UNIVERSIDAD AUTONOMA DE MADRID

ESCUELA POLITECNICA SUPERIOR

Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación

TRABAJO FIN DE GRADO

Implementación de un modelo HIL en coma fija parametrizable con pérdidas eléctricas

Víctor Pinazo García
Tutor: Alberto Sánchez González
Ponente: Ángel de Castro Martín

Mayo 2017

Implementación de un modelo HIL en coma fija parametrizable con pérdidas eléctricas

AUTOR: Víctor Pinazo García
TUTOR: Alberto Sánchez González

Trabajo realizado en el grupo

HCTLab

Human Computer Tecnology Laboratory

Escuela Politécnica Superior

Universidad Autónoma de Madrid

Mayo 2017

UNIVERSIDAD AUTONOMA DE MADRID

Escuela Politécnica Superior

Resumen

Es indudable que el uso de reguladores digitales para controlar convertidores de potencia es cada vez más frecuente. Esto es debido a los grandes beneficios que aporta el control digital frente al analógico, a la vez que el precio está disminuyendo paulatinamente.

Antes de pasar a la etapa de producción, un controlador digital para un convertidor de potencia necesita, al igual que cualquier otro circuito electrónico, superar una serie de rigurosos exámenes para cerciorarse del correcto funcionamiento del mismo. Si no se llegara a detectar un fallo en el conjunto, se podrían causar daños materiales o incluso daños personales debido a la gran cantidad de energía con la que se trabaja. Dichos riesgos provocan que la verificación experimental no sea una tarea sencilla y por tanto nos vemos forzados a realizar distintas pruebas de simulación antes del proceso de producción.

Sin embargo, la simulación de reguladores digitales para una planta analógica no es trivial. Aunque existen simuladores mixtos, estos no permiten realizar simulaciones a tiempo real cuando la frecuencia de conmutación del regulador es muy elevada (cientos de kilohercios o megahercios). Esto es debido a que no se consigue tiempo real cuando el paso de integración es menor que un microsegundo.

Con el fin de acelerar las simulaciones se emplea la técnica HIL (*Hardware in the Loop*). Esta técnica consiste en la digitalización de la planta mediante un modelo matemático y su ejecución en hardware para mejorar el rendimiento de la simulación. De esta forma, se consiguen emulaciones en tiempo real con tiempos de integración de hasta decenas de nanosegundos.

En este proyecto se ha aplicado la técnica HIL, usando FPGAs (acrónimo), a la digitalización de un convertidor conmutado de tipo elevador. Además, se ha llevado a cabo un estudio sobre las diferentes aritméticas utilizables en FPGA. También se muestra el uso de coma fija parametrizable que consigue aunar la rapidez y los recursos característicos de la coma fija, y la adaptabilidad de la coma flotante. Por tanto, obteniendo grandes rendimientos, el sistema se puede adaptar a las condiciones de simulación requeridas por el regulador a probar.

En este proyecto se muestra una comparativa entre diferentes aritméticas escritas en VHDL, teniendo como referencia un modelo diseñado en Matlab Simulink. Además, se muestra la influencia de las pérdidas eléctricas en el modelo, y se presentan modelos VHDL también incluyendo dichos efectos. Los resultados experimentales demuestran que utilizando coma fija parametrizable se consigue modelar el convertidor conmutado elevador con pérdidas eléctricas en tiempo real con tiempos de integración de hasta 31 ns, mientras que los resultados de precisión muestran que el error cometido frente a Simulink es menor del 1%.

Abstract

Nowadays, digital controllers are an increasing tool that helps controlling power converters due to all their benefits, which also include price decreasing.

The way digital controllers are prepared for manufacturing is similar to other electronic circuits – they need to pass a series of tests in order to determine whether their performance is accurate or not. If an error is not found during this process, it could result in both material and personal severe damages, due to the great amount of electricity that converters conduct. These risks make experimental verification a dangerous task and several tests must be carried out before starting the manufacturing.

However, simulations of digital controllers for an analog plant are complex. Even though mixed simulators exist, they are not able to perform real time simulations when the switching frequency is high (hundreds of kilohertzs or megahertzs), as they cannot archive real time when integration is lower than a microsecond.

In order to speed up simulations, a technique called HIL (Hardware in the Loop) may be applied. This technique is based on the digitalization of a plant through a mathematical method and its execution in hardware for improving the simulation performance. Thanks to this method, it is possible to get real time emulations with integration times up to dozens of nanoseconds.

In this Project, the HIL technique was applied by using FPGAs (field-programmable gate array) during the digitalization of a boost converter. What is more, a study regarding the different arithmetics that could apply to FPGA was carried out. It was also proved that the usage of the parametric fixed point can coordinate the speed and resources of the fix point with the adaptability of the floating point. Thus, the system can be adapted to the conditions of the simulation required by the controller that will be tested if the performance is accurate.

This Project also includes a comparative between different VHDL arithmetics, keeping in mind the model that has been designed in Matlab Simulink. The influence of the electric losses in this model was shown including some VHDL methods where it is possible to find these effects. The experimental results also prove the following facts. On the one hand, it is possible to shape a real-time boost converter with electric losses and integration timing up to 31 ns by using the parametric fixed point. On the other hand, accuracy results show that the error made is lower than 1% if compared to Simulink.

PALABRAS CLAVE

Convertidor elevador, fuente conmutada, emulación, simulación, *hardware-in-the-loop*, coma fija parametrizable.

KEYWORDS

Boost converter, switching source, emulation, simulation, hardware-in-the-loop, parametric fixed point.

A mi Madre.

Agradecimientos

Primero, quiero agradecer todo el apoyo que siempre me ha dado mi familia. En especial a mi padre, mi hermano, mi hermana y mi abuela, que son los que me han soportado después de cada jornada de trabajo. También, quiero dar las gracias a mi pareja, que me ha ayudado en los momentos más difíciles: gracias por tus consejos y tu compresión.

Gracias a mis compañeros de clase, que me han aportado risas y buenos momentos y han hecho mucho más ameno el trabajo del día a día. Gracias por tantos momentos únicos y estar siempre a mi lado.

Por último, como no, dar las gracias a mi tutor, Alberto Sánchez, por su apoyo y cercanía durante todo el proyecto. Por sacar tiempo en cualquier momento para mí, sobretodo en la etapa final. Y para terminar también quería dar las gracias a Ángel de Castro, quien me brindó la oportunidad de trabajar en este proyecto y apostó por mí.

Gracias a todos.

Víctor Pinazo García

Mayo 2017

INDICE DE CONTENIDOS

1 Introducción 1
2 Estado del arte 5
3 Modelado de un convertidor conmutado de potencia 7
4 Alternativas aritméticas durante la implementación 11
4.1 Modelado real del conmutador elevador 11
4.2 Modelado en coma fija del conmutador elevador 12
4.3 Modelado en coma fija parametrizable del conmutador elevador 17
5 Incorporación de pérdidas eléctricas 21
5.1 Modelo de un convertidor boost con pérdidas 21
5.2 Modelado real del conmutador elevador con pérdidas 25
5.3 Modelado con coma fija parametrizable del conmutador elevador con pérdidas 27
6 Resultados experimentales 29
6.1 Comparaciones 31
6.2 Área y frecuencia 35
7 Conclusiones y trabajo futuro 37
7.1 Conclusiones 37
7.2 Trabajo Futuro 38
Referencias 39
Glosario 41
Anexos I
A Anexo de resultados obtenidos en simulación I
A.1 Comparación del modelo Simulink sin pérdidas y ModelSim real sin pérdidas I
A.2 Comparación del modelo Simulink con pérdidas y ModelSim real sin pérdidas III
A.3 Comparación del modelo Simulink con pérdidas y ModelSim real con pérdidas V
A.4 Comparación del modelo Simulink y ModelSim parametrizable con pérdidas VI

INDICE DE FIGURAS

Figura 1. Sistema mixto de un convertidor de potencia controlado digitalmente. 2

Figura 2. Topología eléctrica de un convertidor elevador. 7

Figura 3. Topología eléctrica de un *boost* con interruptor cerrado. 9

Figura 4. Topología eléctrica de un *boost* con interruptor abierto y diodo en CCM. 9

Figura 5. Topología eléctrica de un *boost* con interruptor abierto y diodo en DCM. 10

Figura 6. Esquemático del *boost* implementado en *real.* 12

Figura 7. Formato de una señal en notación QX.Y. 13

Figura 8. Esquemático del *boost* implementado en coma fija. 16

Figura 9. Esquemático del *boost* implementado en coma fija parametrizable. 19

Figura 10. Diseño de las pérdidas causadas por la bobina. 21

Figura 11. Diseño de las pérdidas causadas por un condensador. 21

Figura 12. Diseño de las pérdidas causadas por el diodo. 22

Figura 13. Diseño de las pérdidas causadas por el interruptor. 22

Figura 14. Topología de un convertidor elevador con pérdidas. 22

Figura 15. Topología eléctrica de un *boost* con pérdidas e interruptor cerrado. 23

Figura 16. Topología de un *boost* con pérdidas interruptor abierto y diodo CCM. 24

Figura 17. Topología de un *boost* con pérdidas, interruptor abierto y diodo DCM. 24

Figura 18. Esquemático de un *boost* con pérdidas implementado en *real*. 26

Figura 19.Esquemático de un *boost* con pérdidas implementado en coma fija parametrizable. 28

Figura 20. Esquemático de un convertidor *boost* en *Simulink.* 29

Figura 21. Simulación v_C sin pérdidas en *Simulink* y en formato *real.* 32

Figura 22. Simulación $v_C$con pérdidas en *Simulink* y en formato *real* sin pérdidas. 33

Figura 23. Simulación con pérdidas en *Simulink* y en formato *real* con pérdidas. 34

Figura 24. Simulación con pérdidas en *Simulink* y en formato parametrizable con pérdidas eléctricas. 35

INDICE DE TABLAS

Tabla 1. Soluciones de i_C y de v_L en función del estado del modelo. 10

Tabla 2. Formato de las señales en QX.Y. 14

Tabla 3. Tamaño de las señales según la operativa. 15

Tabla 4. Soluciones de i_C y de v_L en función del estado del modelo con pérdidas. 25

Tabla 5. Valores empleados en el diseño del circuito sin pérdidas. 30

Tabla 6. Valores empleados en el diseño del circuito con pérdidas. 31

Tabla 7. Recursos ocupados por la FPGA. 36

1 Introducción

En las últimas décadas, la electrónica de potencia ha contribuido de diferentes formas al desarrollo de nuevos modelos para el procesamiento de energía. Es posible definir la electrónica de potencia como aquella parte de la electrónica que permite el manejo y la transformación de la energía, permitiendo controlar y transformar niveles de voltajes y corrientes. Se trata de una de las ramas más importante de la electrónica, debido a la gran variedad de aplicaciones que tiene. Esta área resulta tan influyente debido al gran número de dispositivos que requieren de unas condiciones de alimentación determinadas.

Antiguamente, los reguladores lineales eran los encargados de realizar el control de electricidad. Sin embargo, el uso de disipadores en su interior los ha caracterizado por su baja eficiencia. Hoy en día, existen nuevas formas de transformar la energía consiguiendo resultados similares y perdiendo menos energía. Uno de los dispositivos capaces de mejorar esta eficiencia son los convertidores conmutados, formados por elementos no disipativos, por lo que sufren menor cantidad de pérdidas resistivas. Dentro de estos componentes, los más característicos serán los condensadores y bobinas que permitirán almacenar y descargar energía. También gozan de gran importancia los conmutadores activos o pasivos como los MOSFETs, IGBTs, diodos, etc., cuyo fin es dejar pasar corriente sólo cuando es necesario, aumentando la eficiencia del convertidor. Este conjunto debe ser combinado con un sistema de control que genere las señales que manipulen el estado de los interruptores.

En la actualidad, existen numerosos tipos de convertidores. Entre los convertidores más sencillos destacan tres tipos. El primer grupo es conocido como convertidores reductores (en inglés, *buck converter*) que disminuyen la tensión de salida. El siguiente grupo es el de los convertidores elevadores (en inglés, *boost converter*), que aumentan la tensión de salida y, por último, los convertidores reductores-elevadores que se encargan de realizar ambas funciones.

El control de convertidores conmutados puede realizase mediante el uso de controladores digitales o analógicos. El uso de los controladores digitales ha aumentado paulatinamente en el ámbito comercial llegando a convertirse en una realidad en los convertidores de potencia. Son indudables las ventajas que aportan estos convertidores, aportando nuevas funcionalidades al convertidor, haciéndolo más robusto ante el envejecimiento de los componentes, etc. [1]. Su principal desventaja es el mayor coste respecto al control analógico, pero, durante las últimas décadas, dicho coste se ha ido reduciendo de forma drástica.

Sin embargo, la simulación de reguladores digitales para una planta analógica no es trivial. Como se puede observar en la Figura 1, estos simuladores necesitan conversores de señal (ADC, Analog to Digital Converter). Aunque existen simuladores mixtos, estos no permiten realizar simulaciones a tiempo real cuando la frecuencia de conmutación del regulador es muy elevada. Además, no se encuentran en el mercado numerosos simuladores mixtos, y su entendimiento de hardware o código digital es limitado. Debido al escaso rendimiento, es habitual encontrar simulaciones mixtas complejas que se ejecutan cientos y miles de veces más lento que el tiempo real de simulación. Para simulaciones largas (segundos o minutos de simulación real), las simulaciones de este tipo se hacen inviables.

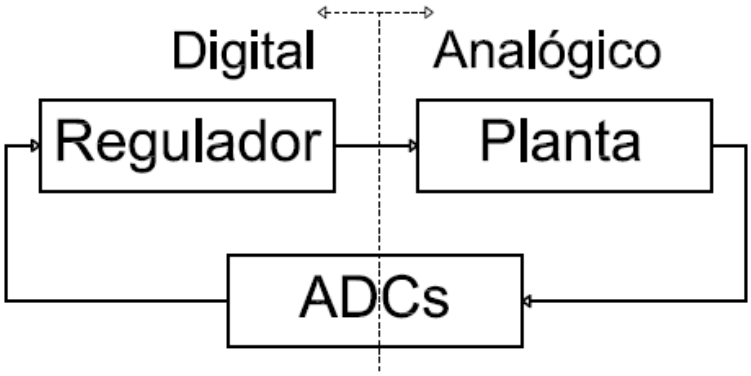


Figura 1. Sistema mixto de un convertidor de potencia controlado digitalmente.

Para resolver el problema de la falta de prestaciones se puede emular el convertidor de potencia en hardware, es decir, crear un modelo matemático del convertidor conmutado e implementarlo en un lenguaje que permita ejecutarlo en hardware. Por ejemplo, utilizando un lenguaje HDL (*Hardware Description Language*) se daría la posibilidad de integrar el diseño en un dispositivo como puede ser una FPGA y así emularlo en hardware real.

Gracias a esta nueva metodología de emulación en tiempo real ya no sería necesario realizar comprobaciones experimentales directamente con el convertidor y, por tanto, se evitarían posibles accidentes, ya que la gran cantidad de energía con la que se trabaja podría provocar daños materiales o personales irreversibles.

Esta técnica que consiste en la digitalización de la planta mediante un modelo matemático y su ejecución en hardware se conoce como HIL (*Hardware-In-the-Loop*), y permite mejorar sustancialmente el rendimiento de la simulación. De esta forma, se consiguen emulaciones en tiempo real con tiempos de integración de hasta decenas de nanosegundos. A pesar de que en este TFG (Trabajo Fin de Grado) se va a utilizar esta metodología para la simulación de una topología en concreto, esta técnica es aplicable para la verificación de cualquier tipo de controlador digital junto con su correspondiente planta. En este TFG en particular se va a implementar el modelo de la planta de un convertidor elevador o *boost*. La característica principal de este convertidor es que proporciona a la salida mayor tensión que a la entrada por lo que también se conoce al sistema como convertidor elevador.

También se mostrará una comparativa entre las diferentes aritméticas utilizadas para modelar el convertidor en VHDL, tomando como referencia un modelo diseñado en *Matlab Simulink* [2]. Al final se ha optado por la notación en coma fija parametrizable ya que este modelo presenta la rapidez y precisión que caracteriza a la coma fija, pero además se dotará al sistema de versatilidad y adaptabilidad, acercándolo a la comodidad de la coma flotante. Con esta nueva técnica se consigue que el modelo pueda adaptarse a nivel numérico a las condiciones de simulación que especifique el usuario final.

Por último, hay que tener en cuenta que en cualquier circuito eléctrico real existen pérdidas. Por ello, se llevará a cabo un estudio para comprobar el efecto en la simulación de estas pérdidas y se generará un modelo que tenga en cuenta las pérdidas de primer orden dentro de la emulación. Los resultados experimentales demuestran que al utilizar la coma fija parametrizable, se consigue modelar el convertidor conmutado elevador con pérdidas eléctricas en tiempo real con tiempos de integración de hasta 31 ns.

Por tanto, el objetivo de este proyecto se centrará en la implementación de un modelo de planta que haga la función de un convertidor elevador con pérdidas en VHDL y en donde la verificación junto con el control sea realizable de forma rápida y que se adapte a las condiciones de simulación.

Con este objetivo, el trabajo fin de grado será estructurado de la siguiente manera. En este primer capítulo se han podido observar las motivaciones y objetivos que han impulsado a la realización de este proyecto. En el segundo capítulo será posible realizar un recorrido por los últimos desarrollos que han sido realizados sobre esta tecnología. Para la correcta realización de un modelo *boost* se ha optado por el siguiente proceso, en el capítulo tres se mostrará el modelado de un convertidor elevador ideal. A continuación, en el capítulo cuatro se realizará el traspaso de este modelo a VHDL, llevándose a cabo un estudio de las diferentes aritméticas posibles. Posteriormente en el capítulo sexto se implementará un modelo que tenga en cuenta las pérdidas eléctricas de nuestro circuito. Por último, en el capítulo séptimo se mostrarán los resultados obtenidos en los diferentes modelos realizados además de un exhaustivo estudio de los valores finales.

2 Estado del arte

Durante la introducción se ha comentado la importancia de realizar las pruebas exhaustivas a un regulador digital junto al convertidor conmutado que es regulado. Tradicionalmente se ha utilizado controles analógicos para regular convertidores conmutados. Sin embargo, en los últimos años se han desarrollado grandes avances en lo que se refiere al control digital. Las ventajas de transformar el control al mundo digital son muy variadas, desde la posibilidad de mejorar la robustez del sistema frente a errores a la posibilidad de incorporar algoritmos más complejos pasando por la monitorización y la mayor integración del diseño [1]. Gracias a ello el crecimiento de esta tecnología ha sido posible, y cada vez hay más ejemplos de reguladores digitales para convertidores conmutados [3,4,5].

En el caso de la verificación de un convertidor conmutado, hay que tener en cuenta que este presenta un componente digital como es el regulador y un componente analógico el caso del convertidor. Por tanto, el modelado del convertidor no será una tarea nada trivial.

Una de las posibles alternativas a la hora de realizar el proceso de simulación es el empleo de sistemas mixtos. En [6] se llevó a cabo un análisis de diferentes modelos de un regulador. Una de las primeras simulaciones que se utilizan es en *Matlab Simulink*, donde se obtienen simulaciones rápidas y con gran precisión, pero con el inconveniente de que el modelo que finalmente se implementará en hardware no será el mismo que el diseñado en esta aplicación. Otra de las pruebas fue utilizar un simulador mixto, *Cadence Spectre Virtuoso* [7], para probar un regulador escrito en HDL junto al esquemático del convertidor. Los resultados mostraron que el tiempo de simulación requerido era muy grande. Por último, se realizó una prueba haciendo uso de modelos genéricos de *Spice*, pero del mismo modo se obtuvieron simulaciones de horas para un único ciclo de conmutación.

En la actualidad hay muy pocos simuladores que permitan las simulaciones mixtas analógico-digital, una de las pocas soluciones consiste en integrar las emulaciones mediante dos simuladores, uno para la parte analógica (*PSIM*) y otro para la parte digital (*ModelSim* [8]). Sin embargo, el problema reside en que el enlace de ambas aplicaciones no es trivial debido a diferentes factores como la sincronización de datos, el flujo de re-alimentación entre aplicaciones, etc.

Otra alternativa de simulación consiste en modelar el convertidor de potencia de forma digital, es decir, transformar el modelo del convertidor conmutado a un lenguaje que permita simularlo junto a su sistema de control. Esta opción permite acelerar las simulaciones [1, 9]. Sin embargo, será tarea del diseñador el hallar las ecuaciones diferenciales que rigen el comportamiento del circuito y su codificación en lenguaje HDL. Hay estudios acerca de qué lenguaje es el mejor para emular este tipo de sistemas, en [10] se lleva a cabo una comparación entre VHDL-AMS y VHDL donde se obtienen que las simulaciones en VHDL son 10 veces más rápidas que en VHDL-AMS.

Para mejorar este proceso, cuando las simulaciones son muy largas se empleará una técnica llamada HIL (*Hardware-In-the-Loop*). Esta técnica consiste en la digitalización de la planta mediante un modelo matemático y su ejecución en hardware para mejorar el rendimiento de la simulación. Hoy en día se está invirtiendo mucho en esta técnica y hay

empresas como *Typhoon HIL* [11], *OPAL-RT* [12], *dSPACE* [13] *e Hypersim* [14] que han desarrollado complejas herramientas de simulación que están hoy presentes en el mercado. Este método ha sido testado con éxito en numerosos estudios como en este trabajo [15] y es empleado en muchas de las ramas de la ingeniería y no sólo en la simulación de convertidores conmutados. En [16] se present un control predictivo de supervisión de una planta de energía y un modelo de una microturbina de gas usando RT-LAB. Por otra parte, en [17] se utilizó el simulador de tiempo real *Hypersim* para implementar un modelo de un enlace VSC-HVDC (*Voltage-Source Converter High Voltage Direct Current*) entre Francia y España, obteniendo 20 us de tiempo de integración. La realización de estos sistemas HIL no es trivial, debido a los bajos pasos de integración que necesitan para dotar al sistema de precisión de cálculo. En este proyecto se va a realizar un modelo de planta de un convertidor en una FPGA con el fin de reducir el tiempo de integración hasta las decenas de nano segundo.

Uno de los mayores problemas a la hora de la realización de estos modelos es que para poder simularlos es necesario una buena elección de la aritmética con la que se va a trabajar. En este proyecto se empleará el lenguaje anteriormente mencionado VHDL para poder diseñar un conmutador elevador. Una vez se haya finalizado el proceso de definición del modelo matemático podemos dar paso a la elección del tipo de notación a emplear en nuestras señales.

La notación *real* es un tipo de aritmética que destaca por su sencillez, ya que no es necesario preocuparse por el tamaño de cada una de las señales ni tampoco de la posición que ocupa nuestra coma. Tampoco es una aritmética que suela sufrir de problemas de resolución, puesto que utiliza coma flotante de 64 bits de resolución: IEEE-754 de doble precisión. Sin embargo, no se puede ejecutar sobre la FPGA ya que no es un lenguaje sintetizable. Existen implementaciones de coma flotante sintetizable, pero los tiempos de integración obtenidos son muy limitados, como se vio en [18, 19], y el área utilizada es muy grande. Otra de las notaciones posibles es la coma fija. Se trata de un tipo de notación cuya característica principal es la mejora de los resultados en cuanto a la velocidad de simulación. Además de trata de un modelo que ya es sintetizable y por tanto puede ser ejecutado por una FPGA. Sin embargo, la gestión de la coma ya no es llevada por el sistema si no que es cuestión del diseñador y por tanto se deberá realizar un esfuerzo extra. Otro de los inconvenientes de este formato es la necesidad de definir los tamaños de las señales en función de los valores que se requieran en el sistema. Esto da lugar a la imposibilidad de utilizar un mismo modelo para diferentes condiciones de simulación.

En este trabajo fin de grado se abordará el diseño de un modelo de un convertidor elevador conmutado utilizando una nueva notación, coma fija parametrizable [20]. Se trata de una nueva técnica que permite aunar las características de la coma fija con la versatilidad que da el poder adaptar la posición de la coma como en coma flotante. De esta forma se consiguen resultados a gran velocidad y a la vez somos capaces de adaptar el modelo a las condiciones de simulación requeridas por el usuario.

3 Modelado de un convertidor conmutado de potencia

Tal como se ha visto en el anterior capítulo, existen diferentes variedades de fuentes conmutadas de diferentes tipos. El modelo de planta que se realizará en este apartado será el de un convertidor elevador (también conocido por su término en inglés *boost converter*). Se trata de un convertidor continua-continua o CC/CC (en inglés DC/DC del término *direct current*) donde la tensión de salida será mayor que la tensión de entrada. A pesar de haber realizado el diseño para un convertidor *boost*, cabe destacar que esta metodología es también aplicable a otros tipos de convertidores.

A lo largo del capítulo se expondrá el modelado de un convertidor conmutado, así como los detalles de su implementación.

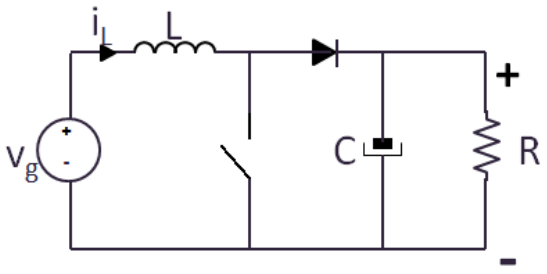


Figura 2. Topología eléctrica de un convertidor elevador.

Se trata de un tipo de fuente conmutada que posee dos interruptores (aunque es habitual sustituir uno de los dos interruptores por un diodo) y al menos un elemento que se encarga de almacenar energía (típicamente un condensador, una bobina o ambas). Además, estos elementos pasivos tendrán la función de regular el rizado de la corriente de entrada y de la tensión de salida.

En este proyecto se va a llevar a cabo el modelado de dicha fuente conmutada a través de VHDL para ser usada en la verificación de reguladores que controlan la planta modelada. Para llevar a cabo este proceso es necesario la obtención de las ecuaciones que rigen el comportamiento de nuestra fuente conmutada (ver Figura 2).

Primero, hay que tener en cuenta que el cálculo de la tensión de salida v_{out} y de la corriente demandada a la entradai_{in} debe realizarse en cada instante de tiempo y de forma continua. En el caso en particular que se ha mostrado anteriormente en la Figura 2 se dan una serie de variables que adquieren los siguientes valores:

$$\begin{aligned} v_{out} &= v_C \\ i_{in} &= i_L \end{aligned} \tag{3.1}$$

Además, hay que mencionar que serán la bobina y el condensador los que impondrán la dinámica a las variables anteriormente mencionadas v_C e i_L haciendo que estas dependan de la inductancia y la capacitancia. Así pues, vamos a considerar las variables v_C e i_L como variables de estado de nuestro modelo. A continuación, vamos a ir describiendo las diferentes ecuaciones para entender el comportamiento de nuestro convertidor. La tensión de caída de la bobina v_L está definida por la ecuación:

$$v_L = L \cdot \frac{di_L}{dt} \tag{3.2}$$

La ecuación diferencial (4.2) se discretiza por ser una variable de estado mediante el método de Euler explícito. Hay que mencionar que este proceso no está exento de pérdidas y por tanto de está cometiendo un error matemático, pero al realizarse con un dt tan pequeño apenas afectará al resultado final. De esta forma se obtiene una ecuación en diferencias en donde la expresión anterior se puede expresar como:

$$i_L(k) = i_L(k-1) + \frac{\Delta t}{L} . v_L(k-1) \tag{3.3}$$

Siendo Δt el paso de integración que se ha utilizado para calcular las variables de estado, y k el instante de tiempo de evaluación de la ecuación y por tanto k-1 será el instante anterior.

De forma análoga, la corriente que atraviesa el condensador de salida i_c se puede describir en cada momento como:

$$i_c = C \cdot \frac{dv_C}{dt} \tag{3.4}$$

Si se vuelve a aplicar el método de Euler explicito, se consigue una ecuación en diferencias que representa la tensión de salida v_C , que en este caso será igual a la tensión que atraviesa el condensador v_C y que se puede definir en un instante de tiempo k como:

$$v_C(k) = v_C(k-1) + \frac{\Delta t}{C} . i_c(k-1) \tag{3.5}$$

De este modo, se han obtenido las ecuaciones que definen las variables de estado del *boost* y, por tanto, las ecuaciones que rigen el funcionamiento de la fuente conmutada.

Con el fin de facilitar la síntesis del modelo propuesto y para transformar estas fórmulas a VHDL, se procede a definir el tiempo de integración Δt como un tiempo fijo, que hará que los términos de las ecuaciones anteriores, $\Delta t/C$ y $\Delta t/L$, sean constantes.

Por otro lado, el funcionamiento principal de nuestro *boost* consiste en dos estados distintos que dependen de la posición en la que se encuentre el interruptor del sistema. Cuando el interruptor está cerrado permite el paso de toda la corriente y, al ser un camino

menos resistivo que los caminos del condensador y de la resistencia de carga, provoca que la corriente que pasa por el diodo sea nula. Además, la corriente que atraviesa al condensador i_C es igual a la corriente que circula por la resistencia de carga i_R. La tensión en bornas de la bobina v_L también es la misma que la tensión de entrada v_g. Este efecto se puede visualizar en la Figura 3.

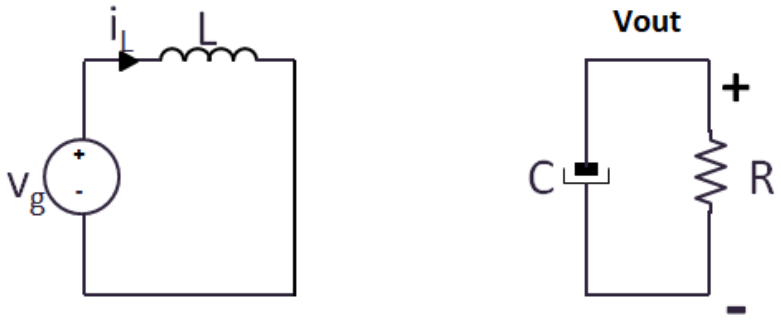


Figura 3. Topología eléctrica de un *boost* con interruptor cerrado.

Con el circuito cerrado, el cálculo de las variables de estado se puede llevar a cabo con las ecuaciones que descritas a continuación:

$$\begin{aligned} i_L(k) &= i_L(k-1) + \frac{\Delta t}{L}.v_g(k-1) \\ v_C(k) &= v_C(k-1) - \frac{\Delta t}{C}.i_R(k-1) \end{aligned} \tag{3.6}$$

El siguiente estado del modelo tiene lugar cuando el interruptor no permite el paso de corriente, es decir, la corriente que atraviesa la bobina recorre también el diodo. En esta situación, cuando el interruptor se encuentra abierto, la corriente de la bobina i_L será la que marcará el estado de nuestro diodo. Cuando i_L sea positiva, el diodo estará en estado de conducción o CCM (en inglés *Continuous Current Mode*) mientras que, si la corriente circula es igual a cero, el diodo se encuentra en estado de no conducción o DCM (en inglés *Discontinuous Current Mode*).

En el caso CCM, la corriente que atraviesa el condensador i_C es igual a $i_L - i_R$, siempre y cuando el interruptor esté abierto y no pase corriente por el interruptor. Tal y como se ha descrito en el párrafo anterior, cuando en el modelo tiene el interruptor abierto y el diodo en estado CCM, el circuito se encuentra en el estado que se muestra a continuación (Figura 4):

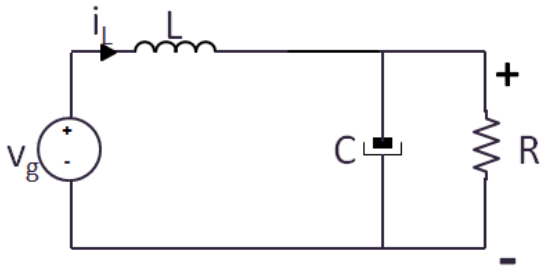


Figura 4. Topología eléctrica de un *boost* con interruptor abierto y diodo en CCM.

En este caso se pueden definir las ecuaciones del modelo de la planta con interruptor abierto y diodo en CCM de la siguiente forma:

$$i_L(k) = i_L(k-1) + \frac{\Delta t}{L}.[v_g(k-1) - v_C(k-1)]$$
$$v_C(k) = v_C(k-1) + \frac{\Delta t}{C}.[i_L(k-1) - i_R(k-1)] \quad (3.7)$$

El último estado del circuito *boost* tiene lugar cuando nuestro interruptor se encuentra abierto y nuestro diodo se encuentra en estado de no conducción, es decir, DCM. En esta ocasión, la corriente que pasa por el diodo será opuesta al sentido de la corriente que atraviesa a la bobina i_L o será nula. Este efecto se puede observar en la Figura 5. En este caso, la corriente que atraviesa el condensador i_C tendrá el mismo valor que la corriente que atraviesa la resistencia de carga i_R.

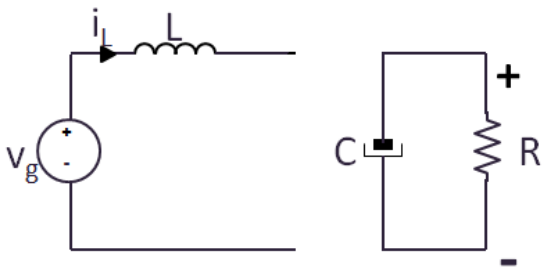


Figura 5. Topología eléctrica de un *boost* con interruptor abierto y diodo en DCM.

Por último, las ecuaciones que se deben implementar en hardware para modelar nuestro sistema son:

$$i_L(k) = i_L(k-1) + \frac{\Delta t}{L}.0 = i_L(k-1)$$
$$v_C(k) = v_C(k-1) - \frac{\Delta t}{C}.i_R(k-1) \quad (3.8)$$

A modo de resumen, la Tabla 1 muestra los resultados de las variables i_C y v_L , según los diferentes estados en los que se encuentre nuestro convertidor elevador, tal y como se ha explicado anteriormente.

INTERUPTOR CERRADO	INTERRUPTOR ABIERTO	
	DIODO EN CCM	DIODO EN DCM
$i_C = -i_R$	$i_C = i_L - i_R$	$i_C = -i_R$
$v_L = v_g$	$v_L = v_g - v_{out}$	$v_L \dashrightarrow circuito\ abierto$

Tabla 1. Soluciones de $\boldsymbol{i_C}$ y de $\boldsymbol{v_L}$ en función del estado del modelo.

Una vez se han definido las ecuaciones que van a sustentar el modelo cabe destacar que, en cada ciclo de reloj, solo se deberá utilizar uno de los tres casos anteriormente explicados.Serán las ecuaciones 3.6, 3.7 y 3.8 las que habrá que modelar en VHDL para realizar el modelado del convertidor conmutado.

4 Alternativas aritméticas durante la implementación

Con el fin de implementar el modelo del convertidor elevador en una FPGA, es necesario transformar todo el diseño a lenguaje HDL. En este trabajo se ha escogido VHDL aunque estos mismos conceptos se podrían aplicar a cualquier otro lenguaje HDL como puede ser Verilog.

El mayor de los problemas al codificar el diseño es la elección de aritmética para llevar a cabo dicha codificación. Para llevar a cabo la elección de la técnica a utilizar habrá que tener en cuenta la complejidad, rapidez y soporte de la FPGA. A continuación, se hará hincapié en cada una de las aritméticas posibles.

4.1 Modelado real *del conmutador elevador*

El primer tipo de señal que va a ser implementado es el tipo ***real,*** un tipo nativo del lenguaje VHDL aceptado por la mayoría de los simuladores. Este tipo de aritmética destaca por su sencillez, ya que no es necesario preocuparse por el tamaño de cada una de las señales ni tampoco de la posición que ocupa nuestra coma. Por eso será el propio software a través de la notación en coma flotante el que realice el ajuste según sea necesario. Para esta precisión, usa un tipo coma flotante de 64 bits de resolución: IEEE-754 de doble precisión.

A pesar de ser la opción más sencilla para el diseñador, tiene un gran inconveniente y es que este tipo de datos anteriormente mencionados no es un formato aceptado para una FPGA y esto hace que el modelo no se pueda simular a tiempo real en una FGPA, es decir, no es sintetizable.

No obstante, el hecho de que sea un modelo no sintetizable no implica que carezca de utilidad, ya que, debido a la gran precisión que arrojan sus resultados, será de gran ayuda para comprobar el funcionamiento adecuado del modelo. Además, tendrá una función crucial a la hora de comparar los resultados de otras aritméticas, ya sea coma fija o parametrizable, con el modelo *real* pudiéndose ver en el Capítulo 7.

Para la elaboración del modelo no hay que olvidar que será necesaria la transformación en lenguaje VHDL de las ecuaciones vistas en el capítulo anterior, de la ecuación 4.6 a la 4.8.

Para llevar a cabo las ecuaciones antes mencionadas, se ha utilizado el hardware que se muestra a continuación en la Figura 6. Para el diseño final del modelo, se han empleado las siguientes unidades: dos multiplexores, dos unidades de registro, dos sumadores y multiplicadores y dos negadores.

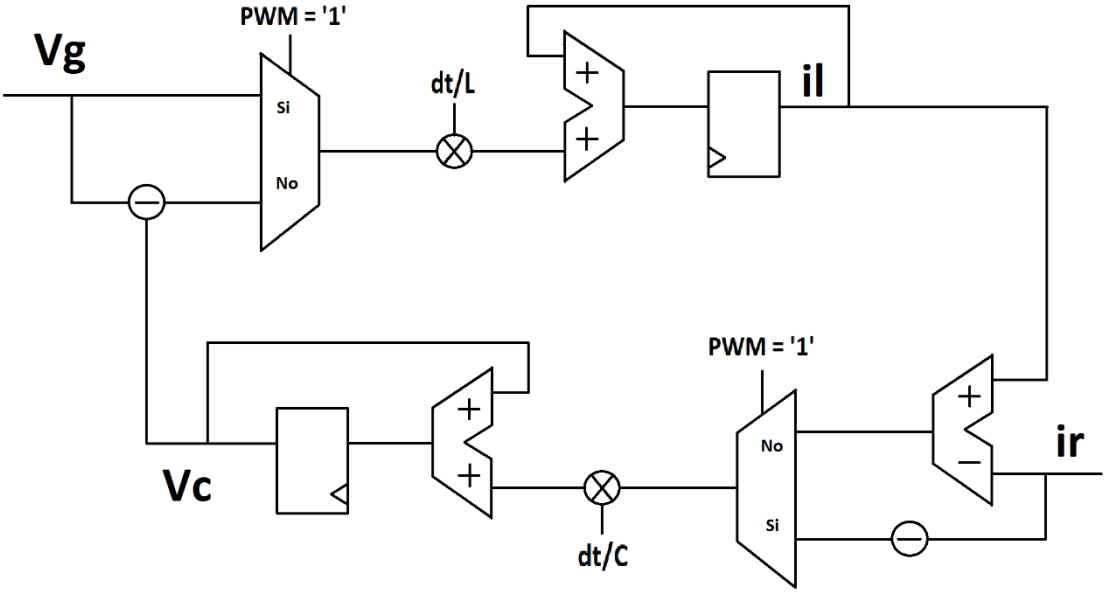


Figura 6. Esquemático del *boost* implementado en *real.*

4.2 Modelado en coma fija del conmutador elevador

Como hemos podido observar con la notación en *real* no es posible la síntesis del modelo. Es por eso que para lograr la síntesis de nuestro diseño se propone el siguiente tipo de aritmética, la coma fija (también llamada formato QX.Y). Se trata de un tipo de notación cuya característica principal es que la gestión de la coma ya no es llevada por el sistema si no que es cuestión del diseñador. Como su nombre indica la coma es estática y será el diseñador del modelo el que deberá decidir cuál es el tamaño apropiado de cada variable y la posición de la coma, siempre habiendo hecho un estudio previo de las condiciones de simulación del modelo.

Este modelo requiere más esfuerzos por parte del diseñador, ya que se trata de un diseño más complejo, pero en contrapartida se consiguen mejores resultados en cuanto al área y la frecuencia. Es cierto que el código en sí se parece mucho al código del modelo *real*, sin embargo, hay que tener en cuenta que para obtener resultados auténticos y no deformados debido al redondeo o por falta de resolución, es necesario un buen trabajo a la hora de acordar el tamaño de las señales. Por eso, deben realizarse una serie de cálculos previos a la trasformación de las ecuaciones 4.6, 4.7 y 4,8 vistas en el Capítulo 3, para tener en mente los posibles desbordamientos (también llamado *overflow* en inglés) y adaptar perfectamente la resolución de las señales.

Como ya se ha comentado anteriormente, la implementación en coma fija no es una tarea trivial, debido al gran número de cálculos previos que se requieren. Por eso, para describir el diseño la biblioteca que se propone es s*fixed*, esta se encuentra dentro del proyecto *VHDL-2008 Support Library [25]* y nos permitirá la traducción de número en coma fija a VHDL. A pesar de que esta librería facilita la etapa de implementación, es importante destacar que siguen siendo necesarios los cálculos de los tamaños de los registros, así como el diseño de las resoluciones.

Este lenguaje permitirá realizar la simulación y la síntesis del modelo gracias a su tipo de datos. Para el diseño, se ha implementado la coma fija con signo, ya que se considera la posibilidad de que existan algunas corrientes o tensiones negativas, es decir, en sentido opuesto a los acordados en los diagramas.

Las señales en coma fija pueden utilizar la notación QX.Y en complemento a dos, que consiste en X bits de parte entera, Y bits de parte decimal y un bit adicional que permite signo. Por ejemplo, la representación Q3.3 tendrá en total un número de bits de 1+3+3. A continuación, se puede observar un ejemplo de representación. El número en binario "010010" quedaría representado en Q3.3 como 010.010, y en decimal sería 2.25. Para la obtención del número en decimal se sigue la notación en complemento a dos descrita en la Figura 7.

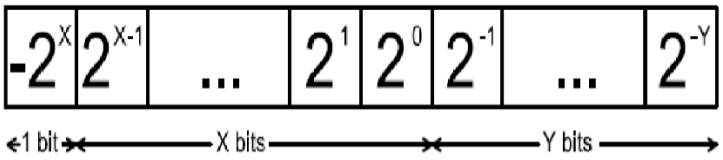


Figura 7. Formato de una señal en notación QX.Y.

Cabe destacar que para la realización de operaciones a través de la notación en coma fija no se debe utilizar las LUTs (*LookUp Table*) presentes en la FPGA, sino que estas se deberán utilizar los DSPs de la FPGA implementados en silicio. Esto es debido a que el rendimiento de los DSPs de la FPGA es muy superior a las LUTs, lo que permitirá obtener tiempos de integración menores y, por tanto, mayor precisión en el modelo. Por ello será un requisito vital adaptarse al tamaño de datos con los que trabaja el DSP. En particular, habrá que tener especial cuidado con el tamaño de los operandos de las multiplicaciones, puesto que es la operación que más penaliza en el caso de realizarse a través de LUTs. Dado que se va a utilizar una FPGA Xilinx Zynq-7000, sus DSPs pueden implementar multiplicaciones con dos operandos de 17 y 25 bits respectivamente.

El tamaño de estos multiplicadores embebidos será un factor a respetar y por tanto es necesario adaptarse a su tamaño, por lo que esto desemboca en una longitud de 17 bits para las variables del modelo y un total de 25 bits para las constantes de entrada que están presentes en el diseño. Sin embargo, dentro del circuito se darán otro tipo de operaciones que no necesitarán de estos bloques para su realización y que, por tanto, no tendrá una limitación aparente de bits. Este es el caso de las sumas y los registros de datos, donde se utilizará la lógica programable.

Tal y como se ha mencionado anteriormente, la elección del tamaño de las variables es un punto crítico dentro de la implementación del diseño. Un error en el número de bits de la parte entera puede desencadenar en una representación totalmente incorrecta de un resultado. En este caso se va a proceder al modelado de las constantes del circuito. Los valores dtL y dtC serán modelados con un total de 25 bits tal como se ha especificado en el párrafo anterior.

No podemos continuar sin comentar la necesidad de aumentar el tamaño de la señal que compone el acumulador de nuestro modelo *boost.* En dicho bloque tienen lugar una serie de pequeños incrementos que pueden dar lugar a números muy grandes y es debido a este

efecto por lo que nos vemos obligados a incrementar el tamaño de nuestra señal. Para calcular el aumento de tamaño se considera que el máximo número de ciclos que nuestro sistema puede estar incrementándose es 8192 y por tanto en bits sería $\log_2(8192) = 13\ bits$, es decir debido a esos incrementos necesitará 13 bits adicionales. Por tanto, al lograr conmutar a 32 MHz (en tiempo 31 ns) sería perjudicial para nuestro convertidor que esté estuviera operando siempre en el modo ON u OFF durante más de
31 ns · 8192 = 253.9 µs. Si esto se diera, la frecuencia de conmutación de control podría lograr forzar al modelo a dar resultados incorrectos.

Para las variables v_g, v_C e i_R del circuito se han utilizado los 17 bits comentados previamente. A continuación, en la Tabla 2 se muestran los diferentes formatos de las señales en el modelo en coma fija. Además, se mostrarán a modo comparativo la diferencia entre las señales v_C y $v_C{}'$ donde $v_C{}'$ se corresponde con la salida del acumulador por lo que se le dotará de mayor resolución.

Señal	Formato	Precisión	Rango equivalente (3 decimales)	Resolución
v_g	Q6.10	17 bits	± 63,999 V	$9{,}765 \cdot 10^{-4}$ V
v_C	Q6.10	17 bits	± 63,999 V	$9{,}765 \cdot 10^{-4}$ V
$v_C{}'$	Q6.49	56 bits	± 63,999 V	$1{,}776 \cdot 10^{-15}$ V
i_L	Q3.13	17 bits	± 7,999 A	$1{,}220 \cdot 10^{-4}$ A
$i_L{}'$	Q3.49	53 bits	± 7,999 A	$1{,}776 \cdot 10^{-15}$A
i_R	Q3.13	17 bits	± 7,999 A	$1{,}220 \cdot 10^{-4}$ A
dtL	Q-15.39	25 bits	$\pm 3{,}051 \cdot 10^{-5}$ s/H	$1{,}455 \cdot 10^{-11}$ s/H
dtC	Q-12.36	25 bits	$\pm 2{,}441 \cdot 10^{-4}$ s/F	$1{,}818 \cdot 10^{-12}$ s/F

Tabla 2. Formato de las señales en QX.Y.

Decir el dt/L calculado en este ejemplo es X, y que el primer decimal significativo corresponde a la posición -15. Por tanto, consideramos que el MSB después del signo es el -15 y por tanto con 24+1 (el 1 de signo) bits llegamos al decimal 39.

De la tabla anterior hay que mencionar algunos elementos negativos que aparecen en la notación QX.Y. Por ejemplo, para el cálculo del formato de dtL el primer bit más significativo se encuentra en la posición -15. Esto hace que si en total tenemos 25 bits donde uno es el bit determinado para el signo, el cálculo la posición Y se obtiene de -15 – 24 = -39 donde llegamos hasta el decimal 39.

Como se ha podido apreciar en la tabla las tensiones v_g y v_C solo admiten 64 V, si tenemos la necesidad de aumentar los voltajes será obligatorio la modificación de la posición de la coma en toda variable del modelo y por supuesto volver a sintetizar el diseño. La elección de los tamaños no es un trabajo trivial, ya que es necesario conocer previamente las condiciones de simulación del modelo para poder elegir

adecuadamente los valores máximos de las variables. Pero como ya hemos explicado anteriormente, el problema es si después estos valores son modificados.

A continuación, se realizará un breve análisis de la metodología utilizada en la elección de los formatos según el tipo de operación. Para esta elección se ha seguido el proceso que aparece en la Tabla 3.

Tipo de operación	Rango del resultado
QX'.Y' + QX''.Y''	Q Max(X',X'') + 1. Min(Y',Y'')
QX'.Y' - QX''.Y''	Q Max(X',X'') + 1. Min(Y',Y'')
QX'.Y' * QX''.Y''	Q X'+X'+ 1. Y'+Y''

Tabla 3. Tamaño de las señales según la operativa.

En la Figura 8 se muestra el esquemático del convertidor *boost*, diseñado para la notación en coma fija. En las señales del circuito queda representados sus formatos mediante la notación QX.Y.

Tal como como se ha podido apreciar, el modelo realizado en coma fija presenta un nivel de complejidad superior al modelo en real debido a la necesidad de tener en cuenta cada operación y tamaño. Sin embargo, cabe destacar que es un modelo sintetizable por una FPGA y que ofrece la posibilidad de llegar a la relación óptima entre precisión y velocidad.

A pesar de esto, la implementación del modelo en notación QX.Y posee otra carencia significativa, y es que se trata de un modelo que no es adaptable a cambios en la simulación. Cualquier modificación en los parámetros en nuestro diseño traerían consigo un esfuerzo extra por parte del diseñador, ya que sería necesario volver a calcular todos los tamaños y tener en cuenta de nuevo todas las operaciones que se realizan en nuestro modelo.

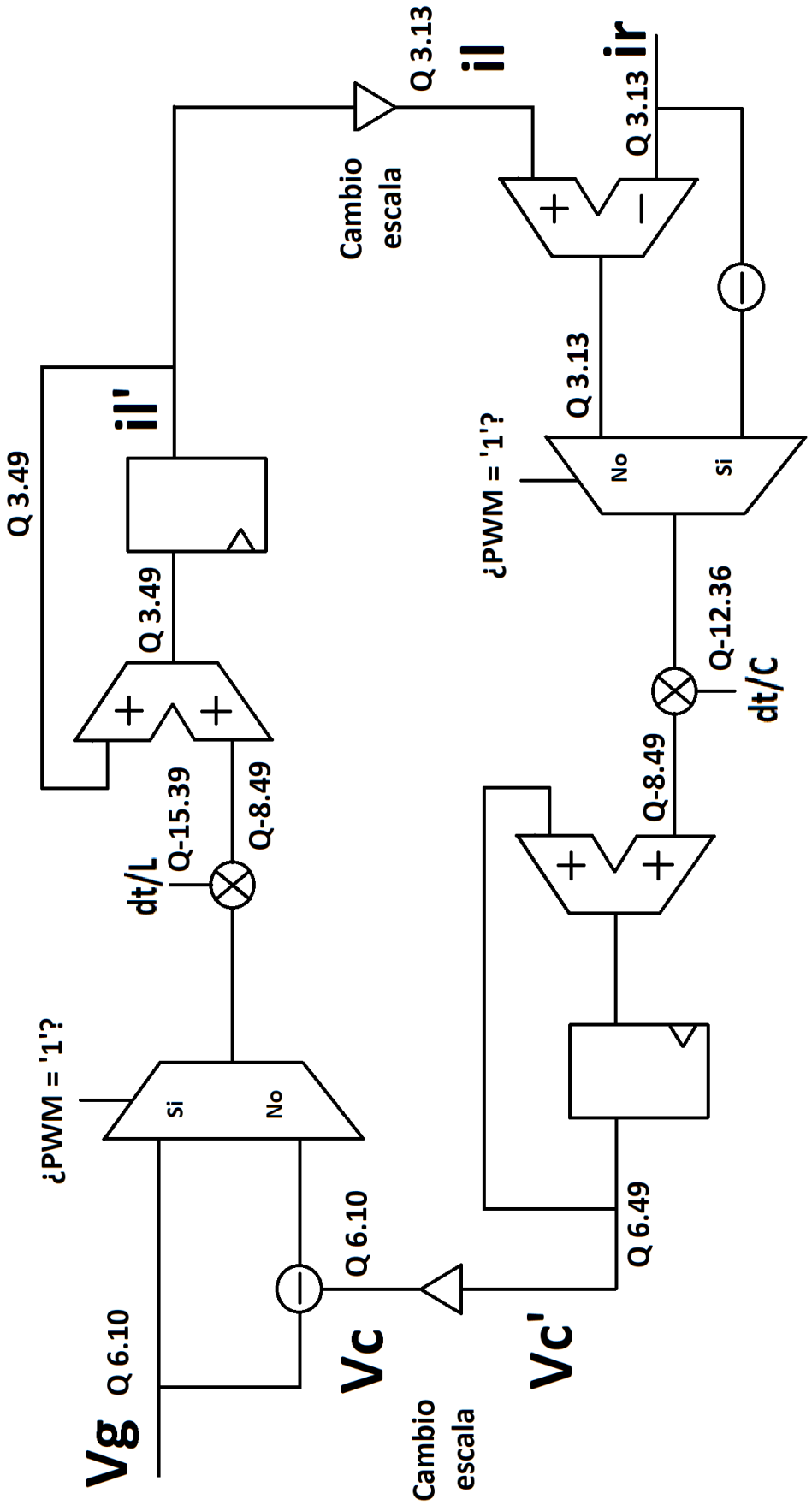


Figura 8. Esquemático del *boost* implementado en coma fija.

4.3 Modelado en coma fija parametrizable del conmutador elevador

Tal y como hemos podido apreciar con anterioridad, a pesar de que el modelo en coma fija es sintetizable si es necesario la modificación de las condiciones de simulación traería consigo la resintetización y rediseño del modelo. Por tanto, para finalizar este recorrido por los distintos tipos de aritmética, es necesario presentar la notación que se ha definido como coma fija parametrizable, que permite adaptar el modelo modificando la posición de la coma sin la necesidad de resintetizar el diseño. Se trata de un modelo de representación basado en la coma fija, pero que ha sufrido diversas modificaciones.

Antes de la implementación de este modelo, cabe mencionar que se ha tenido que realizar un cambio de tipo de datos, se ha empleado el tipo *std_logic_vector* con lo que en este tipo de notación trabajaremos con vectores simples. Este cambio ha venido derivado de que como se explicará posteriormente no hay una posición fija de la coma y por tanto no tiene sentido seguir trabajando con la biblioteca *sfixed.*

Para comenzar con este modelo, se parte de la misma estructura que en los modelos anteriores, pero es de vital importancia definir el tamaño total de las señales mediante un método diferente. Con este nuevo modelo se diseñan el tamaño de las señales a través de un único vector que podrá ser interpretado en complemento a dos, y además a priori este vector no necesita tener en cuenta los tamaños de la parte entera y decimal.

El factor más importante dentro de esta notación es que para determinar el valor definido por esta aritmética es necesario definir las escalas de las señales. Estas escalas serán el elemento clave para la trasformación de estos números en binario a su formato decimal, ya que será el número total en complemento a dos el que habrá que dividir por la escala para obtener el resultado final. Un ejemplo de esto es que, si se dispone del siguiente valor en binario “0010011”, en decimal sería 19 y si la escala fuera 2 el valor de la señal sería en 9,5.

La peculiaridad de este modelo es que la coma no es fija, es decir, se puede mover y acondicionar según los valores gracias a las escalas. De este modo, se podrá dotar al modelo de versatilidad frente a posibles cambios en las condiciones de simulación.

El principal objetivo consiste en realizar un modelo que sirva para simular cualquier convertidor *boost* y proporcionarlo de total independencia frente a las condiciones de simulación tales como la corriente, las tensiones, etc. Es decir, serán únicamente configurados los cambios de escala necesarios para adaptar el modelo a los valores requeridos de tensiones y corrientes. Una de las ventajas de este modelo es que el código diseñado no tendrá que ser modificado si cambian las condiciones de simulación, sino que el modelo solo se sintetizará una sola vez y únicamente se deberán configurar los parámetros para que la simulación se adapte a las nuevas condiciones de contorno.

Tal como hemos comentado anteriormente, una de las tareas que debe realizar el usuario de nuestro modelo es el cálculo de las escalas de las señales que se van a utilizar. Por ello, se va a realizar una breve descripción de cómo debe de realizarse este cálculo. El primer paso de todos es la obtención del número total de bits necesarios para representar el valor máximo de nuestra señal, para ello se realizar $\log_2(Valor\ máximo)$. Por ejemplo, si el número a representar es el 9, $\log_2(9) = 3{,}16$ el cual se redondea siempre hacia el valor más alto, es decir, el número de bits necesarios para la parte entera serán 4. Una vez calculada

la parte entera, de una manera muy sencilla podemos realizar el cálculo de nuestra parte decimal que recibe el nombre de escala. Sabiendo el tamaño total posible de nuestra señal, normalmente limitado por el multiplicador, para la obtención de la escala no tendremos más que restar al tamaño de nuestra señal el número de bits de enteros y así obtendríamos la parte decimal y por tanto la escala. Para seguir el ejemplo anterior, si dispongo una señal con un tamaño total de 17 bits, la escala para acondicionar la señal será $17 - 4 = 13$ bits de parte decimal y por tanto valor de nuestra escala.

Es importante recordar que el modelo a implementar tiene una dependencia directa con la FPGA que se va a utilizar, en este caso Xilinx Zynq-7000 y que, tal como se ha indicado en el anterior apartado, posee multiplicadores de 25x17 bits. A partir de estos valores se empieza a diseñar el tamaño de las señales. Las constantes tendrán un tamaño total de 25 bits, mientras que las señales retroalimentadas utilizarán 17 bits.

Para la realización del modelo, es necesario solicitar los valores de algunos de los parámetros de nuestro circuito, entre ellos uno de los datos más relevantes son los valores máximos entre la tensión de entrada y de salida, v_g y v_{out}. Además, se deben conocer los valores de la bobina, el valor del condensador y con ello calcular las escalas de las transformaciones. A continuación, en la Figura 9 se muestra el diagrama completo de nuestro convertidor *boost* en coma fija parametrizable. En esta figura se puede apreciar además el tamaño de las señales que tiene el sistema.

Tal como hemos mencionado con anterioridad, a continuación, se muestra el diagrama de nuestro convertidor *boost* en la Figura 9.

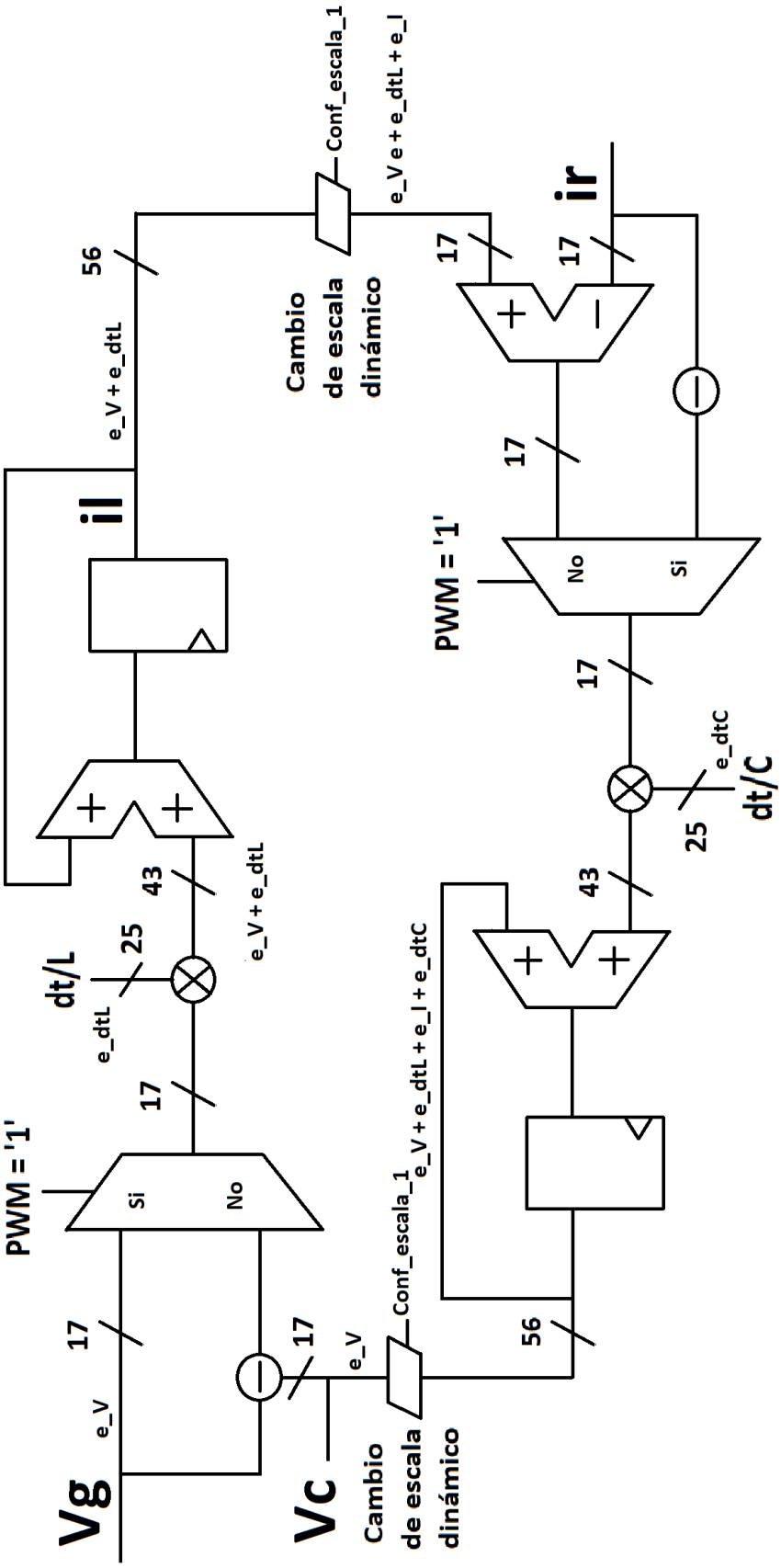


Figura 9. Esquemático del *boost* implementado en coma fija parametrizable.

5 Incorporación de pérdidas eléctricas

El modelo explicado en el capítulo anterior se corresponde con un convertidor conmutado donde todos los elementos son ideales, pero en la realidad este comportamiento ideal no ocurre y hay que tener en cuenta varios factores si se desea una simulación realista. Evidentemente, la inclusión de pérdidas eléctricas hace que el modelo sea más complejo, tanto a la hora de diseñarlo, como en su implementación, por lo que el paso de integración mínimo que permita tiempo real se verá afectado. En este capítulo se van a estudiar las pérdidas eléctricas de primer orden y se detallará el diseño del modelo con dichas pérdidas. Posteriormente, en el capítulo XX se mostrarán los resultados del modelo con y sin pérdidas y su variación en el tiempo de integración mínimo.

5.1 Modelo de un convertidor boost *con pérdidas*

Para modelar el comportamiento real de un convertidor conmutado con pérdidas eléctricas es necesario conocer cada uno de los componentes que hay en el diseño. No hay que olvidar que es necesaria la transformación de estos efectos en lenguaje VHDL y que esta transformación basará en el modelo ideal que se ha presentado en el apartado 3.1.

La bobina, tal como se muestra en la Figura 10, es el primer elemento en añadir pérdidas eléctricas al conmutador elevador. El cobre interior del interruptor causa las pérdidas, que pueden modelarse como una resistencia $R_L$que deberá incorporarse al conmutador.

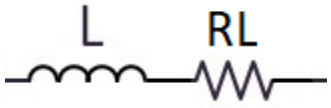


Figura 10. Diseño de las pérdidas causadas por la bobina.

El segundo elemento al que se le pueden atribuir las pérdidas del circuito es el condensador. Como primera aproximación, se pueden modelar las pérdidas causadas por dicho elemento como una resistencia en serie R_C. Es más, esta resistencia también conocida en el mundo de la electrónica por el término ESR (*Equivalent Series Resistance*) o resistencia serie equivalente. Las pérdidas de este elemento finalmente se pueden interpretar como se muestra en la Figura 11.

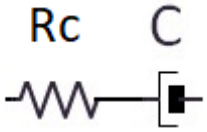


Figura 11. Diseño de las pérdidas causadas por un condensador.

El siguiente elemento a estudiar es el diodo. En este caso, el efecto provocado por el componente se puede modelar mediante una primera aproximación como una fuente de tensión V_D a la que se añade una resistencia en serie R_D. Esta aproximación se puede modelar como se muestra en la Figura 12.

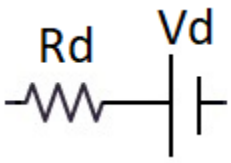


Figura 12. Diseño de las pérdidas causadas por el diodo.

El último elemento que puede añadir pérdidas al modelo es el interruptor. Como primera aproximación se puede representar las pérdidas causadas por el elemento como una resistencia en serie R_S. Por ello, se obtiene un circuito equivalente al que queda representado en la Figura 13.

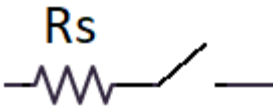


Figura 13. Diseño de las pérdidas causadas por el interruptor.

Debido a la intervención de los elementos mencionados anteriormente, el resultado del circuito del convertidor *boost* es el que se muestra en la Figura 14. Se puede apreciar cómo se parte de la base del circuito ideal al que se le han añadido cada uno de los componentes adicionales que causan las pérdidas en el resultado final. A lo largo de este capítulo, se va a realizar el desarrollo de las nuevas ecuaciones que rigen el comportamiento del convertidor conmutado con base en todos los cálculos en lo anteriormente explicado.

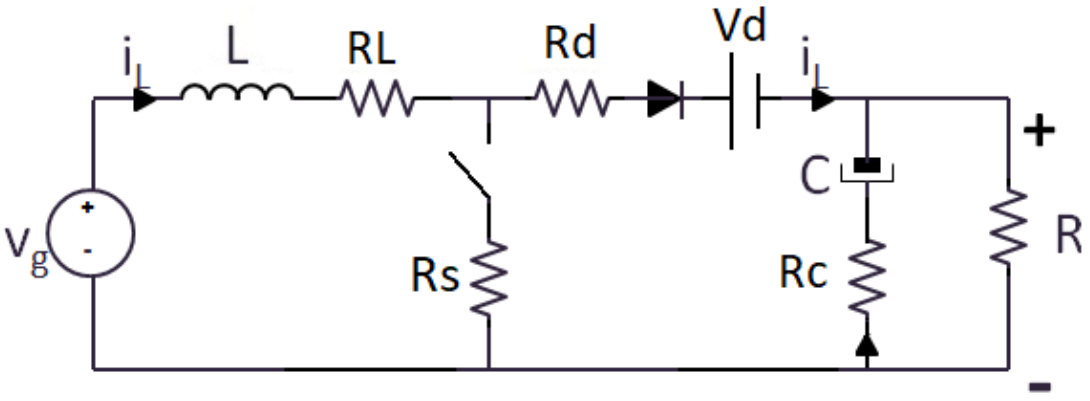


Figura 14. Topología de un convertidor elevador con pérdidas.

Para poder desarrollar el modelado del circuito se seguirá una metodología similar a la utilizada en el capítulo 3.1., por lo que se hará un estudio más exhaustivo en las diferencias que presentan ambos modelos. Primero, debe señalarse que ambos diseños utilizan las

mismas ecuaciones de estado. i_L ya se ha definido como la corriente que atraviesa la bobina y v_C como la tensión que cae en bornas del condensador

$$i_L(k) = i_L(k-1) + \frac{\Delta t}{L}.v_L(k-1)$$
$$v_C(k) = v_C(k-1) + \frac{\Delta t}{C}.i_c(k-1) \quad (5.1)$$

Como ya se explicó en el capítulo 3.1, la discretización de las variables de estado se ha realizado mediante el método de Euler explícito. De este modo, se han representado las ecuaciones que definen las variables de estado del conmutador elevador.

Las diferencias más significativas respecto al modelo ideal vienen dadas en función del estado de los conmutadores del circuito. Como ya se presentó en el capítulo 3.1, el modelo presentaba tres estados que podían variar en función del estado de los conmutadores del circuito. El primer estado sucede cuando el interruptor se cerraba y, por tanto, permitía todo el paso de la corriente. Es el caso que se puede visualizar en le Figura 15. En este caso en particular, se deben tener en cuenta las resistencias R_L y R_S del modelo a la hora de calcular v_L. Además, la corriente que circula por un condensador i_C coincide con i_R y por eso no dependerá directamente de la resistencia serie del condensador R_C.

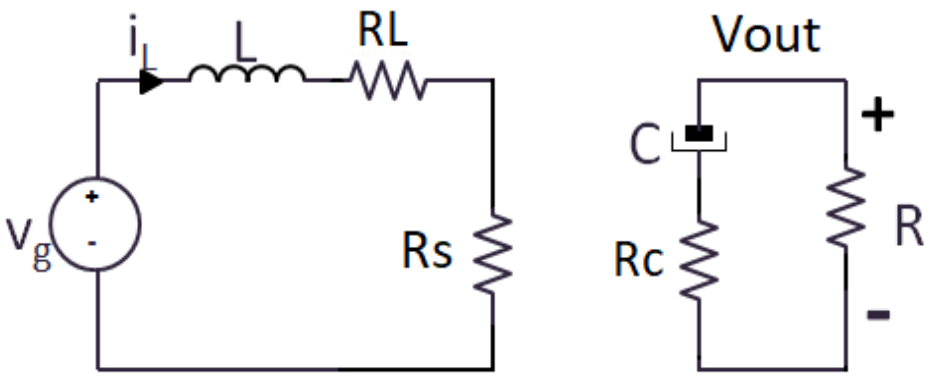


Figura 15. Topología eléctrica de un *boost* con pérdidas e interruptor cerrado.

Con el circuito en este estado, se pueden calcular las variables que quedan definidas en las siguientes ecuaciones:

$$i_L(k) = i_L(k-1) + \frac{\Delta t}{L}.[v_g(k-1) - i_L(k-1)\cdot(R_L + R_S)]$$
$$v_C(k) = v_C(k-1) - \frac{\Delta t}{C}.i_R(k-1) \quad (5.2)$$

El siguiente estado del modelo tiene lugar cuando el interruptor se encuentra abierto y, por tanto, no permite el paso de la corriente. En esta situación, el estado del circuito lo establece el estado del diodo.

En primer lugar, en el caso CCM, es decir, en estado de conducción, la topología que tiene lugar es la que se muestra en la Figura 16. En este estado, hay que tener en cuenta que la tensión que cae en bornas de la bobina v_L no es la misma que la estudiada en el caso ideal. Esto se debe al efecto de las pérdidas que ha sido introducido por el diodo como $V_D y\ R_D$.

Por otro lado, la corriente que atraviesa el condensador i_C es igual a $i_L - i_R$.

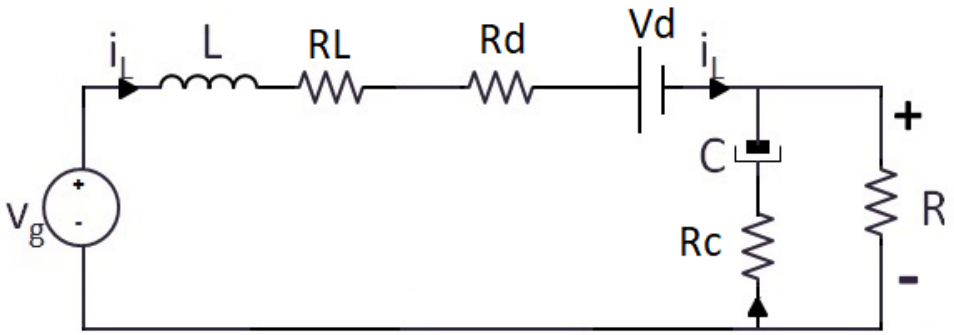


Figura 16. Topología de un *boost* con pérdidas interruptor abierto y diodo CCM.

Cuando el circuito presenta el estado anterior, las ecuaciones del modelo de la planta son las siguientes:

$$i_L(k) = i_L(k-1) + \frac{\Delta t}{L}.[v_g(k-1) - v_C(k-1) - i_L(k-1)\cdot(R_L + R_S)]$$
$$v_C(k) = v_C(k-1) - \frac{\Delta t}{C}.[i_L(k-1) - i_R(k-1)] \quad (5.3)$$

Para el último caso, el interruptor se encuentra abierto y el diodo está en el estado DCM. El resultado de las ecuaciones diferenciales para este estado es igual al modelo sin pérdidas y este efecto se puede observar en la Figura 17.

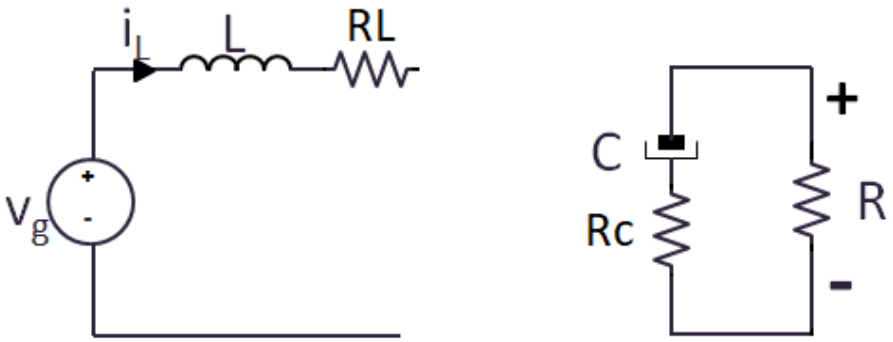


Figura 17. Topología de un *boost* con pérdidas, interruptor abierto y diodo DCM.

Por último, las ecuaciones que se deben implementar en hardware para modelar el comportamiento de la planta son:

$$i_L(k) = i_L(k-1)$$
$$v_C(k) = v_C(k-1) + \frac{\Delta t}{C}.[-i_R(k-1)] \quad (5.4)$$

A modo de resumen y como se realizó en el capítulo 4, se muestra en la Tabla 4 los resultados de las variables i_C y v_C según el estado en el que se encuentre el convertidor *boost*.

INTERUPTOR CERRADO	INTERRUPTOR ABIERTO	
	DIODO EN CCM	DIODO EN DCM
$i_C = -i_R$	$i_C = i_L - i_R$	$i_C = -i_R$
$v_L = v_g - i_L \cdot (R_S + R_L)$	$v_L = v_g - v_C - V_D - i_L \cdot (R_D + R_L)$	$v_L \dashrightarrow circuito\ abierto$

Tabla 4. Soluciones de $\boldsymbol{i_C}$ y de $\boldsymbol{v_L}$ en función del estado del modelo con pérdidas.

Para finalizar, debemos destacar que en cada ciclo de reloj solo se deberá utilizar uno de los tres casos anteriormente mencionados. Además, serán las ecuaciones 5.2, 5.3 y 5.4 las que habrá que modelar para realizar el modelado del convertidor elevador.

Como ya se ha mencionado en el capítulo anterior, para poder modelar el convertido elevador en una FPGA es necesario utilizar un lenguaje HDL, en este caso VHDL. Al igual que con el modelo del convertidor *boost* ideal, se deben recorrer distintos tipos de aritmética y estudiar cada una de las variantes. Cabe destacar que no se ha realizado el modelo en coma fija, puesto que es un hito intermedio entre la notación real y la parametrizable y carece de utilidad. A continuación, se realizará un breve recorrido por los diferentes tipos de aritmética implementadas.

5.2 Modelado real *del conmutador elevador con pérdidas*

A continuación, se hará un breve resumen de la trasformación del modelo convertidor *boost* a VHDL mediante la notación *real*. Como ya se comentó en el apartado 4.1, se trata de un tipo de aritmética que destaca por su sencillez. El diseñador no debe preocuparse por el tamaño de las señales o la posición de la coma, ya que será el propio sistema el que se encargue de gestionar estos problemas.

A pesar de tratarse de un modelo sencillo, debe dejarse muy claro que no se puede implementar dentro de una FPGA, puesto que es un lenguaje no sintetizable. Esto hace que este tipo de notación no sea válido para el objetivo de este proyecto, que es la simulación de un convertidor elevador en una FPGA. Sin embargo, aunque no sea sintetizable este modelo resulta de gran utilidad ya que servirá de ayuda a la hora de verificar si se ha llevado a cabo correctamente la discretización y la transformación de las ecuaciones que rigen el comportamiento del modelo.

Para la elaboración del modelo, se ha utilizado el hardware que se muestra a continuación en la Figura 18. Mediante este esquema, quedan reflejadas las ecuaciones de diferencias que se han obtenido en el apartado 5.1.

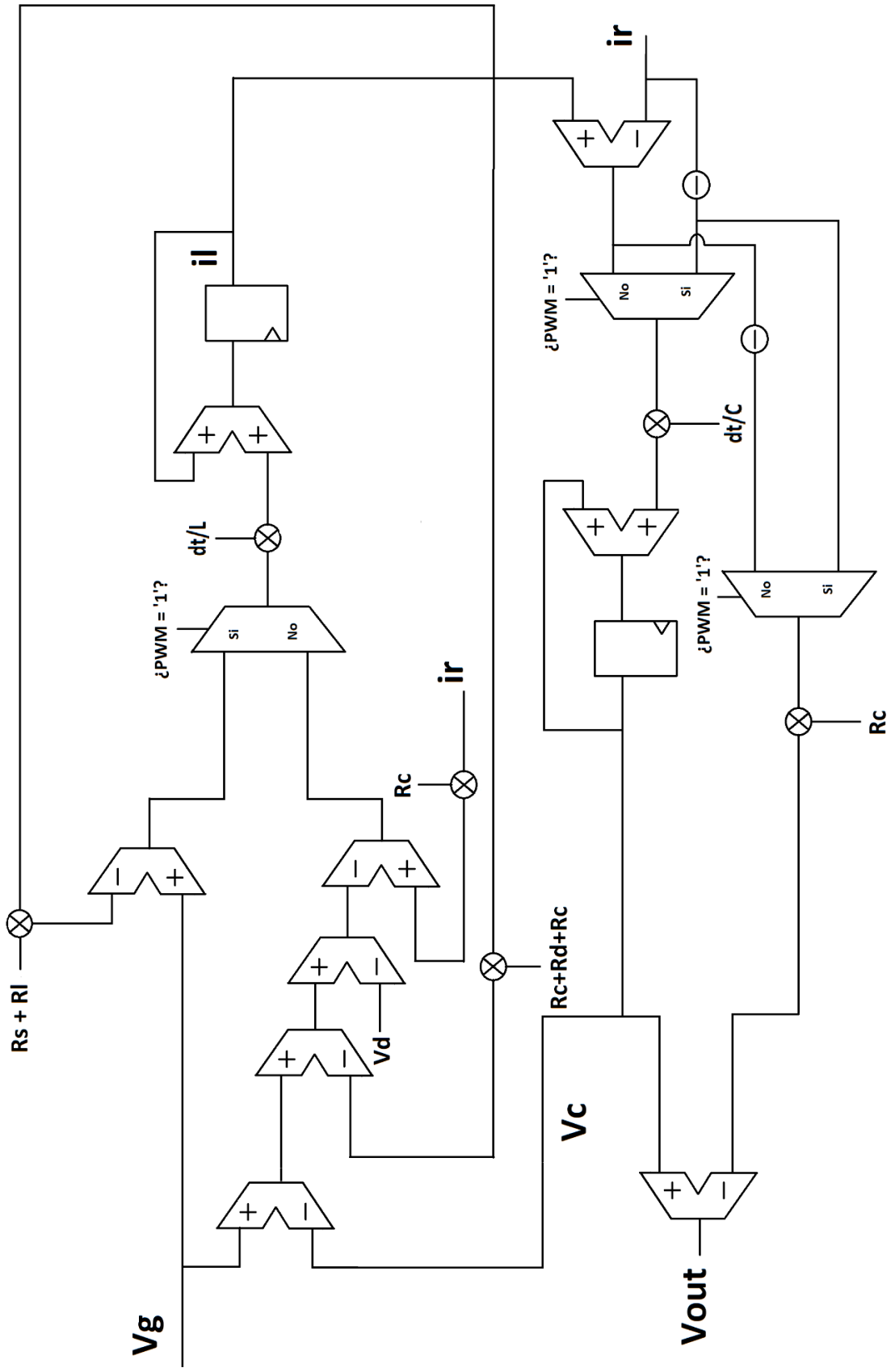


Figura 18. Esquemático de un *boost* con pérdidas implementado en *real*.

5.3 Modelado con coma fija parametrizable del conmutador elevador con pérdidas

Tal y como se ha comentado en el apartado anterior, la notación *real* no es una opción viable para el objetivo del proyecto debido a que no puede ser implementado en una FPGA. Por ello se debe a utilizar otro sistema de representación. La notación en coma fija tampoco es una solución útil, ya que obliga a recalcular y resintetizar todo el modelo si se modifican las condiciones de simulación. Así pues, este apartado muestra la implementación del modelo con pérdidas eléctricas utilizando la notación en coma fija parametrizable.

Hay que recordar que se trata de un modelo en el que se permite adaptar la posición de la coma a diferentes situaciones de simulación. Además, otro de los cambios más significativos y como ya como se mencionó en el apartado sin pérdidas, para la realización de este modelo se utilizarán vectores simples empleando el tipo *std_logic_vetor.*

Uno de los primeros requisitos primordiales a la hora de diseñar el modelo en coma fija parametrizable es la definición del tamaño total de las señales, labor que debe realizar el diseñador. Los elementos ya presentes en el modelo sin pérdidas mantienen su tamaño para este nuevo diseño, excepto las señales que ahora dependan de las nuevas pérdidas.

Durante el proceso de cálculo, tal y como se aprecia en la Figura 19, se ha tenido que extender algunas de las señales e incrementar el número de bits como en v_g y v_C . Esto se ha hecho debido que puede haber algunos valores de pérdidas tan pequeños que es necesario aumentar la precisión de estas señales con el fin de que puedan tener influencia real en el modelo. Todas las entradas y salidas presentan los mismos tamaños que en el modelo sin pérdida, donde se utilizaban 17 o 25 bits. En este caso hay una excepción, ya que Vd debe entrar al sistema con el mismo tamaño que la otra señal a sumar y, por tanto, ha de ser una entrada de 19 bits.

Uno de los objetivos principales de este nuevo diseño es proporcionar al modelo total independencia frente a posibles cambios en las condiciones de simulación. Para ello, el diseñador será el que, para acondicionar el modelo a sus especificaciones, deberá calcular las escalas con el fin de mover la posición de la coma de cada señal. A pesar de que este cálculo se pueda realizar más de una vez, una de las ventajas más importantes del sistema es que no implica la resintetización del modelo.

El nuevo esquemático con sus correspondientes pérdidas es representado en la Figura 19 y con el mero hecho de observar la figura se puede apreciar el aumento de complejidad que ha sufrido el diseño. Debido a su complejidad, este diseño necesitará más elementos hardware, como por ejemplo LUTs y DSPs; y además sufrirá una ralentización en el proceso de simulación. Sin embargo, su comportamiento se acercará de forma notable al convertidor real.

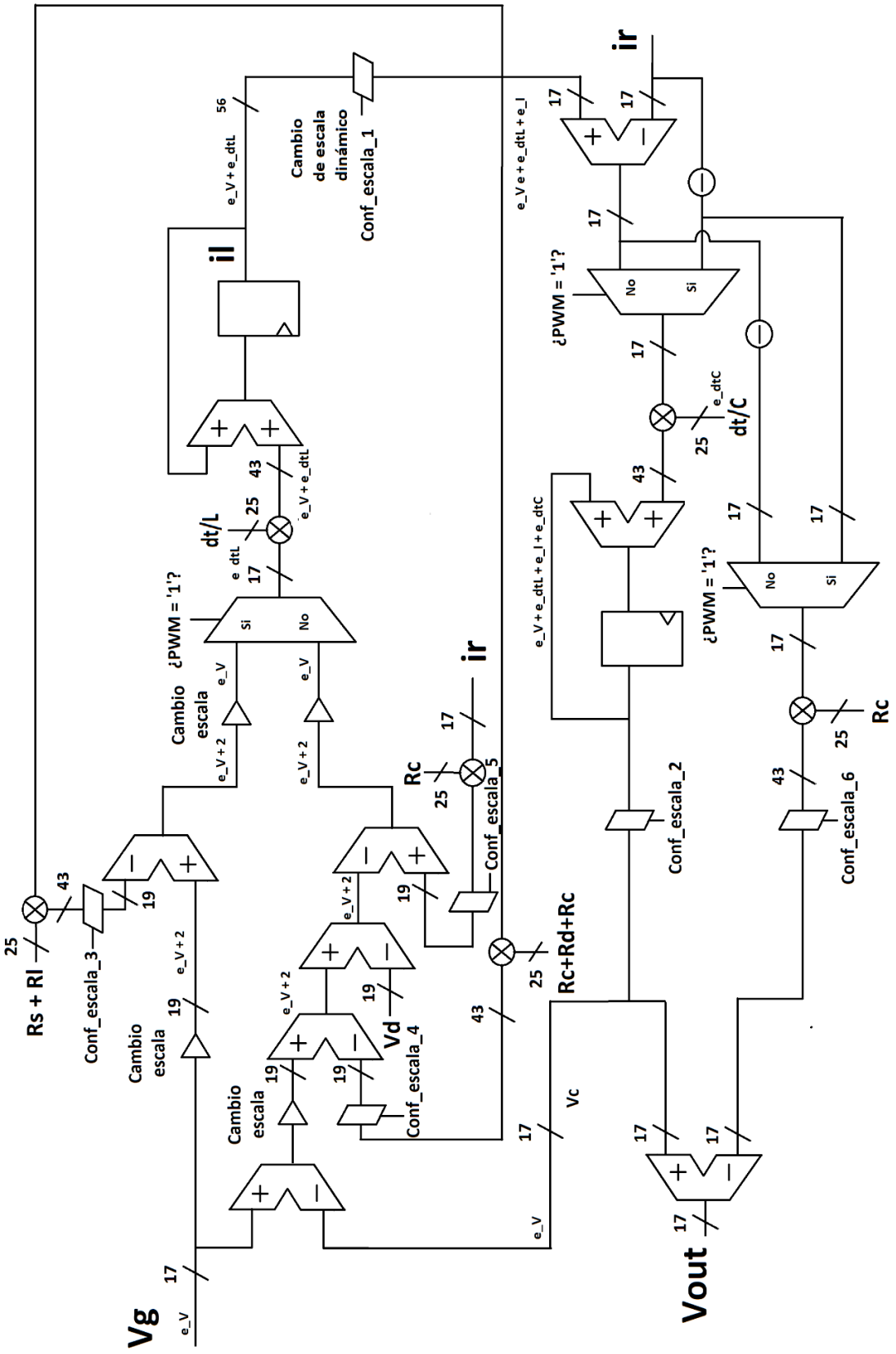


Figura 19. Esquemático de un *boost* con pérdidas implementado en coma fija parametrizable

6 Resultados experimentales

No existe una metodología estándar que rija la verificación de reguladores digitales para convertidores conmutados, ya que, como se ha explicado anteriormente, no es una tarea trivial debido a la unión de un elemento analógico con otro digital. La alternativa que se ha utilizado es la implementación de un sistema en el que se emula un modelo de un convertidor conmutado en hardware real. En este caso, el modelo se ha implementado en una FPGA Xilinx Zynq-7000 (XC7Z020).

A través del proceso de diseño que se ha explicado en los capítulos anteriores, se han obtenido varios tipos de modelos: un modelo *real*, un modelo en coma fija y un modelo en coma fija parametrizable. El modelo *real* no es sintetizable, pero es útil para comprobar de forma rápida si la extracción de ecuaciones realizada en los capítulos 3.1 y 5.1 es correcta. Por otra parte, el modelo en coma fija se considera un hito intermedio para el entendimiento del desarrollo del modelo en coma fija parametrizable. Esto es debido a que el modelo en coma fija no se puede adaptar a diferentes rangos de tensiones y corrientes, por lo que carece de utilidad real. La coma fija parametrizable soluciona esta limitación. Sin embargo, si la coma fija parametrizable se configura con el mismo número de decimales que el modelo en coma fija, sus resultados de precisión serían idénticos, por lo que no se mostrarán en este capítulo resultados de precisión para el modelo en coma fija. Por tanto, los resultados que se mostrarán a continuación se basan en los modelos de planta explicados en los capítulos 3.1 y 5.1.

A lo largo de este capítulo se expondrán los resultados finales que se buscaban durante la implementación de este proyecto. Además, se mostrarán las pruebas que se han llevado a cabo durante el desarrollo. Estas han servido para verificar el funcionamiento adecuado de la simulación del convertidor conmutado.

Existe un amplio abanico de posibilidades a la hora de elegir la herramienta a utilizar para la simulación de sistemas mixtos, que mezclan señales analógicas y digitales. En este caso, para la verificación de las simulaciones se ha empleado el simulador *Matlab Simulink 8.7*. Se trata de una herramienta muy extendida en el diseño de dispositivos electrónicos, ya que permite realizar simulaciones de forma fácil y precisa. Por tanto, todos los modelos escritos en VHDL se compararán con un modelo realizado en *Simulink* con y sin pérdidas eléctricas tal como muestra la Figura 20.

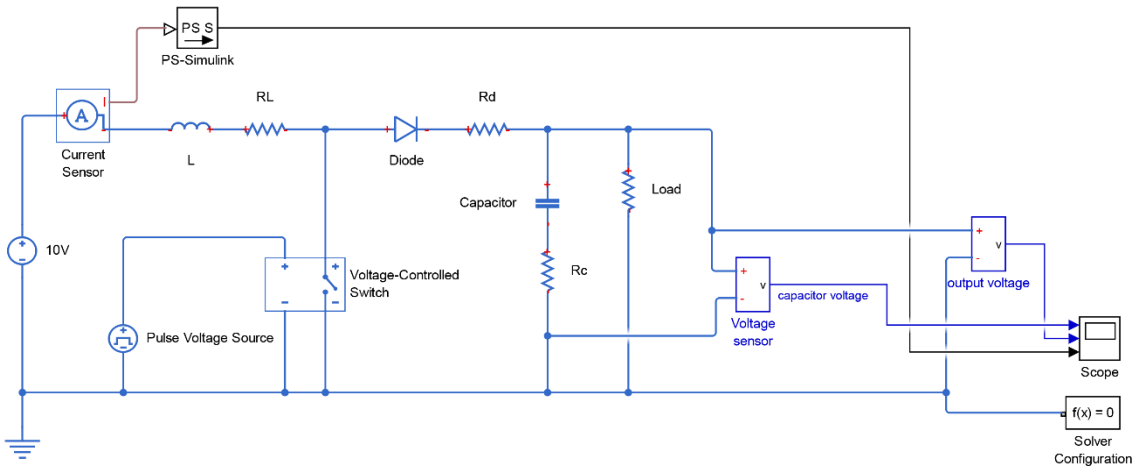


Figura 20. Esquemático de un convertidor *boost* en *Simulink.*

Por otra parte, para la extracción de resultados en los modelos escritos en VHDL se ha utilizado *ModelSim.* Por último, para la extracción de los recursos hardware y los tiempos de integración mínimos obtenidos por cada modelo, se ha utilizado la herramienta *Vivado 16.2*. Tanto las simulaciones en Simulink como en ModelSim han sido exportadas y comparadas usando Matlab.

Los resultados se pueden clasificar en tres tipos. Primero, se van a llevar a cabo varias simulaciones cuyo objetivo será la verificación de las ecuaciones descritas, así como la correcta discretización para el periodo de integración que posteriormente se propondrá. Para la realización de este proceso de verificación, se realizarán comparaciones entre las simulaciones llevadas a cabo en *Simulink* y en *ModelSim*. Durante esta etapa, también se podrá apreciar el efecto que tienen las pérdidas en la simulación del modelo.

En segundo lugar, se mostrarán los resultados de recursos que necesita nuestro sistema. Por último, y como continuación al desarrollo anterior, se obtendrán el tiempo de simulación y de emulación de nuestro modelo en coma fija parametrizable con pérdidas y sin ellas. Es decir, se obtendrá el periodo mínimo de discretización de nuestro modelo que marcará el tiempo de reloj de nuestra simulación. Cabe destacar que tanto el periodo de integración como el tiempo de reloj deben ser iguales para lograr que nuestra simulación pueda ir en tiempo real.

Antes de comenzar con la representación de resultados, es importante dejar claro el entorno de simulación que se ha utilizado, ya que todos los resultados se han utilizado para los mismos parámetros. Además, debe recordarse la Figura 2 en la que habrá que especificar los valores de cada componente del diseño. Los elementos pasivos del convertidor conmutado toman el valor de 1 µH la bobina y 0,1 µF el condensador. Además, debe establecerse la frecuencia a la que conmuta el interruptor (periodo de conmutación), que será 100 kHz. Por último, se fija la frecuencia de integración en 32,25 MHz, la tensión de entrada en 10 V y la tensión de salida en 20 V.

A continuación, se mostrará la Tabla 5 que hace referencia a los valores que se utilizarán en el modelo VHDL sin pérdidas.

Componente	Valor
Bobina (L)	1 µH
Condensador (C)	0,1 µF
Paso de integración (dt)	31 ns
Periodo de conmutación	10 µs
Tensión de entrada	10 V
Tensión de salida	20 V

Tabla 5. Valores empleados en el diseño del circuito sin pérdidas.

En todas las simulaciones realizadas en VHDL se va a utilizar un tiempo de integración de 31 ns. Ya que como se verá posteriormente en la sección de área y frecuencia, el mínimo paso de integración posible con coma fija parametrizable con pérdidas será el comentado.

No se debe olvidar que uno de los objetivos de este proyecto es realizar un estudio acerca del efecto de las pérdidas para dotar al sistema de mayor precisión. Debido a esto, en el capítulo anterior se han expuesto los diferentes elementos que causaban pérdidas de primer orden en nuestro modelo (se puede recordar la topología del circuito con la Figura 14). A continuación, se mostrará en la Tabla 6 los valores utilizados en nuestros componentes con pérdidas.

Componente	Valor
R_L	0,6965 Ω
R_S	0,3 Ω
R_D	0,01 Ω
R_C	0,3 Ω
V_D	1,3 V

Tabla 6. Valores empleados en el diseño del circuito con pérdidas.

6.1 Comparaciones

Antes de continuar, hay que dejar claro que todas las simulaciones se han llevado a cabo durante un tiempo total de 12 ms. Se ha escogido dicho tiempo de simulación ya que es un periodo suficiente para lograr ver en las figuras representadas el transitorio desde reposo hasta régimen permanente y su llegada a dicho régimen permanente.

Cabe destacar que las dos variables de estado de nuestro modelo, $v_C \; e \; i_L$, están interrelacionadas tal como se podía apreciar en las ecuaciones del circuito (5.1) mostradas en el capítulo 5. Y por tanto con el fin de mejorar la claridad del documento se van a mostrar únicamente la variable de estado de la tensión que cae en bornas del condensador v_C para cada una de las variables de nuestro diseño. No obstante, si se desea realizar una mayor profundización se adjunta en el Anexo A los resultados de las dos variables de estado para cada uno de los modelos.

Una vez definidos los valores de los componentes del convertidor conmutado, se va a proceder a realizar las simulaciones. En primer lugar, se va a llevar a cabo la simulación sin pérdidas en *Simulink* y en *ModelSim* mediante la notación real. Debe recordarse que este formato impide la conversión del modelo a hardware real, pero será muy útil para comprobar si se han calculado correctamente las ecuaciones y si el tiempo de integración utilizado (31 ns) es suficientemente pequeño para no cometer demasiado error en la discretización de las ecuaciones diferenciales vistas en el capítulo 3.1 La visualización de la Figura 21 permite la comparación de la variable de estado v_C .

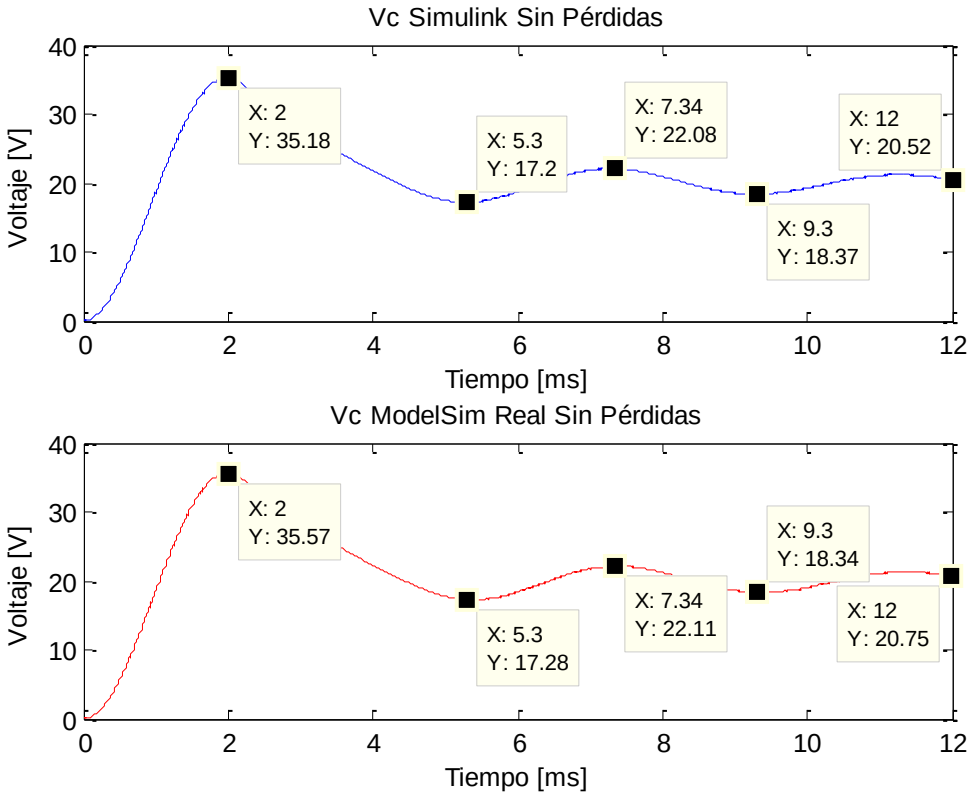


Figura 211. Simulación v_C sin pérdidas en *Simulink* y en formato *real.*

Tal y como se puede apreciar, apenas hay una ligera variación en los valores de ambas simulaciones. Uno de los inconvenientes que presenta el uso de *Simulink* es que no es posible obtener de manera fácil el valor en un punto exacto de la gráfica que coincida con una muestra tomada por *ModelSim*. Esto es debido a que las simulaciones realizadas por *ModelSim* utilizan otros métodos de discretización diferentes a Euler explícito con tiempo de integración constante. Debido a que su tiempo de integración es variable, no es fácil extraer de la simulación valores con una resolución temporal perfecta. En cualquier caso, se puede observar que el punto con mayor error (tiempo igual a 2 ms), el error es menor del 1%.

Gracias a estos resultados, se puede apreciar claramente cómo la variación entre las señales es mínima y, por tanto, se está llevando a cabo una discretización adecuada de las ecuaciones y una transformación correcta de las variables de estados a lenguaje VHDL.

Una vez se haya comprobado que el proceso anterior es correcto, se procede a realizar la siguiente comparación. En este caso, se va a demostrar uno de los objetivos de este proyecto, pues se da comienzo a una comparación del modelo *real* sin pérdidas y la simulación en *Simulink* de un convertidor elevador con pérdidas reales en los componentes. Para llevar a cabo está simulación, debe recordarse que en cada componente se han utilizados los valores descritos previamente en la Tabla 6. Al igual que con la figura anterior, se van a comparar la variable de nuestro sistema v_C , que podrá ser visualizada en la Figura 22.

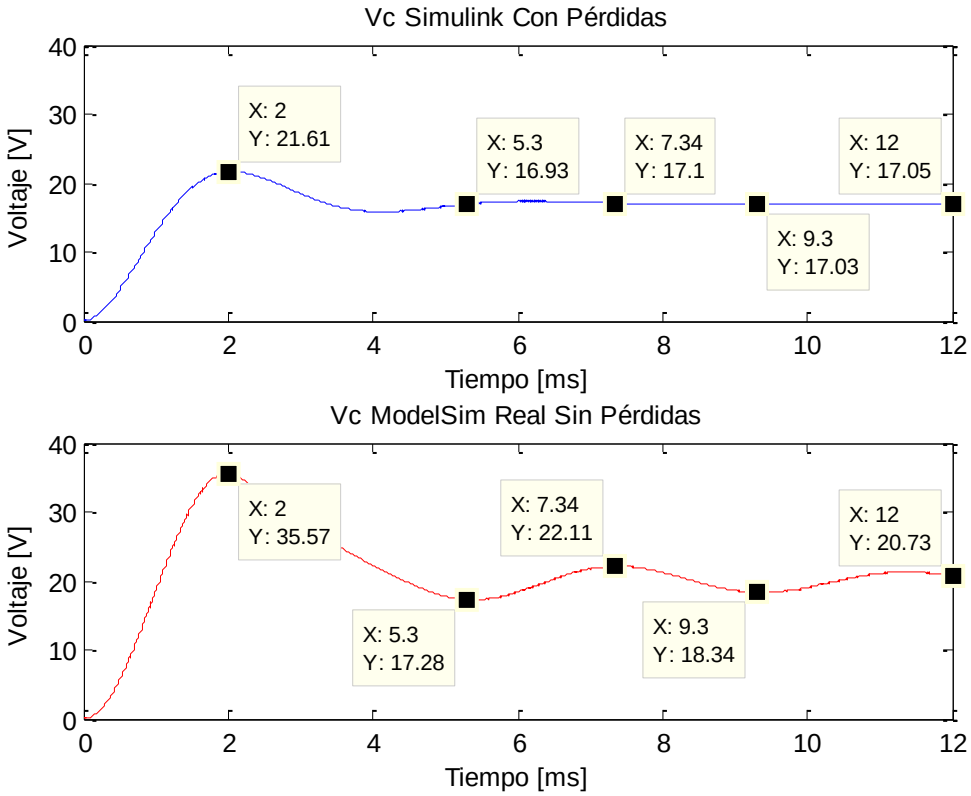


Figura 222. Simulación $\boldsymbol{v_C}$con pérdidas en *Simulink* y en formato *real* sin pérdidas.

Como bien se puede apreciar en la Figura 22, y como era de esperar, existen discrepancias notables entre los valores de las señales de los dos modelos a causa de las pérdidas eléctricas. Debido a la gran diferencia que existe entre los valores de ambas simulaciones, se llega a la conclusión de que las pérdidas en un convertidor conmutado son importantes y que, gracias al modelado de las mismas, será posible dotar al sistema de mayor precisión y similitud a comportamientos reales.

La siguiente comparación que se va a realizar tiene el mismo objetivo que la primera comparación realizada entre los modelos sin pérdidas que se mostraban en la Figura 20. Por tanto, se va a llevar a cabo la emulación del modelo *real* con pérdidas y del modelo en *Simulink* con pérdidas para verificar sea correcto que el proceso de discretización para el periodo de integración acordado. La comparación de ambos modelos queda representada en la siguiente Figura 23. Gracias a la comparación de ambos modelos se puede comprobar que se ha realizado correctamente todo el procedimiento de obtención del efecto de las pérdidas. En cualquier caso, se puede observar que el punto con mayor error (tiempo igual a 2 ms), el error es menor del 1%.

Una vez finalizadas estas tres representaciones, es posible concluir que en los diseños realizados no se están cometiendo errores debido a los tiempos de discretización y que el modelado de las ecuaciones de estado que rigen el comportamiento del sistema se ha obtenido correctamente.

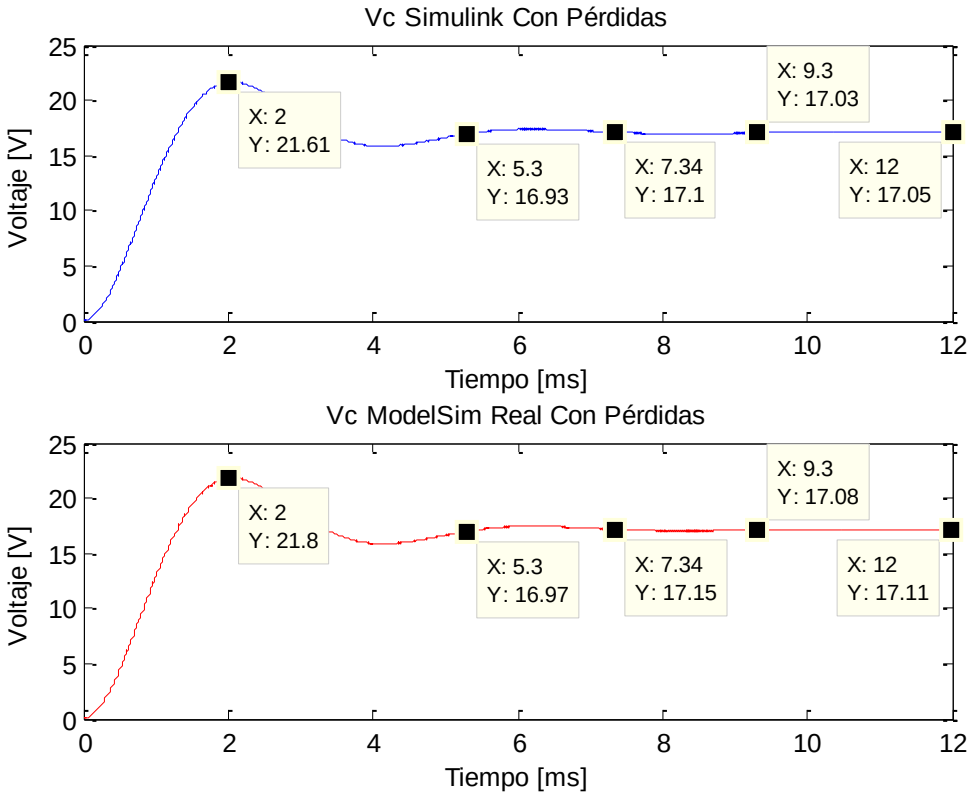


Figura 233. Simulación con pérdidas en *Simulink* y en formato *real* con pérdidas.

Por último, debe mencionarse que en las comparaciones de los modelos *real* con los de *Simulink* apenas se aprecia alguna variación de los resultados debido a que ambos sistemas se caracterizan por si gran precisión. En el caso de la aritmética *real*, el simulador utiliza coma flotante de doble precisión, con 53 bits de mantisa, por lo que su resolución es muy grande. Sin embargo, como ya se comentó anteriormente, el modelo *real* no es sintetizable por una FPGA y por tanto no se podrán realizar simulaciones en hardware a tiempo real. Por esta razón se ha implementado el modelo en coma fija parametrizable. Hay que recordar que se trata de un tipo de aritmética en donde la posición de la coma no es fija y por tanto es posible adaptar la simulación a las condiciones del usuario sin necesidad de resintetización de la planta. Pero, por otro lado, la complejidad de este modelo es mucho mayor debido a la necesidad de calcular correctamente los tamaños de las señales y las escalas para dicha adaptabilidad. También hay que destacar que, para realizar el diseño en coma fija parametrizable, se ha tenido en cuenta el tamaño de los recursos de la FPGA, por lo que se ha limitado el número de bits de los operandos de cada operación matemática. Esta limitación se ha realizado también intentando una pérdida mínima de precisión, pero es necesario compararlo con los resultados obtenidos con Simulink.

A pesar de haber llevado a cabo la implementación del modelo en coma fija parametrizable sin perdidas este no se va a representar ya que se considera un hito intermedio en el proyecto. Aunque con el modelo sin pérdidas se obtiene un diseño sintetizable y funcional, se ha observado en la Figura 21. que las pérdidas son un factor importante dentro del modelo. Por eso, se llevará a cabo la implementación del modelo en coma fija parametrizable con pérdidas, en el que, además de todas las características del modelo sin pérdidas, se dota al sistema de mayor precisión con la incorporación del efecto que producen. Para comprobar que el modelo presenta un comportamiento adecuado, a continuación, en la

Figura 24, se va a realizar una comparación entre el modelo emulado en *Simulink* con pérdidas y el modelo en *ModelSim* parametrizable con pérdidas.

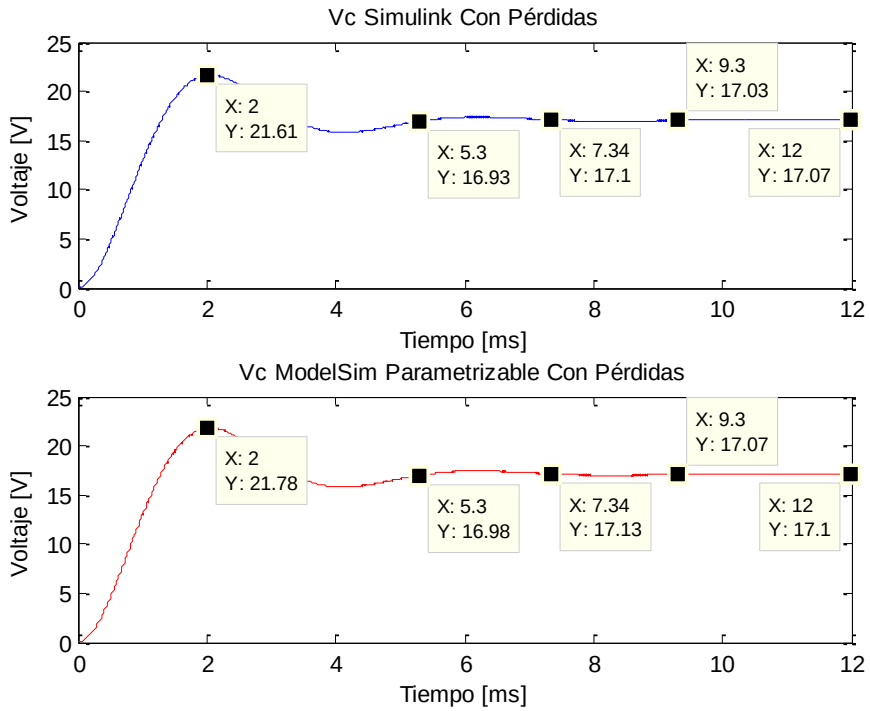


Figura 244. Simulación con pérdidas en *Simulink* y en formato parametrizable con pérdidas eléctricas.

Tal y como se puede apreciar en las anteriores gráficas, ambos modelos presentan diferencias mínimas en los valores, casi inapreciables a simple vista. En este caso, se puede observar que el punto con mayor error (tiempo igual a 2 ms), el error es menor del 1%.

Para finalizar se pueden extraer varias conclusiones de los resultados obtenidos. En primer lugar, con el modelo *real* se ha podido demostrar que las ecuaciones se han extraído correctamente y además de que el tiempo de 31 ns no genera apenas error en los resultados. Posteriormente, gracias a la comparativa de los modelos con y sin pérdidas hemos llegado a la conclusión que el efecto de las pérdidas es un factor importante a tener en cuenta ya que modifica en gran medida el resultado. Por último, con el fin de dotar al sistema de versatilidad frente a diferentes entornos de simulación ha quedado demostrado que el modelo parametrizable sigue resultando preciso, obteniendo errores mínimos.

6.2 Área y frecuencia

En el anterior apartado, se ha realizado un estudio acerca de la precisión numérica de los resultados. Sin embargo, existen otros factores que también hay que tener en cuenta además de los ya mencionados. Durante esta sección, se van a mostrar los resultados de área y frecuencia obtenidos en la síntesis de los modelos en la FPGA Xilinx Zynq-7000 (XC7Z020clg484-1). Para la obtención de los resultados se han utilizado los modelos que habían sido implementados anteriormente. Puesto que el modelo en *real* no permite hacer su transformación a hardware real por no ser sintetizable, en este apartado se mostrarán los resultados del modelo en coma fija parametrizable con y sin pérdidas.

Para la obtención de los resultados se ha utilizado una de las herramientas previamente mencionadas al principio del apartado: *Vivado 2016.2*. A través de este programa se realizará la implementación de los distintos modelos diseñados para obtener la frecuencia y los recursos empleados. Mediante la interfaz de esta herramienta, se podrá obtener el periodo de integración al que como mínimo debe ir el diseño, así como el número de recursos utilizados. Los recursos que se mostrarán gracias al programa serán el número de *Look up Table* (LUTs), el número de *Flip Flop* (FF) y el número de multiplicadores utilizados (DSP), además se mostrará para cada caso el periodo del convertidor elevador.

A continuación, se muestran en la Tabla 11 los valores obtenidos mediante *Vivado2016.2*. De este modo es posible apreciar la diferencia de los recursos empleados por la FPGA Xilinx Zynq-7000 XC7Z020 además del periodo de integración obtenido.

Modelo	Número de LUTs	Número de FF	Número de DSP	Periodo de reloj (ns)
Parametrizable sin pérdidas	415	109	2	17
Parametrizable con pérdidas	1305	109	6	31

Tabla 7. Recursos ocupados por la FPGA.

Se puede apreciar claramente el efecto que tiene la incorporación de las pérdidas al modelo gracias a la visualización de la tabla anterior. La incorporación al modelo de pérdidas implica incrementar en 214 % el número de LUTs, un 200 % el número de DSP y un aumento del periodo mínimo de integración en un 82,35 %. Este aumento de los recursos y del tiempo viene dado a que el modelo de pérdidas es mucho más complejo y por tanto implica más operaciones dentro del sistema. Por ejemplo, se puede ver que el modelo sin pérdidas usa dos multiplicadores ($v_C \cdot \frac{dt}{L}$ e $i_C \cdot \frac{dt}{C}$), mientras que el modelo con pérdidas añade 4 multiplicadores para el cálculo de las citadas pérdidas. Sin embargo, la degradación del periodo de reloj es asumible teniendo en cuenta la gran mejora en términos de precisión cuando se añaden las pérdidas eléctricas de primer orden.

Por último, es importante destacar que el periodo de integración es el que marca el tiempo del reloj del sistema, ya que no hay que olvidar que uno de los objetivos del proyecto es poder realizar la simulación en una FPGA a tiempo real. Es decir, cada 31 ns de ejecución de la FPGA (un ciclo de reloj), se estarán emulando 31 ns de tiempo real. En comparación con sistemas vistos en el estado del arte, se obtienen claras mejoras en el tiempo de integración, debido a que se obtienen decenas de ns como tiempo de integración frente a microsegundos con sistemas comerciales.

7 Conclusiones y trabajo futuro

7.1 Conclusiones

Antes de la producción de un regulador digital para un convertidor de potencia es necesario realizar pruebas exhaustivas para cerciorarse del correcto funcionamiento del mismo. Las comprobaciones experimentales pueden traer consigo accidentes que causen daños materiales o lo que es peor, personales. Por tanto, no cabe duda de la importancia que tiene la simulación de estos componentes.

El mayor de los problemas a la hora de realizar estos exámenes es la necesidad de usar simulaciones mixtas, ya que la planta es analógica mientras que el control es digital. Existen simuladores mixtos comerciales, pero los tiempos de simulación que ofrecen son muy lentos, además de su limitación en el número de lenguajes digitales soportados. Una de las alternativas utilizadas para acelerar el proceso de simulación es el empleo de la técnica HIL. En este proyecto, se ha diseñado un modelo HIL de un convertidor elevador utilizando una FPGA programada mediante lenguaje VHDL. Sobre el modelo de la planta definido mediante las ecuaciones diferenciales se han realizado tres modelos utilizando diferentes aritméticas: *real*, coma fija y coma fija parametrizable. Además, con el fin de dotar al sistema de mayor realismo, se ha aplicado el mismo procedimiento en el modelo de un convertidor elevador con pérdidas eléctricas de primer orden.

Todos los modelos se han comparado con un diseño realizado en *Matlab Simulink*. El primer modelo que se ha implementado es el modelo en *real*. Gracias a este diseño, se ha comprobado que el diseño del modelo matemático y la discretización con el paso de integración escogido (31 ns) son correctos. Los resultados muestran que el error en las variables de estado con el modelo VHDL era muy pequeño, tratándose, en el peor de los casos, de errores por debajo del 1 %. Este procedimiento también se ha implementado en el modelo con pérdidas donde, al igual que en el caso sin pérdidas, se obtienen errores por debajo del 1 %. Por último, en el modelo *real* se ha demostrado cómo el efecto de las pérdidas es de vital importancia a la hora de realizar la simulación del convertidor y que es un factor relevante a tener en cuenta. Puesto que este tipo de aritmética no permite su implementación dentro de una FPGA, se ha procedido a realizar dos modelos adicionales.

En segundo lugar, se ha llevado a cabo el diseño de un modelo en coma fija, que sí permite ser implementado en una FPGA, pero está limitado numéricamente en tiempo de diseño. Debido a esta limitación, este modelo se considera un hito intermedio para el entendimiento del desarrollo del modelo en coma fija parametrizable y, por tanto, no se han aportado resultados. Para la realización del último modelo del convertidor elevador, se ha empleado la notación en coma fija parametrizable. Este nuevo modelo dota al sistema de versatilidad, ya que es posible ajustar el diseño a diferentes condiciones de simulación sin necesidad de resintetizar el código. Si se hace un buen ajuste de la posición de la coma y del tamaño de las señales, la diferencia entre los valores de *Simulink* y de *Modelsim* es mínima, llegando a obtener un error por debajo del 1 % para el peor caso, pero obteniendo simulaciones en tiempo real.

Por otro lado, para poder comprobar que los modelos realizados se pueden ejecutar en tiempo real, se han implementado los modelos en una FPGA Xilinx Zynq-7000 y se han obtenido periodos de ejecución de hasta 31 ns.

Por lo tanto, en este Trabajo Final de Grado se ha demostrado que los métodos empleados son totalmente realizables en los procesos de verificación de convertidores conmutados, y en particular, para un convertidor elevador.

7.2 Trabajo Futuro

En este proyecto se ha llevado a cabo el desarrollo de un modelo parametrizable con pérdidas eléctricas de un convertidor de tipo elevador. Sin embargo, se puede continuar trabajando con este modelo y aportarle mejoras, por ejemplo, el diseño de una interfaz para el usuario o su integración con un periférico. Otro de los posibles puntos a desarrollar podría ser la realización de un estudio acerca del efecto del *pipeline* en algunas partes del diseño con el fin de lograr una aceleración global del sistema.

Además, se podría continuar con un estudio sobre el efecto que pueden ocasionar las pérdidas de segundo orden en el modelo, comprobando si es rentable la adición de estas a pesar de la complejidad de su realización y los retardos introducidos.

Por otro lado, sería provechoso ampliar la biblioteca de modelos parametrizables ejecutados para obtener un diseño de cada convertidor conmutado. De esta forma, sería posible la verificación de cualquier controlador para los convertidores más tradicionales.

Referencias

[1] A. de Castro, *Aplicación del Control Digital Basado en Hardware Específico para Convertidores de Potencia Conmutados*. PhD thesis, Universidad Politécnica de Madrid, 2003.

[2] Matlab, "www.mathworks.com", 2017.

[3] B. Patella, A. Prodic, A. Zirger, D. Maksimovic, *"High-frequency digital PWM controller IC for DC-DC converters", Power Electronics, IEEE Transactions on,* Vol 18, issue 1.

[4] A. Peterchev, J. Xiao, S. Sanders, *"Architecture and IC implementation of a digital VRM controller", Power Electronics, IEEE Transactions on*, vol 18, issue 1

[5] Maksimovic, Zane, Erickson, *"Impact of digital control in power electronics", in: 2004 Proceedings of the 16th International Symposium on Power Semiconductor Devices and ICs*, 2004.

[6] A. Prodic and D. Maksimovic, *"Mixed-signal simulation of digitally controlled switching converters", in Computers in Power Electronics, 2002. Proceedings. 2002 IEEE Workshop on*, pp. 100 105, Jun. 2002.

[7] Spectre, "www.cadence.com", 2017.

[8] Modelsim, "www.mentor.com", 2017.

[9] P. Zumel, C. Fernandez, A. Lazaro, and A. Barrado, *"Digital compensator design for dc-dc converters based on FPGA: an educational approach", in IEEE Indus-trial Electronics, IECON 2006 - 32nd Annual Conference on*, Nov. 2006.

[10] A. de Castro, T. Riesgo, O. Garcia, and R. Prieto, *"Comparing vhdl and vhdl-ams for modelling and simulation of power converters with digital control", in XVIII Conference on Design of Circuits and Integrated Systems (DCIS)*, Nov. 2003.

[11] Typhoom HIL, "www.typhoon-hil.com", 2017.

[12] OPAL-RT, "www.opal-rt.com", 2017

[13] dSPACE, "www.dspace.com", 2017

[14] Hypersim, "www.opal-rt.com/systems-hypersim", 2017

[15] Juan Carlos Martínez Quintero y Jaime Eduardo Andrade Ramírez, "Implementación de controladores en sistemas retroalimentados usando electrónica embebida y simulación hardware in the loop", Universidad Tecnológica de Pereira (Colombia), 2013.

[16] J.-Y. Dieulot, F. Colas, L. Chalal, G. Dauphin-Tanguy, "*Economic supervisory predictive control of a hybrid power generation plant, Electric Power Systems Research"* 127, 2015.

[17] S. Dennetire, H. Saad, B. Clerc, J. Mahseredjian, *"Setup and performances of the real-time simulation platform connected to the INELFE control system", Electric Power Systems Research 138, special Issue: Papers from the 11th International Conference on Power Systems Transients (IPST)* , 2016.

[18] O. Goni, A. Sanchez, E. Todorovich & A. de Castro, *"Resolution Analysis of Switching Converter Modelsfor Hardware-in-the-Loop", in IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1162-1170, May 2014.

[19] Sanchez, A. de Castro & J. Garrido, *"A comparison of simulation and hardware-in-the-loop alternatives for digital control of power converters", in IEEE Transactions on Industrial Informatics*, vol. 8, no. 3, pp. 491-500, Ago 2012.

[20] A. Sanchez, I. Villar, A. de Castro, F. López-Colino & J. Garrido, *"Hardware-in-the-loop using parametrizable fixed point notation", IEEE Workshop on Control and Modeling for Power Electronics, Trondheim*, Jun 2016.

Glosario

ADC (Analog-to-Digital Converter)

ALU (Unidad Aritmético Lógica)

AC (Alternating Current)

ADC (Analog-to-Digital Converter)

ADMS (Analog-Digital Mixed Signal)

CC (Corriente Continua)

CCM (Continuous Conduction Mode)

DAC (Digital to Analog Converter)

DC (Direct Current)

DCM (Discontinuous Conduction Mode)

DSP (Digital Signal Processor)

FF (Flip Flop)

FPGA (Field-Programmable Gate Array)

HDL (Hardware Description Language)

HIL (Hardware-In-the-Loop)

IGBT (Insulated Gate Bipolar Transistor)

MOSFET (Metal-oxide-semiconductor Field-effect transistor)

MSB (Most Significant Bit)

PWM (Pulse-Width Modulation)

VHDL (VHSIC Hardware Description Language)

Anexos

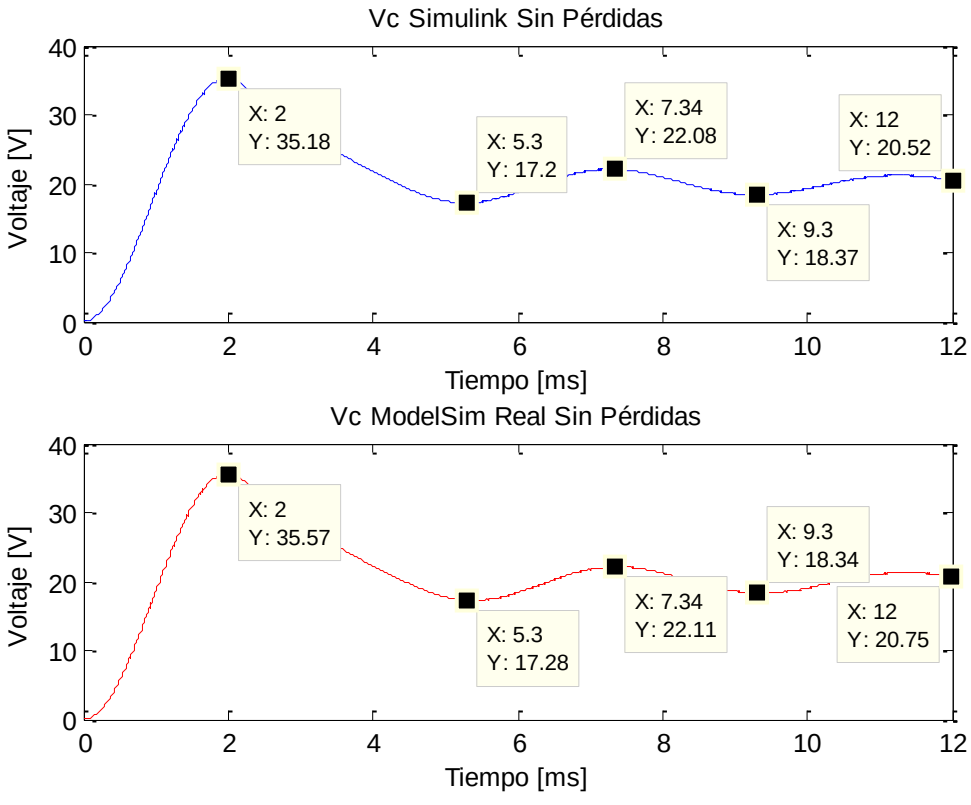
A Anexo de resultados obtenidos en simulación

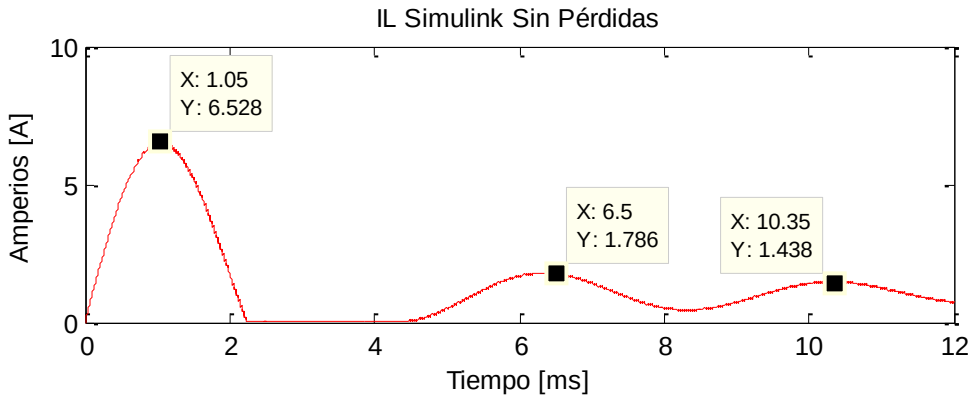
Durante este anexo van a ser representados los valores obtenidos en los diferentes modelos realizados en este proyecto. Hay que recordar que para la obtención de estos resultados en VHDL ha sido necesario definir los valores de nuestro modelo. A continuación, se mostrará las Tabla 5 y 6 que hace referencia a los valores que se utilizarán en el modelo VHDL con y sin pérdidas.

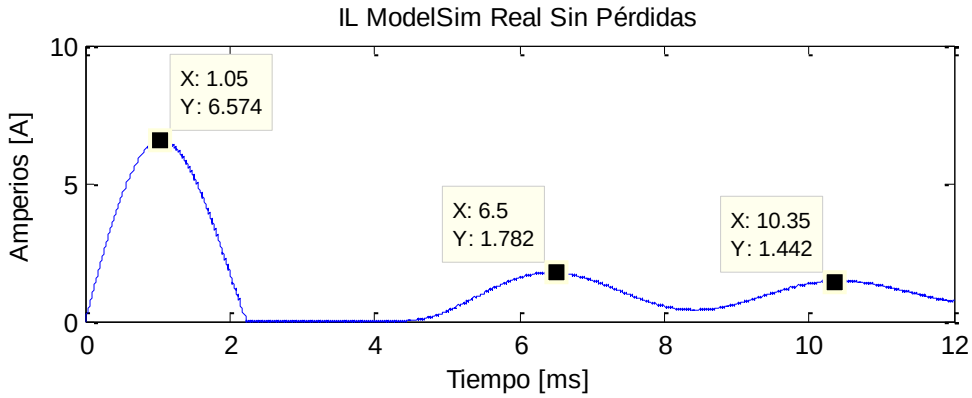
Componente	Valor
Bobina (L)	1 µH
Condensador (C)	0,1 µF
Paso de integración (dt)	31 ns
Periodo de conmutación	10 µs
Tensión de entrada	10 V
Tensión de salida	20 V

Componentes pérdidas	Valor
R_L	0,6965 Ω
R_S	0,3 Ω
R_D	0,01 Ω
R_C	0,3 Ω
V_D	1,3 V

A.1 Comparación del modelo Simulink *sin pérdidas y* ModelSim real *sin pérdidas*







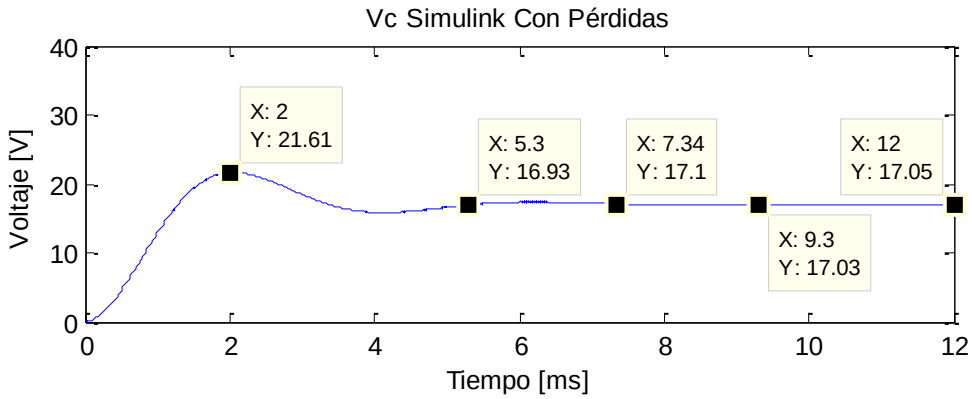
Valores v_C

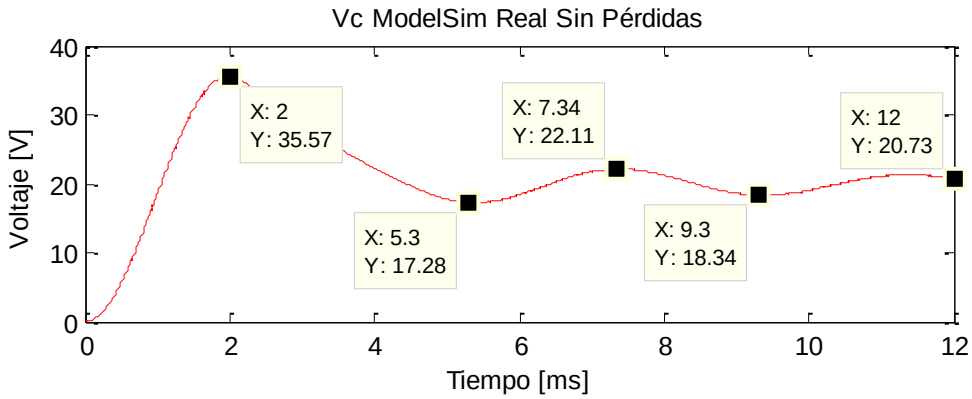
	2 ms	5,3 ms	7,34 ms	9,3 ms	12 ms
Simulink sin pérdidas	35,18 V	17,2 V	22,08 V	18,37 V	20,52 V
Modelo *Real* sin pérdidas	35,57 V	17,28 V	22,1 V	18,34 V	20,77 V

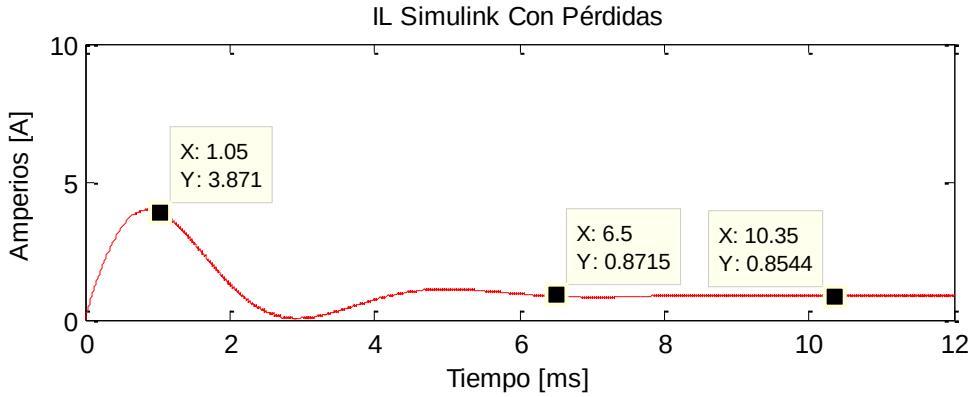
Valores i_L

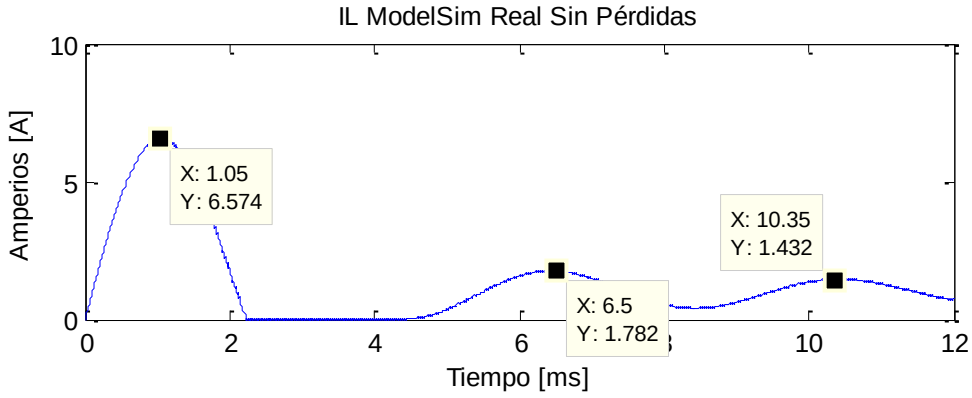
	1,05 ms	6,5 ms	10,35 ms
Simulink sin pérdidas	6,52 A	1,78 A	1,43 A
Modelo *Real* sin pérdidas	6,57 A	1,78 A	1,44 A

A.2 Comparación del modelo Simulink *con pérdidas y* ModelSim real *sin pérdidas*









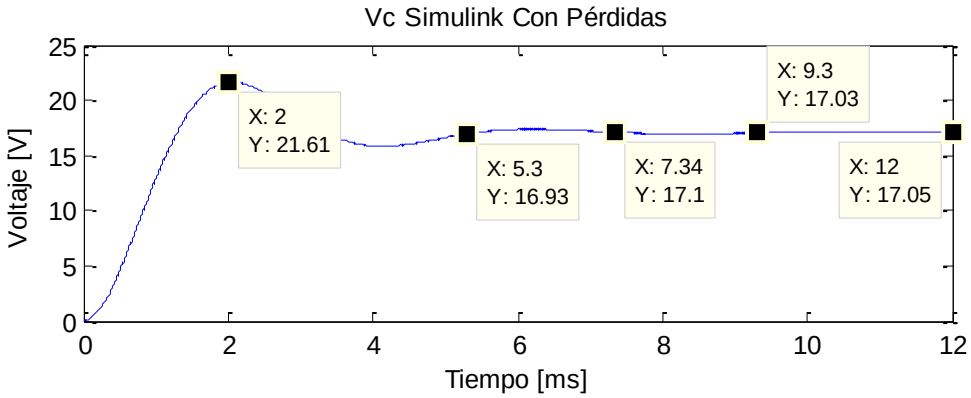
Valores v_C

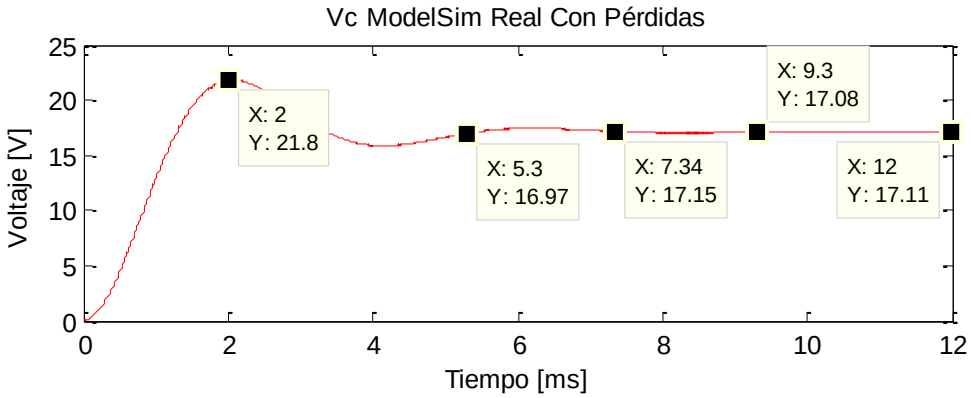
	2 ms	5,3 ms	7,34 ms	9,3 ms	12 ms
Simulink con pérdidas	21,61 V	16,93 V	17,10 V	17,1 V	17,05 V
Modelo *Real* sin pérdidas	35,57 V	17,28 V	22,1 V	18,34 V	20,73 V

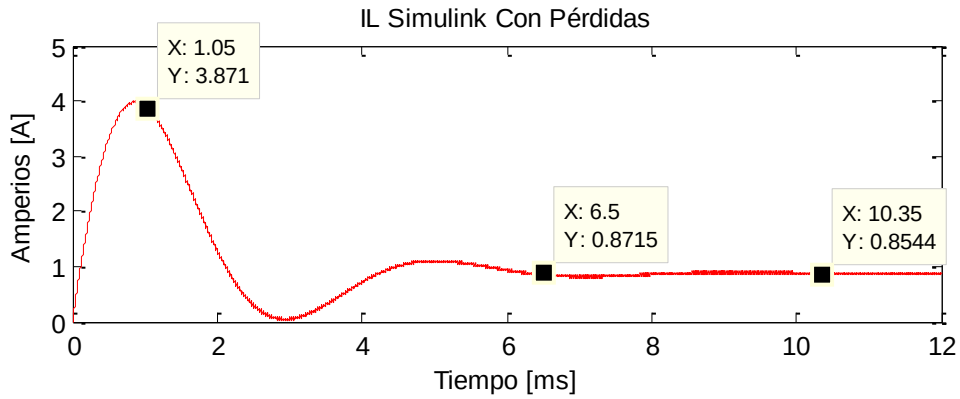
Valores i_L

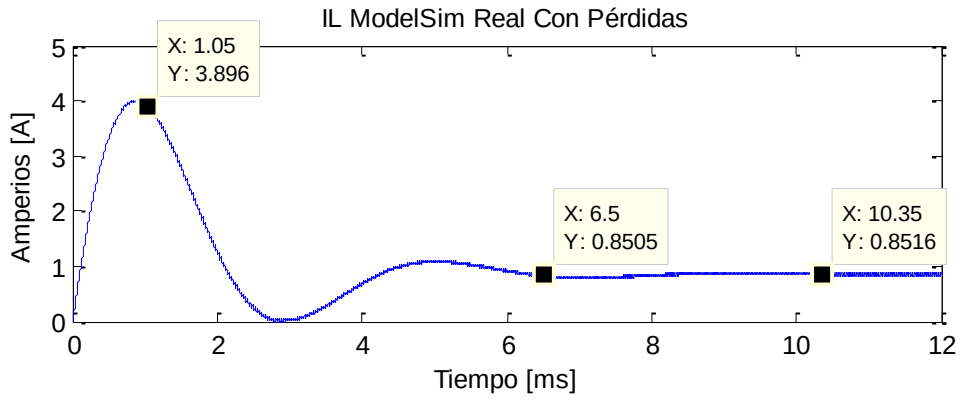
	1,05 ms	6,5 ms	10,35 ms
Simulink con pérdidas	3,87 A	0,87 A	0,85 A
Modelo *Real* sin pérdidas	6,57 A	1,78 A	1,43 A

A.3 Comparación del modelo Simulink *con pérdidas y* ModelSim real *con pérdidas*









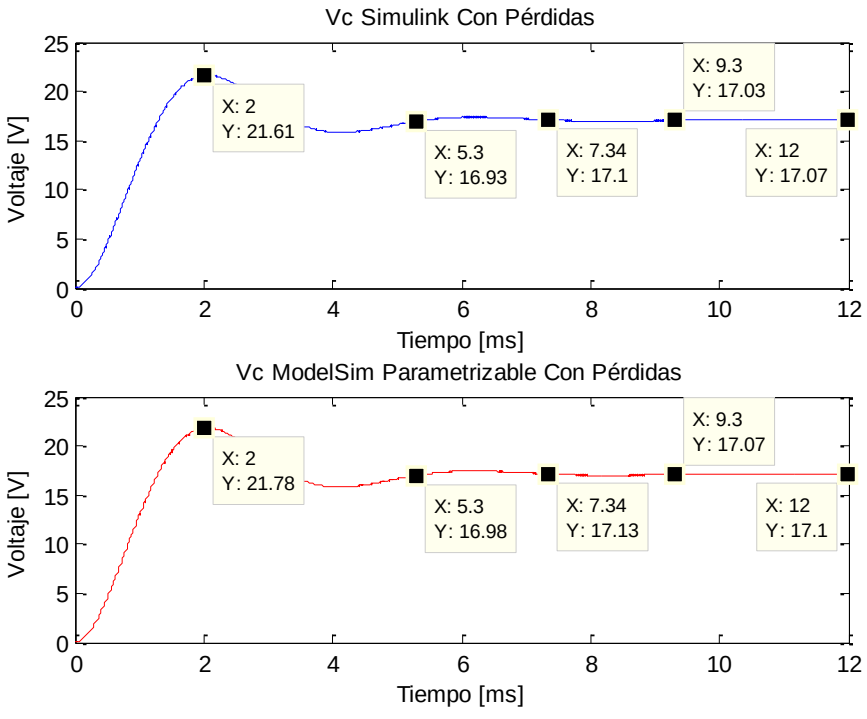
Valores v_C

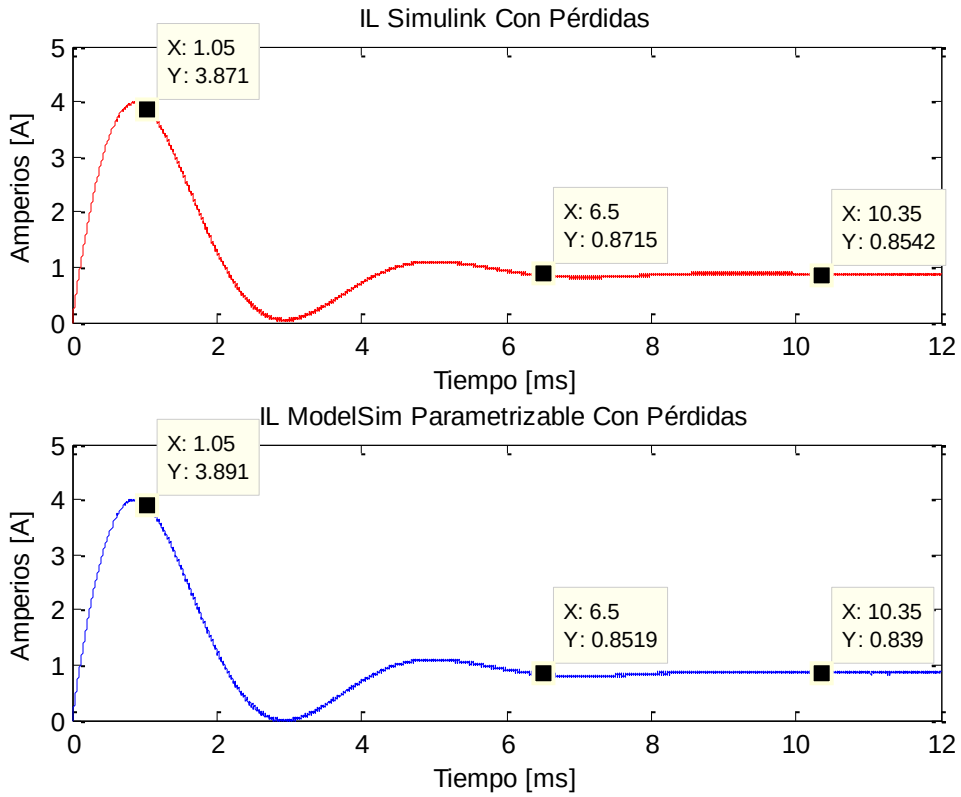
	2 ms	5,3 ms	7,34 ms	9,3 ms	12 ms
Simulink con pérdidas	21,61 V	16,93 V	17,1 V	17,1 V	17,05 V
Modelo *Real* con pérdidas	21,8 V	16,97 V	17,15 V	17,15 V	17,11 V

Valores i_L

	1,05 ms	6,5 ms	10,35 ms
Simulink con pérdidas	3,87 A	0,87 A	0,85 A
Modelo *Real* con pérdidas	3,89 A	0,85 A	0,85 A

A.4 Comparación del modelo Simulink *y* ModelSim *parametrizable con pérdidas*





Valores v_C

	2 ms	5,3 ms	7,34 ms	9,3 ms	12 ms
Simulink con pérdidas	21,61 V	16,93 V	17,1 V	17,03 V	17,05 V
Modelo parametrizable con pérdidas	21,78 V	16,98 V	17,13 V	17,07 V	17,1 V

Valores i_L

	1,05 ms	6,5 ms	10,35 ms
Simulink sin pérdidas	3,87 A	0,87 A	0,85 A
Modelo *Real* sin pérdidas	3,89 A	0,85 A	0,83 A