más se le acerque y que permita a su vez realizar las operaciones.

Por defecto Vivado HLS intenta ejecuta las operaciones en paralelo reduciendo la latencia del diseño. Con la directiva se puede mejorar el rendimiento permitiendo diferentes formas de ejecución de la función o iteraciones del bucle, solapándolas en el tiempo.

En la Figura 3.2. (a) y (b) obtenida de la documentación del software [11] se muestran respectivamente las latencias de un código con tres funciones y otro de un bucle con tres operaciones en cada iteración. A priori los ciclos de reloj necesarios son 8 y 6 respectivamente. Al incluir la directiva se solapan las funciones A, B y C reduciéndose el número de ciclos a 5. En el caso del bucle, cuando termina la primera operación de la primera iteración comienza la operación siguiente, simultáneamente a la primera operación de la segunda iteración, reduciéndose la latencia a 4 ciclos.

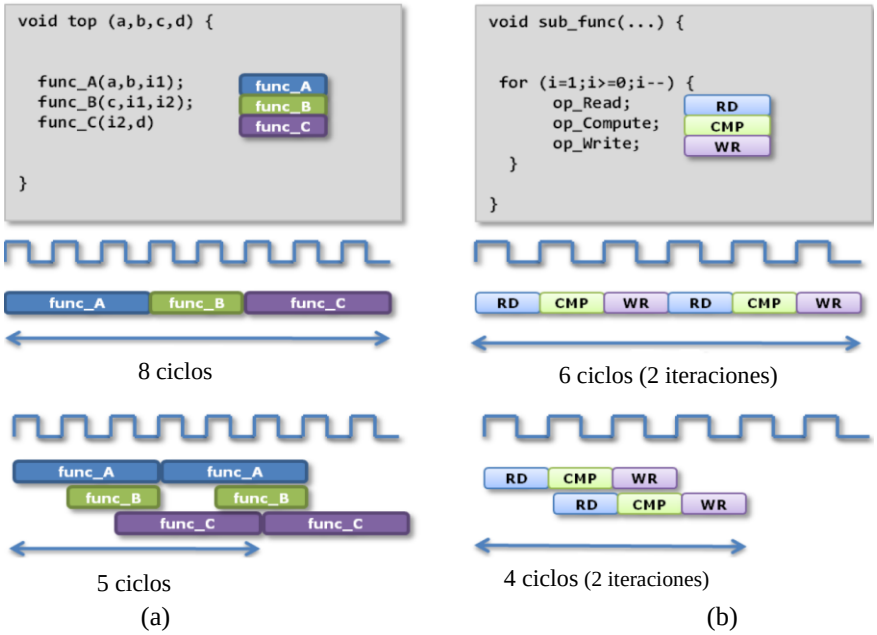


Figura 3.2. Comportamiento de la directiva PIPELINE a nivel de (a) función (b) bucle.

Aunque es posible modificar el parámetro *II*, para un primer análisis no se ha cambiado el valor por defecto. En la TABLA 3.2 pueden verse los resultados conseguidos para esta optimización. La disminución del área ocupada por debajo del 100% se consigue a expensas del tiempo de ejecución del algoritmo, que aumenta hasta 1.72 µs, y de un aumento del reloj mínimo a 15.92 ns. Como era de esperar ante una reducción tan amplia de los recursos utilizados, la potencia ha disminuido a menos de la mitad. El *II* mínimo que se ha conseguido es de 107 ciclos.

TABLA 3.2.
RESULTADOS

Mínimo periodo de reloj (ns)	15.92				
Ciclos de reloj utilizados	108				
II	107				
T_C (µs)	1.72				
Potencia	1509				
Utilización de componentes (%)	BRAM	DSP48E	FF	LUT	SLICE
	0	58	13	28	0

El periodo de reloj obtenido es superior al periodo de la FPGA (10 ns). Por ello esta implementación es desechada. Con el objetivo de obtener una implementación válida, se ha realizado un análisis variando el *II* mediante la imposición de directivas. Puesto que la prioridad del software es ejecutar las operaciones en el menor número de ciclos posible, al aumentarlo se permite al software analizar más combinaciones de periodos de reloj y utilización de componentes.

En la TABLA 3.3 se presentan los resultados obtenidos con la mejor solución que se ha alcanzado, la cual cumple los requerimientos del sistema. El resto de resultados para los distintos *II* se muestran en el ANEXO IV.

TABLA 3.3.
RESULTADOS

Mínimo periodo de reloj (ns)	8.63				
Ciclos de reloj utilizados	121				
II	118				
T_C (µs)	1.04				
Potencia	1289				
Utilización de componentes (%)	BRAM	DSP48E	FF	LUT	SLICE
	0	29	11	23	0

Con la optimización del área ocupada se ha conseguido finalmente reducir los componentes utilizados a unos porcentajes mínimos sin apenas aumento del tiempo de cómputo. Esto conlleva, además de la viabilidad de implementación de la solución en la FPGA seleccionada, la disminución de la potencia consumida cumpliendo a su vez ampliamente con la restricción de T_C.

3.3 Resultados de las estimaciones

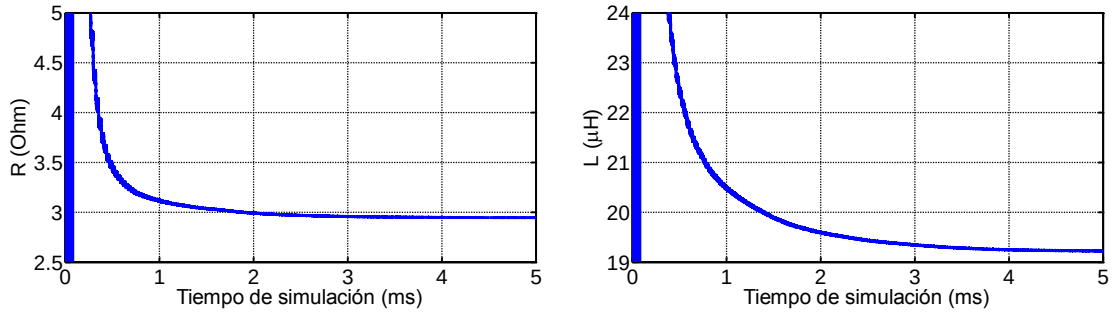
En este apartado se presentan los resultados de las estimaciones de R y L en función del tiempo de simulación T_{STOP}, alcanzados mediante el código sintetizado por el software Vivado HLS. Dichos resultados se han obtenido gráficamente en MATLAB.

Los parámetros utilizados han sido los elegidos en las conclusiones del capítulo anterior. Las señales I_L, V_0 y V_C fueron obtenidas experimentalmente mediante un banco de pruebas del laboratorio con una F_S de 10 MHz, un ciclo de servicio D = 0.5 y un número de bits igual a 10.

Para minimizar la variación de impedancia que presenta un sistema real inductor-recipiente, en este proyecto se ha utilizado una carga patrón R-L formada por componentes discretos. La impedancia de la carga patrón ha sido medida en pequeña señal mediante un analizador de impedancias Agilent 4294A, obteniendo unos resultados de impedancia de R = 2.85 Ω y L = 18.65 µH. El valor del condensador de resonancia es también el mismo: C = 1440 nF.

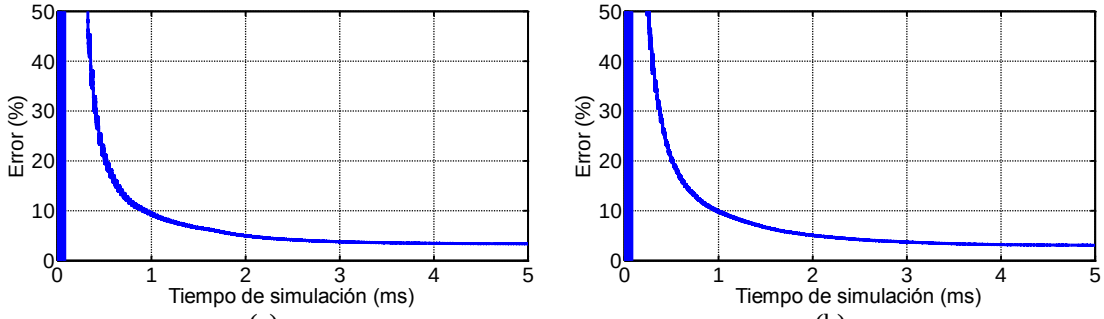
Solo aparecen los resultados relativos a las f_{sw} de 35 y 75 kHz, puesto que son los límites de la zona de estudio y las más representativas. El resto de gráficas se encuentran en el ANEXO IV.

En la Figura 3.3 y Figura 3.4 se aprecia la respuesta del algoritmo para una f_{sw} de 35 kHz, que con un tiempo de respuesta de poco más de 1 ms, consigue unas estimaciones de R y L con errores de entorno al 3%.



(a) (b)

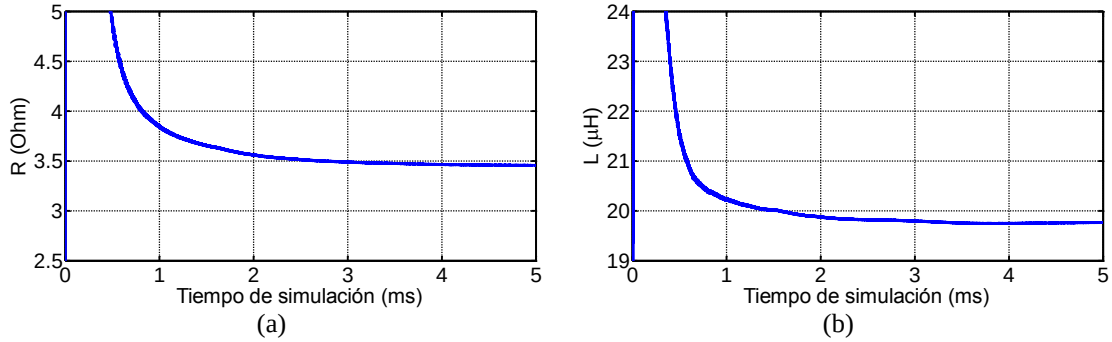
Figura 3.3. (a) Estimación de la resistencia R (b) Estimación de la inductancia L en función del tiempo de simulación T_{STOP} para frecuencia de conmutación f_{sw} de 35 kHz.



(a) (b)

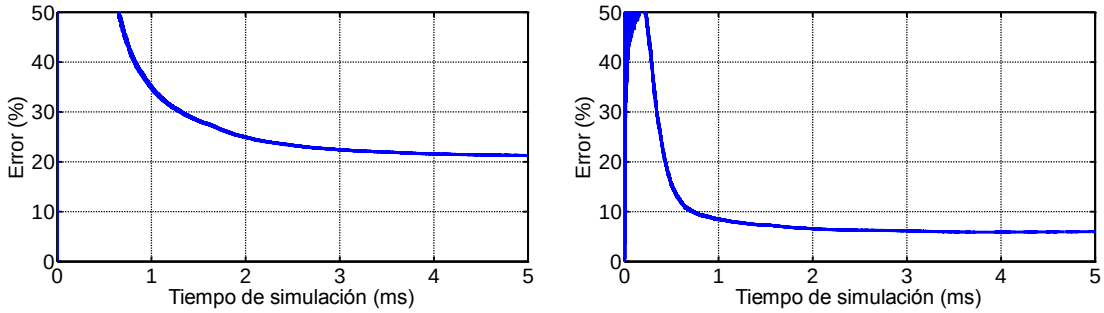
Figura 3.4. (a) Error en la estimación de la resistencia R (b) Error en la estimación de la inductancia L en función del tiempo de simulación T_{STOP} para frecuencia de conmutación f_{sw} de 35 kHz

Conforme aumenta la f_{sw} lo hacen tanto el tiempo de respuesta como los errores de estimación. En el caso de 75 kHz, mostrado en la Figura 3.5 y en la Figura 3.6, el tiempo de respuesta es de unos 2 ms. La repercusión del cambio de f_{sw} es mucho mayor en la estimación de R que en la de L, alcanzando errores del 21% y del 7% respectivamente y siendo los máximos respecto del resto de frecuencias de conmutación.



(a) (b)

Figura 3.5. (a) Estimación de la resistencia R (b) Estimación de la inductancia L en función del tiempo de simulación T_{STOP} para frecuencia de conmutación f_{sw} de 75 kHz.



(a) (b)

Figura 3.6. (a) Error en la estimación de la resistencia *R* (b) Error en la estimación de la inductancia *L* en función del tiempo de simulación T_{STOP} para frecuencia de conmutación f_{sw} de 75 kHz.

3.4 Conclusiones tras la implementación

Aunque inicialmente se superó con creces la ocupación máxima, se consiguió encontrar una solución que cumpliese con las restricciones de tiempo y componentes disponibles. Esto ha permitido, no solamente implementar el algoritmo propuesto, sino valorar la utilidad de Vivado HLS como herramienta de síntesis, optimizando el diseño de forma cómoda y rápida.

En cuanto a los resultados relativos a la FPGA, en el mejor caso para el que se cumplía con el área ocupada y el mínimo periodo de reloj, el tiempo necesario para ejecutar el algoritmo ha sido de 1.04 µs, por lo que no se introducen grandes errores producidos a causa de un tiempo de cómputo elevado.

Tras la implementación del algoritmo en Vivado HLS se han obtenido estimaciones con errores máximos en *R* del 21% y en *L* del 7% para una f_{sw} de 75 kHz, que es el caso más desfavorable. El tiempo de respuesta máximo se sitúa en torno a los 2 ms también para dicha f_{sw}. Tal y como se comprobó previamente en el capítulo anterior, los valores de error son menores al disminuir la frecuencia de conmutación.

4. CONCLUSIONES

El objetivo del presente proyecto es el de analizar un sistema de identificación de cargas de inducción en tiempo real basado en el algoritmo de mínimos cuadrados para implementar en una FPGA.

Para ello se realizó un estudio de las distintas formas del algoritmo en MATLAB, introduciendo y variando parámetros como el ruido en las señales de medida, el número de bits usado en la representación de las mismas, la frecuencia de muestreo F_S o el tiempo de cómputo T_C.

Los resultados del estudio de la frecuencia de conmutación f_{sw} se reflejan en la TABLA 2.3, en forma de los valores máximos de error obtenidos. A la vista de estos se descartó la forma con entradas V_0 e I_L por su elevada sensibilidad al ruido introducido al representar dichas señales con menos bits. Aunque los valores de error máximos resultantes con la forma Euler hacia adelante y trapezoidal son similares cuando el tiempo de cómputo T_C es igual al periodo de muestreo, con la trapezoidal el tiempo de convergencia y el nivel de dispersión del error son menores.

Con el estudio de T_C se obtuvieron los resultados presentados en la TABLA 2.4. A partir de ellos se descartó para la implementación la forma Euler hacia adelante, ya que el tiempo de ejecución del algoritmo se limitaba a 0.6 µs, mientras que con la trapezoidal aumentaba hasta los 2.5 µs.

Los resultados presentados en los capítulos del presente proyecto hacen referencia únicamente a los obtenidos con frecuencia de muestreo de 10 MHz, ya que con 20 MHz los errores son mayores en todos los casos.

Se escogió la forma recursiva trapezoidal con entradas V_0, I_L y V_C en 10 bits y frecuencia de muestreo de 10 MHz para analizar su implementación en una FPGA.

Dicho análisis de implementación se llevó a cabo mediante la herramienta Vivado HLS utilizando señales tomadas experimentalmente en un banco de pruebas. Los resultados referentes a la mejor solución encontrada se presentan en la TABLA 3.3. Estos mostraron que, para la FPGA disponible, se conseguía una implementación viable sin necesidad de un elevado tiempo de cómputo, evitando así el aumento de los errores de estimación por esa causa. Para alcanzar estos resultados se aprovechó la facilidad que proporciona el software a la hora de introducir directrices sin necesidad de cambiar el código que modela el algoritmo.

Se representaron finalmente en MATLAB las estimaciones obtenidas alcanzando unos errores máximos en las mismas del 21% y el 7% para R y L respectivamente. Se observa por tanto un elevado aumento con respecto a los resultados del análisis previo del algoritmo. Esto es debido a las distintas contribuciones de

factores como las tolerancias de los componentes, ruido no previsto en las medidas o la variabilidad de la carga patrón a lo largo de la zona de estudio.

La conclusión final es que la implementación de la forma óptima del algoritmo mínimos cuadrados es viable mediante los recursos disponibles, aunque se incurre en errores importantes a altas frecuencias de conmutación. Como trabajo futuro se propone estudiar la utilización de filtros para reducir los errores producidos por el ruido que portan las señales medidas, ya que en general el algoritmo presenta una fuerte sensibilidad al ruido en la señal de entrada I_L.

BIBLIOGRAFÍA

[1] J. Acero, J. M. Burdio, L. A. Barragan, D. Navarro, R. Alonso, J. Ramon, F. Monterde, P. Hernandez, S. Llorente, and I. Garde, "Domestic Induction Appliances," *Industry Applications Magazine, IEEE*, vol. 16, pp. 39-47, 2010.

[2] O. Jimenez, L. A. Barragan, D. Navarro, J. I. Artigas, I. Urriza, and O. Lucia, "FPGA-based harmonic computation through 1-bit data stream signals from delta-sigma modulators applied to induction heating appliances," in *Applied Power Electronics Conference and Exposition (APEC), 2011 Twenty-Sixth Annual IEEE*, 2011, pp. 1776-1781.

[3] O. Jimenez, L. A. Barragan, D. Navarro, O. Lucia, J. I. Artigas, and I. Urriza, "FPGA-based real-time calculation of the harmonic impedance of series resonant inductive loads," in *36th Annual Conference on IEEE Industrial Electronics Society IECON 2010*, 2010, pp. 1715-1720.

[4] O. Jimenez, L. A. Barragan, I. Urriza, O. Lucia, D. Navarro, and J. I. Artigas, "FPGA-based real-time harmonic impedance measurement of series resonant loads by using lock-in algorithm," in *37th Annual Conference on IEEE Industrial Electronics Society IECON 2011*, 2011, pp. 2808-2813.

[5] D. Palacios, "Identificación y control electrónico aplicado al calentamiento por inducción," Departamento de Ingeniería Electrónica y Comunicaciones, Universidad de Zaragoza, 2005.

[6] J. Acero, "Estudio teórico y experimental del calentamiento por inducción domestico de cualquier material conductor," Departamento de Ingeniería Electrónica y Comunicaciones, Universidad de Zaragoza, 2005.

[7] Available: http://www.xilinx.com/fpga/asic.htm

[8] Xilinx. (2011). *"Spartan-6 FPGA Datasheet"*. Available: http://www.xilinx.com

[9] O. Lucia, L. A. Barragan, J. M. Burdio, O. Jimenez, D. Navarro, and I. Urriza, "A Versatile Power Electronics Test-Bench Architecture Applied to Domestic Induction Heating," *IEEE Transactions on Industrial Electronics*, vol. 58, pp. 998-1007, 2011.

[10] O. Lucia, J. M. Burdio, I. Millan, J. Acero, and D. Puyal, "Load-Adaptive Control Algorithm of Half-Bridge Series Resonant Inverter for Domestic Induction Heating," *IEEE Transactions on Industrial Electronics*, vol. 56, pp. 3106-3116, 2009.

[11] Xilinx. (2012). *"Vivado Design Suite User Guide"*. Available: http://www.xilinx.com/products/design-tools/vivado/integration/esl-design/hls/index.htm

[12] Xilinx. (2012). *"Vivado Design Suite Tutorial"*. Available: http://www.xilinx.com/products/design-tools/vivado/integration/esl-design/hls/index.htm