



**UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL**
FACULTAD DE EDUCACIÓN TÉCNICA
PARA EL DESARROLLO
CARRERA DE INGENIERÍA EN ELÉCTRICO MECÁNICA

TEMA:

**Implementación de aplicaciones prácticas en electrónica de
potencia utilizando los entornos de simulación Xilinx y
Simulink**

AUTOR:

Vega Freire, Hugo Miguel

Componente práctico del examen complejo previo a la
Obtención del grado de **INGENIERO EN ELÉCTRICO
MECÁNICA**

REVISOR:

M. Sc. Romero Paz, Manuel de Jesús

Guayaquil, Ecuador

20 de septiembre del 2021




**UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL**
FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO
CARRERA DE INGENIERÍA EN ELÉCTRICO MECÁNICA

CERTIFICACIÓN


Certificamos que el presente **componente práctico del examen complejo**, fue realizado en su totalidad por **Vega Freire, Hugo Miguel** como requerimiento para la obtención del título de **INGENIERO EN ELÉCTRICO MECÁNICA**.

REVISOR



M. Sc. Romero Paz, Manuel de Jesús

DIRECTOR DE CARRERA



M. Sc. Heras Sánchez, Miguel Armando

Guayaquil, a los 20 días del mes de septiembre del año 2021



**UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL**
FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO
CARRERA DE INGENIERÍA EN ELÉCTRICO MECÁNICA

DECLARACIÓN DE RESPONSABILIDAD

Yo, **Vega Freire, Hugo Miguel**

DECLARÓ QUE:

El **componente práctico del examen complejo, Implementación de aplicaciones prácticas en electrónica de potencia utilizando los entornos de simulación Xilinx y Simulink** previo a la obtención del Título de **Ingeniero en Eléctrico Mecánica**, ha sido desarrollado respetando derechos intelectuales de terceros conforme las citas que constan en el documento, cuyas fuentes se incorporan en las referencias o bibliografías. Consecuentemente este trabajo es de mi total autoría.

En virtud de esta declaración, me responsabilizo del contenido, veracidad y alcance del Trabajo de Titulación referido.

Guayaquil, a los 20 días del mes de septiembre del año 2021

EL AUTOR

Hugo Vega

VEGA FREIRE, HUGO MIGUEL



**UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL**
FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO
CARRERA DE INGENIERÍA EN ELÉCTRICO MECÁNICA

AUTORIZACIÓN

Yo, **Vega Freire, Hugo Miguel**

Autorizó a la Universidad Católica de Santiago de Guayaquil a la **publicación** en la biblioteca de la institución del **componente práctico del examen complejo, Implementación de aplicaciones prácticas en electrónica de potencia utilizando los entornos de simulación Xilinx y Simulink**, cuyo contenido, ideas y criterios son de mi exclusiva responsabilidad y total autoría.

Guayaquil, a los 20 días del mes de septiembre del año 2021

EL AUTOR

Hugo Vega

VEGA FREIRE, HUGO MIGUEL

REPORTE DE URKUND

URKUND Abrir sesión

Documento	Vega_EC_A2021.docx (D112384200)
Presentado	2021-09-10 16:55 (-04:00)
Presentado por	fermandopm23@hotmail.com
Recibido	edwin.palacios.ucsg@analysis.urkund.com
Mensaje	Revisión EC Hugo Vega Mostrar el mensaje completo 1% de estas 16 páginas, se componen de texto presente en 1 fuentes.

Lista de fuentes	Bloques
Categoría	Enlace/nombre de archivo
	TT-Miguel-Flores-31-08-17.SDA.doc
	TRABAJO DE TITULACION AVANCE 4.docx
	TT Chavez W.docx
	http://201.159.223.180/bitstream/3317/15533/1/T...
	http://repositorio.ucsg.edu.ec/bitstream/3317/15...
	http://repositorio.ucsg.edu.ec/bitstream/3317/11...

1 Advertencias. Reiniciar Exportar Compartir

UNIVERSIDAD CATÓLICA DE SANTIAGO DE GUAYAQUIL FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO CARRERA DE INGENIERÍA EN ELÉCTRICO MECÁNICA

TEMA:
Implementación de aplicaciones prácticas en electrónica de potencia utilizando los entornos de simulación Xilinx y Simulink

AUTOR: Vega Freire, Hugo Miguel

Componente práctico del examen complejoivo previo a la obtención del grado de INGENIERO EN ELÉCTRICO MECÁNICA



DEDICATORIA

Mi proyecto de titulación lo dedico especialmente a mis queridos abuelos María Inés Valencia Villacis y José Miguel Freire Escobar quienes desde muy temprana edad nunca me dejaron solo, siempre recibí su amor, y apoyo incondicional tanto moral como económico para que yo pudiera culminar mi carrera.

A mi madre Daysy Freire por ser mi fuente de motivación e inspiración y siempre estar a mi lado dándome la fortaleza para seguir adelante, y por supuesto no podía dejarlos de mencionarlos a mis queridos hermanos a quienes los quiero mucho, quienes con sus palabras de aliento me dieron esa fortaleza para seguir adelante y cumplir con mis ideales.

A mis padres, que sin ellos esto no hubiera sido posible.

EL AUTOR

VEGA FREIRE, HUGO MIGUEL

AGRADECIMIENTO

El agradecimiento de este proyecto es para el forjador de mi camino mi padre Victor Vega por acompañarme siempre y darme la oportunidad de lograr uno de mis objetivos que es ser un profesional, al grupo selecto de maestros que tuve el privilegio de tenerlos en cada año de mis estudios transmitiéndome sus conocimientos para poder ser un excelente profesional.

No fue sencillo este caminar, pero gracias a mi familia, amigos y a todas aquellas personas que durante estos cinco años estuvieron a mi lado apoyándome he llegado a la meta.

Les agradezco y les hago presente mi afecto a mi linda familia, queridos amigos, y estimados maestros.

EL AUTOR

VEGA FREIRE, HUGO MIGUEL



**UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL**
FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO
CARRERA DE INGENIERÍA EN ELÉCTRICO MECÁNICA

TRIBUNAL DE SUSTENTACIÓN

f. 

M. Sc. ROMERO PAZ, MANUEL DE JESUS
DECANO

f. 

M. Sc. PALACIOS MELÉNDEZ, EDWIN FERNANDO
COORDINADOR DEL ÁREA

f. 

M. Sc. CÓRDOVA RIVADENEIRA, LUIS SILVIO
OPONENTE

ÍNDICE GENERAL

Índice de Figuras	XI
Índice de Tablas.....	XIII
Resumen	XIV
Capítulo 1: Descripción del componente práctico	2
1.1. Introducción.....	2
1.2. Objetivo General.	3
1.3. Objetivos Específicos.	3
Capítulo 2: Fundamentación teórica.	4
2.1. Matlab.....	4
2.2. Entorno de desarrollo para simulación de hardware en bucle (HIL).....	7
2.3. Entorno de desarrollo para simulación MatLab/Simulink.....	7
2.4. Software System Generator de Xilinx.....	9
2.3.1. Conjuntos de bloques de Xilinx en Simulink.....	10
2.3.2. Tipos de señales.	11
2.3.3. Proyectos Multifase.	13
2.3.4. Métodos de compilación.....	14
Capítulo 3: Desarrollo del componente práctico.	16
3.1. Descripción general de las aplicaciones prácticas.	16
3.2. Aplicación práctica 1: Simulación de filtro de paso bajo.....	16
3.3. Aplicación práctica 2: Simulación de filtro de paso alto.....	21
3.4. Aplicación práctica 3: Simulación de inversor de medio puente.....	24
3.5. Aplicación práctica 4: Simulación de inversor de puente.	30
3.6. Aplicación práctica 5: Simulación de Control de retroalimentación de un inversor de puente.....	33

Conclusiones.	37
Recomendaciones	38
Bibliografía	39

Índice de Figuras

Capítulo 2:

Figura 2. 1: Interfaz del software de MatLab.....	6
Figura 2. 2: Apertura de 19 ventanas de Matlab.....	8
Figura 2. 3: Conjuntos de bloques de System Generator en Simulink.....	10
Figura 2. 4: Librería del conjunto de bloques de Xilinx.	11
Figura 2. 5: Aplicación de System Generator de Xilinx en Simulink.....	13
Figura 2. 6: Generar esquema de compilación en lenguaje de bajo nivel....	15

Capítulo 3:

Figura 3. 1: Circuito esquemático de un filtro pasa bajo RC.	17
Figura 3. 2: Diagrama de Bode real (rojo) y asintótico (azul).	18
Figura 3. 3: Diagrama de bloques del filtro de tiempo discreto.	20
Figura 3. 4: La senoide azul es la señal filtrada en tiempo continuo, la púrpura es la discretizada.	21
Figura 3. 5: Circuito esquemático de un filtro pasa alto RC.	22
Figura 3. 6: Diagrama de Bode real (rojo) y asintótico (azul).	22
Figura 3. 7: Diagrama de bloques del filtro pasa alto discretizado.....	24
Figura 3. 8: La senoide verde es la señal filtrada en tiempo continuo, la roja es discretizada.	24
Figura 3. 9: Circuito esquemático del inversor de medio puente con carga óhmica-inductiva.	25
Figura 3. 10: Diagrama de bloques del inversor de medio puente.....	27
Figura 3. 11: Representación del diagrama de bloques de Simulink de la señal moduladora.	28
Figura 3. 12: Diagrama de bloques de la carga en System Generator de Xilinx.	29
Figura 3. 13: Diagrama de corriente de salida para inversores de medio puente.....	30
Figura 3. 14: Circuito esquemático de inversor de puente.....	31
Figura 3. 15: Diagrama de bloques del inversor puente.	32
Figura 3. 16: Representación del diagrama de bloques de Simulink de la señal moduladora.	32

Figura 3. 17: Representación del diagrama de bloques en Simulink.	32
Figura 3. 18: Diagrama de corriente de salida para inversores de puente...	33
Figura 3. 19: Esquema teórico del modelo a crear en Simulink.	34
Figura 3. 20: Diseño del filtro FIR paso de banda usando Simulink.....	35
Figura 3. 21: Controlador PI.....	35
Figura 3. 22: Carga L-R.	35
Figura 3. 23: Señal de salida discreta.....	36

Índice de Tablas

Capítulo 2:

Tabla 3. 1: Tipos de solucionadores de ODE explícitos de pasos variables.. 9

Tabla 3. 2: Tipos de solucionadores de ODE explícitos de pasos fijos..... 9

Resumen

En el presente documento se desarrolla el componente práctico del examen complejo denominado “Implementación de aplicaciones prácticas en electrónica de potencia utilizando los entornos de simulación Xilinx y Simulink”. En el campo de la electrónica, los sistemas eléctricos y demás, durante la fase de creación de prototipos de un sistema, las simulaciones son necesarias para depurar el código creado y probar los resultados obtenidos como si el sistema funcionara en un entorno real. En este componente práctico se trata de la simulación en tiempo real, siendo uno de los métodos más utilizados en los sistemas HIL y cómo se implementa en la electrónica de potencia. Se estudiaron tres modelos de simulación en tiempo real: continuo, discreto e híbrido. El entorno Xilinx System Generator, es un entorno de desarrollo que permite el diseño de modelos para simulaciones híbridas en tiempo real. Se intentará comprender cómo está estructurado y se darán algunos ejemplos en el campo de la electrónica de potencia.

Palabras claves: Potencia, Electrónica, Simulación, Discreto, Semiconductores, Inversores

Capítulo 1: Descripción del componente práctico

1.1. Introducción.

Los sistemas electrónicos de potencia se utilizan ampliamente en la actualidad para proporcionar procesamiento de potencia para aplicaciones que van de la informática y las comunicaciones hasta la electrónica médica, el control de aparatos, el transporte y la transmisión de alta potencia. Los niveles de potencia asociados van desde milivatios hasta megavatios. Estos sistemas generalmente involucran circuitos de conmutación compuestos por interruptores semiconductores como tiristores, MOSFET y diodos, junto con elementos pasivos como inductores, condensadores y resistencias, y circuitos integrados para control. El análisis y diseño de tales sistemas presenta importantes desafíos.

El modelado y la simulación son ingredientes esenciales del proceso de análisis y diseño en la electrónica de potencia. Ayudan al ingeniero de diseño a comprender mejor el funcionamiento del circuito. Con este conocimiento, el diseñador puede, para un conjunto dado de especificaciones, elegir una topología, seleccionar los tipos y valores apropiados de los componentes del circuito, estimar el rendimiento del circuito y completar el diseño asegurándose, utilizando la simulación de Monte Carlo, el análisis del peor de los casos y otros análisis de fiabilidad y rendimiento de producción: que el rendimiento del circuito cumplirá con las especificaciones incluso con las variaciones previstas en las condiciones de funcionamiento y los valores de los componentes del circuito.

La mayor disponibilidad de computación potente ha hecho que la simulación directa sea ampliamente accesible y ha ampliado el conjunto de enfoques de análisis y modelado manejables. La simulación de un esquema de producción completo sigue siendo un objetivo difícil de alcanzar; Los obstáculos incluyen la necesidad de una construcción extensa de modelos, tiempos de simulación excesivamente largos, los desafíos de reconocer y explotar automáticamente la estructura modular o jerárquica o de escala de

tiempo, las dificultades de acoplar diversas modalidades de modelado y simulación, y los efectos del diseño, empaquetado y parásitos.

La mayoría de los sistemas de energía eléctrica en la actualidad se están volviendo electrónicos, ya que esto ofrece un alto potencial de ahorro de costos durante el ciclo de vida, una gran mejora en la eficiencia del sistema, alta densidad, regulación de voltaje, confiabilidad, tamaño más pequeño y peso más liviano con un crecimiento continuo de la complejidad del sistema.

1.2. Definición del problema.

La carrera de Ingeniería Eléctrico-Mecánica ha pasado por varias modificaciones en sus mallas de estudios, y en el año 2020 se armoniza la titulación como Ingeniería en Electricidad. La malla rediseñada en el año 2020 consta como asignatura Electrónica de Potencia y en base a esto, surge la necesidad de proponer aplicaciones prácticas de electrónica de potencia para el aprendizaje experimental de los estudiantes utilizando las herramientas de simulación Xilinx y MatLab/Simulink.

1.3. Objetivo General.

Realiza la implementación de aplicaciones prácticas en electrónica de potencia utilizando los entornos de simulación Xilinx y Simulink

1.4. Objetivos Específicos.

- a. Describir los fundamentos teóricos de los entornos de simulación a nivel de hardware.
- b. Diseñar las aplicaciones prácticas de electrónica de potencia utilizando los entornos de simulación Xilinx y Simulink.
- c. Evaluar los resultados obtenidos de las prácticas propuestas utilizando los entornos de simulación Xilinx y Simulink.

Capítulo 2: Fundamentación teórica.

En este capítulo se desarrolla la parte teórica de la electrónica de potencia y de la simulación de hardware en bucle aplicado a la electrónica de potencia.

2.1. Introducción a la electrónica de potencia.

La electrónica de potencia se encuentra en la encrucijada entre la electrónica (bajas corrientes) y la ingeniería eléctrica (alta corriente). Es el desarrollo de semiconductores de potencia (diodos, tiristores y transistores ... etc.). La electrónica de potencia sustituye las transformaciones de energía que antes realizaban grupos de máquinas rotativas, tales como: (Fuente Alba, 2011)

1. El diodo rectificador es equivalente al conmutador.
2. Los rectificadores controlados son análogos a un grupo de motores generadores de corriente continua asíncronos.
3. Los atenuadores reemplazan a los autotransformadores de derivación variable o de control deslizante
4. El inversor autónomo fue producido por el conjunto de motor DC acoplado a un alternador.
5. El circuito chopper (conocido como convertidores de DC a DC) funciona a baja potencia con relés vibradores, ahora puede funcionar a muy alta potencia con tiristores.

La electrónica de potencia casi siempre trabaja en conmutación, esta es una de sus principales características, y esto permite obtener una buena salida. Las señales de control son hoy generadas por sistemas de microcomputadoras en lógica programada que reemplazan cada vez más al control analógico realizado por los amplificadores operacionales. (Rodríguez Molina, 2018)

En este trabajo del componente práctico del examen complejo se intentará modelar determinadas aplicaciones básicas que deberán ser estudiadas en el curso de Sistemas Electrónicos de Potencia mediante

circuitos equivalentes. Este modelado del modo de conducción continua debería permitir calcular los voltajes y corrientes promedio para nodo del circuito. Además, es fundamental definir las pérdidas y consecuentemente la eficiencia del sistema para un determinado punto de funcionamiento (aunque esta parte no se considera en el presente trabajo).

Por ejemplo, para facilitar el análisis de un circuito, es una buena idea ignorar las ondulaciones de corriente en los inductores y las ondulaciones de voltaje en los terminales de los condensadores y solo tener en cuenta la componente directa (DC) de las señales.

La electrónica de potencia solo es posible a través del componente electrónico, creador de señal o transmisor. Así es como se explican en el capítulo 3 en las cinco aplicaciones práctica simuladas en tiempo real mediante los entornos de simulación Xilinx y MatLab/Simulink. (Aguirre D., 2013)

2.2. Componentes semiconductores de la electrónica de potencia.

Entonces, los componentes serán las interfaces o “drivers” que conectan la pequeña potencia de las señales de control con la alta potencia obtenida al conectar los transistores MOS o IGBT o bien los triacs y, por supuesto, los tiristores (González, 2015). El enfoque principal del presente componente práctico es implementar cinco aplicaciones prácticas de sistemas electrónicos de potencia para la parte experimental de la asignatura.

La electrónica de potencia utiliza componentes semiconductores para realizar funciones de conmutación (interruptores) encargadas de adaptar los voltajes y corrientes de una red de distribución para satisfacer las necesidades de la carga a suministrar. Los interruptores no controlados están hechos por los diodos de potencia. Cuando las aplicaciones requieren una intervención externa, entran en juego componentes controlables: entre estos, se estudian los tiristores y transistores de potencia bipolares o MOS, y transistor IGBT más moderno. (Doménech A. et al., 2019)

En el caso de los rectificadores no controlados (diodos semiconductores) la entrada es un voltaje alterno monofásico. Los rectificadores con diodos proporcionan voltajes DC constantes, los que tienen tiristores u otro componente controlado dan voltajes DC de valores promedio variables.

En el caso de los inversores, se desarrollan tres aplicaciones prácticas, dos en las que se implementa inversores de medio puente y puente, y otra en la que se utiliza la modulación por ancho de pulso (PWM), su control y sus efectos. Las condiciones transitorias juegan un papel importante en el comportamiento de los convertidores.

2.3. Visión general de Hardware en bucle (HIL).

Hardware en bucle (*Hardware In the Loop, HIL*) o pruebas de simulación de hardware en bucle (*Hardware In the Loop Simulation Testing, HILST*) es, en cierto sentido, lo opuesto a la creación de prototipos de control rápido (*Rapid Control Prototyping, RCP*). Las leyes de control se implementan en el hardware final mientras que el convertidor de potencia, la planta y la retroalimentación se simulan los sensores. Al igual que con RCP, la simulación debe ejecutarse en tiempo real porque parte del sistema (aquí, el controlador) es hardware físico. Además, debe haber hardware para interactuar con la salida de la ley de control y las señales del sensor de retroalimentación. Esta configuración se muestra en la Figura 2.1. (Ellis, 2012)

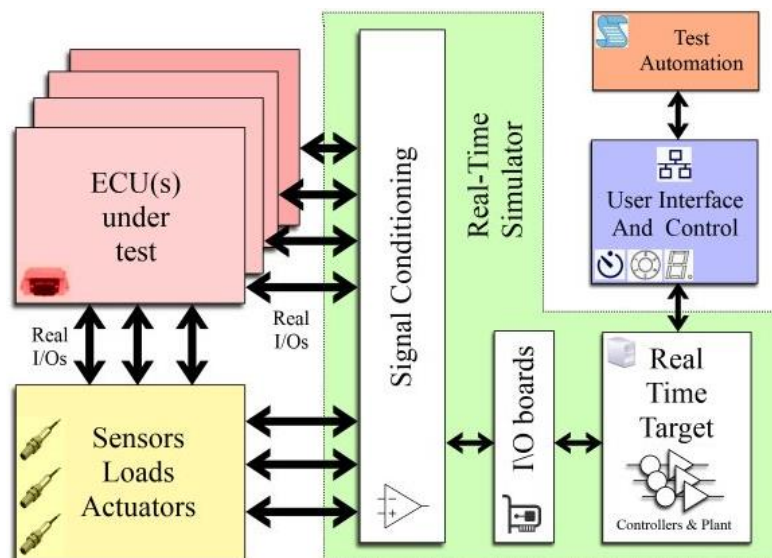


Figura 2. 1: Interfaz del software de MatLab.
Fuente: (The MathWorks, Inc, 2016)

Las pruebas de hardware en el bucle (HIL) son una metodología de prueba que se puede utilizar durante el desarrollo de controladores integrados en tiempo real para reducir el tiempo de desarrollo y mejorar la eficacia de las pruebas.

2.4. Entorno de desarrollo para simulación de hardware en bucle (HIL).

Después de haber hablado en el capítulo anterior sobre la simulación en tiempo real y cómo se utiliza durante mucho tiempo en la simulación HIL, y de haber mencionado los tipos de modelos de simulación en tiempo real, en este capítulo, en cambio, se considera un entorno de desarrollo que permite diseñar el modelado para la simulación híbrida en tiempo real. Es decir, que se utiliza el entorno Xilinx System Generator que interactúa con el entorno MatLab permitiendo una buena integración con las bibliotecas de Simulink, lo que a la vez logra convertir una señal continua en una señal discreta y digitalizada. Se crearán ejemplos en el entorno Xilinx System Generator con el objetivo de mostrar el uso de este software en la creación del componente de modelado de simulación en tiempo real.

2.5. Entorno de desarrollo para simulación MatLab/Simulink.

MatLab es un lenguaje de programación para aplicaciones científicas (procesamiento de señales digitales, diseño de simuladores, síntesis de sistemas de control, entre otras.). El entorno de simulación MatLab es un intérprete de comandos, que se puede proporcionar de forma interactiva o estar contenido en archivos en el disco (archivos M). Incluye un gran conjunto de funciones predefinidas y numerosas bibliotecas (cajas de herramientas) para diversas aplicaciones.

El potencial de MatLab se puede ampliar fácilmente (es fácil crear nuevas cajas de herramientas) y es posible convertir un programa MatLab en código C y C ++ de forma automática. A continuación, se muestra la ventana de MatLab que se utiliza para el componente práctico del examen complejo (ver figura 2.2). Simulink es un entorno gráfico que le permite describir y simular sistemas complejos. La simulación de un sistema dinámico con Simulink consta de dos pasos básicos:

1. Fase 1: consiste en crear un modelo gráfico del sistema a simular. Este modelo describe las relaciones matemáticas existentes entre las entradas y salidas del sistema.
2. Fase 2: consiste en utilizar el entorno Simulink para simular el comportamiento del sistema durante su evolución temporal en un período de tiempo establecido por el usuario.

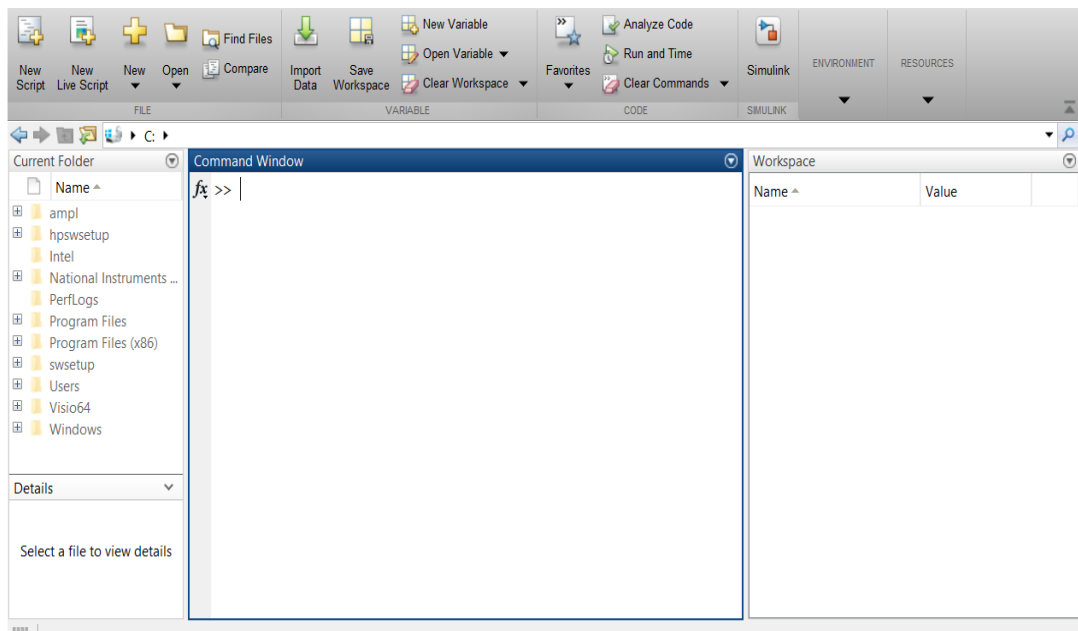


Figura 2. 2: Apertura de 19 ventanas de Matlab.
Elaborado por: Autor.

De hecho, Simulink utiliza la información contenida en el modelo gráfico, que el usuario debe especificar en la fase de descripción, para generar las ecuaciones dinámicas del modelo y resolver numéricamente el problema de simulación en cuestión. Simulink interactúa directamente con MATLAB Workspace. Los modelos descritos, por tanto, pueden contener variables contenidas en el Espacio de Trabajo de la sesión actual.

Asimismo, los resultados de la fase de simulación se pueden pasar directamente al Workspace en forma de nuevas variables, listas para ser analizadas. Los solucionadores que ofrece Simulink para resolver sistemas se dividen en dos familias:

- a. los métodos de integración de pasos variables (véase la tabla 2.1);
- b. los métodos de integración de pasos fijos (véase la tabla 2.2).

Ambos métodos contienen numerosas integraciones.

Tabla 3. 1: Tipos de solucionadores de ODE explícitos de pasos variables

Solucionador de ODE	Orden de precisión	Método
ode45	Medio	Pareja Runge-Kutta, Dormand-Prince (4,5)
ode23	Bajo	Runge-Kutta (2,3) par de Bogacki y Shampine
ode113	Variable, de bajo a alto	Implementación de PECE de Adams-Bashforth-Moulton

Elaborado por: Autor.

Tabla 3. 2: Tipos de solucionadores de ODE explícitos de pasos fijos

Solucionador de ODE	Orden de precisión	Técnica de integración
ode1	Primero	Método de Euler
ode2	Segundo	Método de Heun
ode3	Tercero	Fórmula de Bogacki-Shampine
ode4	Cuarto	Fórmula Runge-Kutta (RK4) de cuarto orden
ode5	Quinto	Fórmula Dormand-Prince (RK5)
ode8	Octavo	Fórmula de Dormand-Prince RK8(7)

Elaborado por: Autor.

2.6. Entorno de simulación System Generator de Xilinx.

System Generator es un entorno de programación y desarrollo para sistemas en tiempo real, que le permite programar dispositivos de hardware, creando un sistema flexible directamente en programación de alto nivel. También permite que los proyectos se compongan de una variedad de entornos, tales como modelos de flujo de datos, lenguajes de diseño de hardware tradicionales - Verilog HDL (VHDL) y funciones derivadas del lenguaje de programación MatLab, que se pueden utilizar en paralelo y sintetizado en hardware de trabajo. Los resultados de la simulación de System Generator tienen la característica de ser precisos en bits y en ciclos, lo que

significa que los resultados obtenidos en la simulación corresponden exactamente a los resultados observados en el hardware. Las simulaciones son significativamente más rápidas que las de los simuladores HDL tradicionales y los resultados son más fáciles de analizar.

2.6.1. Conjuntos de bloques de Xilinx en Simulink.

Los bloques que componen el proyecto no solo reemplazan los componentes de hardware, sino que responden al entorno de diseño, ajustando automáticamente los resultados que se producen. Tanto es así que, para el modelado del componente práctico, los conjuntos de bloques del generador de sistemas se utilizan como los conjuntos de bloques de Simulink y se encuentran en el editor de bloques de Simulink (véase la figura 2.3).

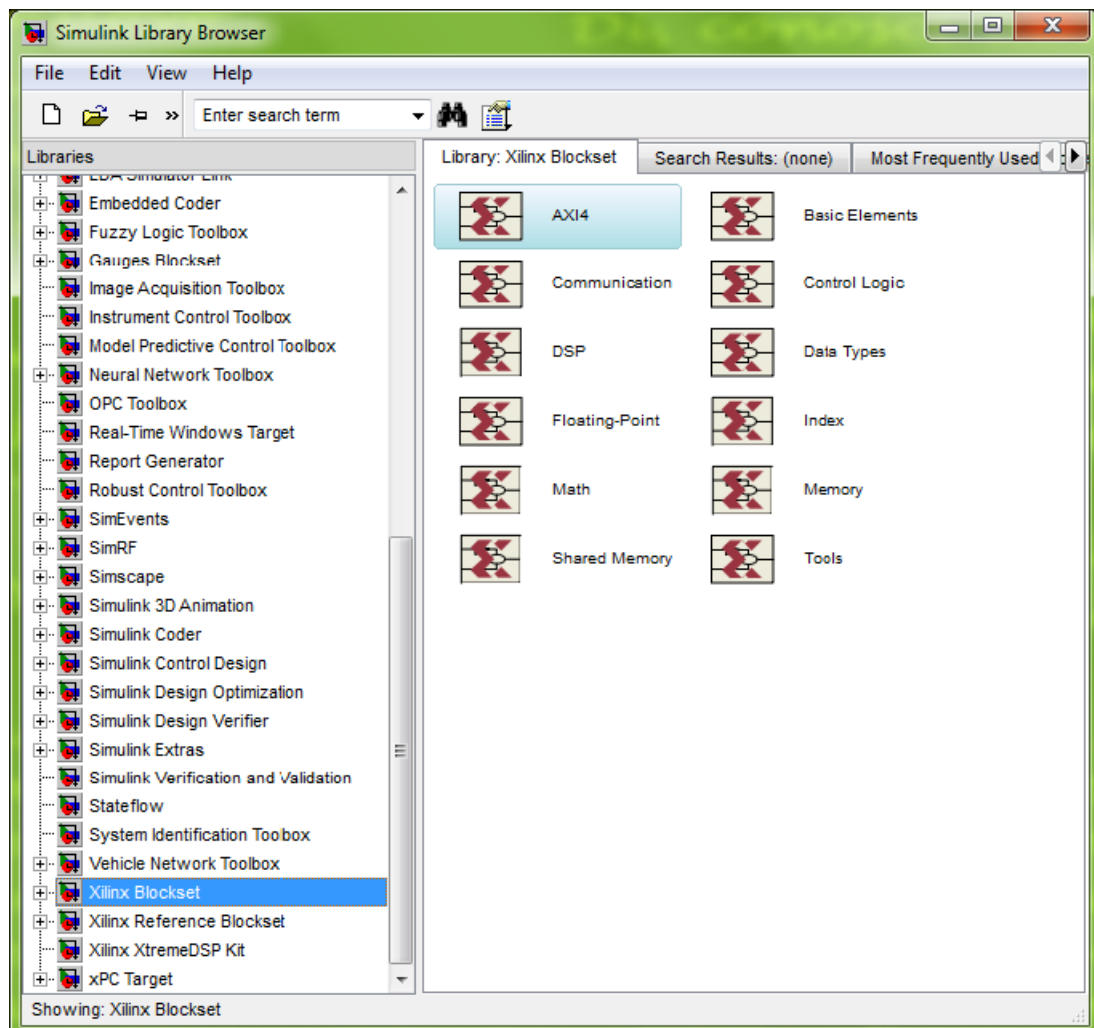


Figura 2. 3: Conjuntos de bloques de System Generator en Simulink.

Elaborado por: Autor.

Los bloques proporcionan abstracciones de funciones matemáticas, lógicas, de memoria y DSP (Procesamiento Digital de Señales) que se pueden utilizar para construir un procesamiento de señales sofisticado y/u otros sistemas. También hay bloques que actúan como interfaces para otras herramientas de software (por ejemplo, FDATool, ModelSim) y también el bloque System Generator para generar el código que se describirá más adelante.

Las dos familias más grandes de conjuntos de bloques que se utilizan son: (a) Xilinx Blockset, que contiene los bloques básicos, algunos de los cuales son de bajo nivel y otros de alto nivel, tal como se muestra en la figura 2.4; y (b) Xilinx Reference Blockset, que contiene los bloques compuestos del generador del sistema, de hecho, cada bloque de este conjunto de bloques es una estructura mixta, es decir, que se puede implementar como un subsistema enmascarado, con parámetros de configuración de bloques.

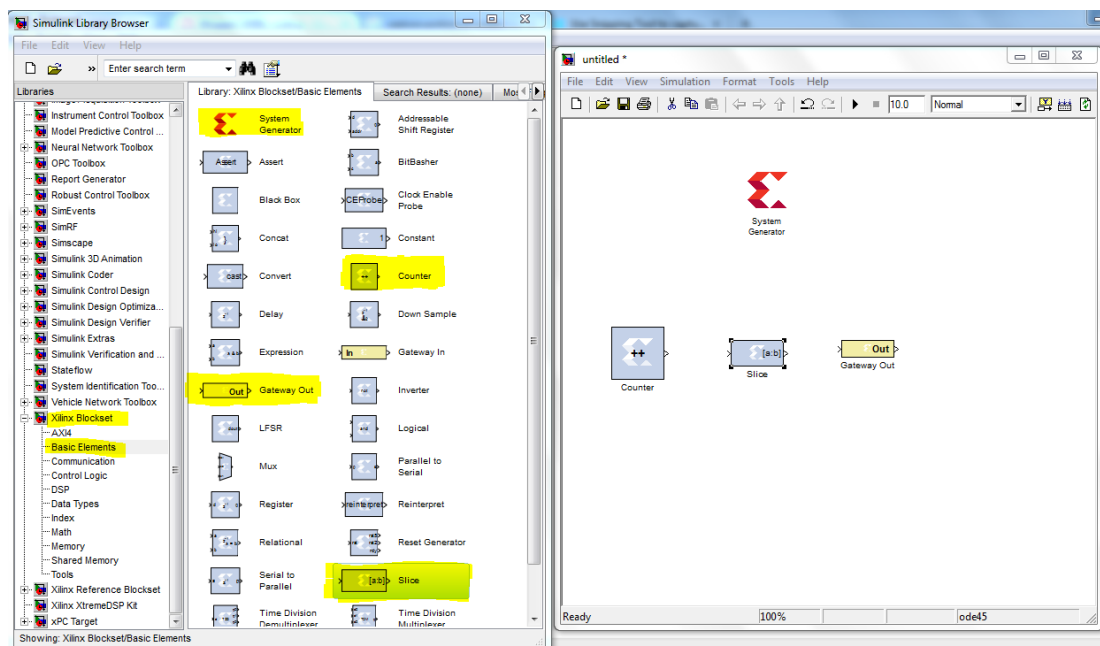


Figura 2. 4: Librería del conjunto de bloques de Xilinx.

Elaborado por: Autor.

2.6.2. Tipos de señales.

En el entorno de simulación System Generator de Xilinx, las señales no son solo bits, sino que pueden ser números de coma fija con o sin signo, y los cambios en el proyecto se traducen automáticamente en cambios apropiados

en el tipo de señal. Para proporcionar simulaciones con precisión de bits en hardware, los conjuntos de bloques procesan la señal de punto fijo de precisión booleana y aleatoria. Sin embargo, dado que los bloques se implementan en el entorno de Simulink que funciona por defecto con señales de doble punto flotante, la conexión entre los bloques Xilinx y los bloques que no son Xilinx está garantizada por los bloques Gateway.

El bloque Gateway convierte una señal de punto flotante doble en una señal Xilinx, y viceversa, con los bloques "Gateway In" y "Gateway Out". En función de sus tipos de entrada, puede establecer el tipo de salida para un bloque especificando cómo se manejará la cuantificación. La capacidad de cuantificación de un bloque incluye un redondeo imparcial hacia más o menos infinito, según el signo o el truncamiento. Las opciones de desbordamiento incluyen saturación y truncamiento.

Además, las simulaciones de System Generator son modelos de bits y ciclos verdaderos. Una simulación es bit-verdadera, cuando en los "límites" el valor producido en la simulación es bit a bit idéntico al valor correspondiente producido en el hardware. Una simulación es un ciclo verdadero cuando en los "límites" los valores producidos corresponden a tiempos reales. Los "límites" del diseño son los puntos donde existen los bloques Gateway System Generator, de hecho, cuando un proyecto se traduce en hardware, Gateway In (respectivamente, Gateway Out) se convierte en un puerto de entrada de alto nivel (respectivamente, salida).

Los proyectos de System Generator son sistemas de tiempo discreto. En otras palabras, las señales y los bloques que las producen tienen frecuencias de muestreo asociadas. La frecuencia de muestreo de un bloque determina la frecuencia con la que se despierta el bloque (lo que permite que se actualice su estado). Un modelo simple que muestra el comportamiento de los sistemas de tiempo discreto es el de la figura 2.5; contiene una puerta de enlace que es impulsada por un generador de Simulink (onda sinusoidal). El bloque Gateway In está configurado con un período de muestreo de un segundo.

El bloque Gateway Out convierte la señal de punto fijo a punto flotante doble (para que pueda analizarse en el entorno de Simulink), pero la frecuencia de muestreo no cambia. La salida introducida en Simulink Scope muestra la versión sin alterar y muestreada de la onda sinusoidal.

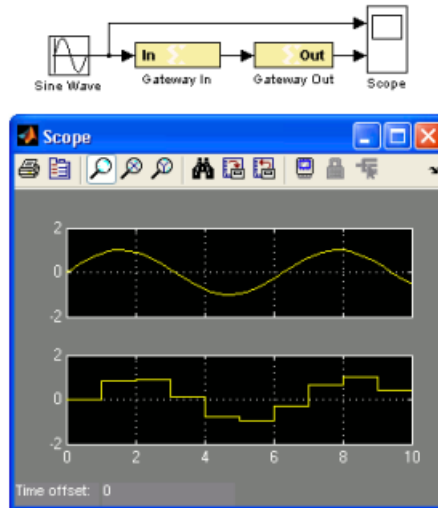


Figura 2. 5: Aplicación de System Generator de Xilinx en Simulink.
Elaborado por: Autor.

2.6.3. Proyectos Multifase.

System Generator también admite proyectos multifase, es decir, proyectos que tienen señales que se ejecutan a diferentes frecuencias de muestreo y las compilan automáticamente en el hardware. De esta forma, los proyectos son factibles de forma natural y sencilla para Simulink. Cabe señalar que algunos conjuntos de bloques se pueden sobremuestrear, lo que significa que su procesamiento interno se realiza a una velocidad mayor que su velocidad de datos.

En hardware, esto significa que el bloque necesita más de un ciclo de reloj para procesar una muestra de datos. En Simulink, estos bloques no tienen un efecto observable en las frecuencias de muestreo; de hecho, los bloques sobremuestreados no provocan una variación explícita de la frecuencia de muestreo, pero System Generator considera la frecuencia del bloque interno, junto con todas las demás frecuencias de muestreo cuando pasa a la generación. del reloj lógico para la implementación del hardware. Esto significa que debe tenerse en cuenta la tasa de procesamiento interno de los bloques sobremuestreados al especificar el valor del período del

sistema Simulink en el cuadro de diálogo del bloque del constructor del sistema.

2.6.4. Métodos de compilación.

La compilación de bajo nivel del proyecto se realiza automáticamente en System Generator. Las formas en que crea una plantilla varían y dependen de la configuración en el bloque System Generator. Además de producir descripciones HDL de hardware, la herramienta genera archivos auxiliares. Algunos archivos (por ejemplo, archivos de proyecto y de restricciones) ayudan a las herramientas posteriores, mientras que otros (por ejemplo, banco de pruebas de VHDL) se utilizan para la verificación del diseño.

Los proyectos se compilan y simulan utilizando el bloque System Generator (véase la figura 2.6). Antes de que un proyecto se pueda simular o traducir a hardware, debe incluir el bloque System Generator. Al crear un nuevo modelo, debe agregar un bloque de System Generator inmediatamente. Al presionar “Generate”, System Generator compila el modelo o parte del modelo en un lenguaje equivalente de bajo nivel.

El tipo de compilación especifica los diferentes resultados que se obtienen, y estos son:

- Dos tipos de listas de red: HDL Netlist y NGC Netlist.
HDL Netlist es el tipo más utilizado. En este caso, el resultado es una colección de archivos HDL y algunos archivos auxiliares que simplifican el procesamiento posterior. La colección está lista para ser transformada con una herramienta de síntesis, en una secuencia de instrucciones para configurar, por ejemplo, una FPGA. Los archivos que se crean se describen con más detalle para su compilación. NGC Netlist es similar a HDL Netlist, pero los archivos resultantes son archivos NGC en lugar de archivos HDL;
- Bitstream produce una secuencia de configuración FPGA que está lista para ejecutarse en una plataforma de hardware FPGA. Con esta compilación FPGA puede participar en simulaciones en Simulink;

- Herramienta de exportación EDK, para exportar el Xilinx Embedded Development Kit en diferentes variedades de co-simulación de hardware;
- Análisis de tiempo: un informe sobre el tiempo del proyecto.

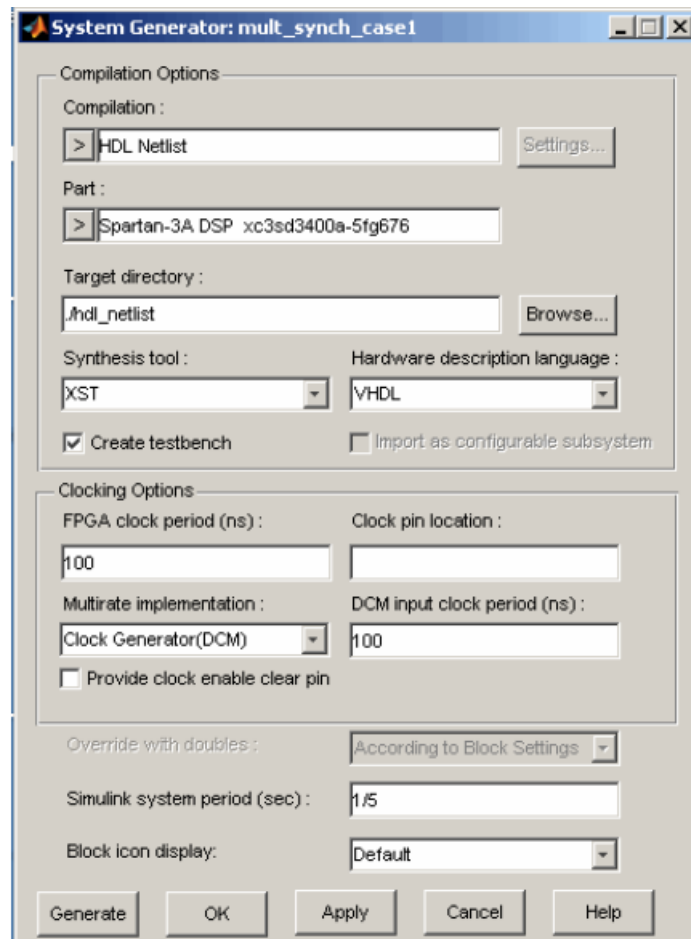


Figura 2. 6: Generar esquema de compilación en lenguaje de bajo nivel.
Elaborado por: Autor.

Además de la posibilidad de ser leídos por otros lenguajes estándar, como ya se mencionó anteriormente, es posible a través del bloque Black Box introducir en la simulación otros lenguajes estándar (como VHDL) que pueden componer parte de la simulación.

Capítulo 3: Desarrollo del componente práctico.

En esta sección se desarrollan aplicaciones prácticas para la materia de electrónica de potencia mediante simulación en System Generator y Simulink que contribuyen a la formación de los estudiantes que siguen la Carrera de Electricidad.

3.1. Descripción general de las aplicaciones prácticas.

A continuación, se muestran algunos ejemplos realizados en el entorno System Generator de Xilinx en el campo de la electrónica de potencia. Estos son ejemplos en los que se realiza el modelado para la simulación en tiempo real que es la parte principal para luego trabajar con Hardware-in-the-Loop.

En las aplicaciones prácticas implementadas, los conjuntos de bloques (blocksets) del entorno System Generator en el caso de filtros de primer orden tendrán como señales de entrada un generador en Simulink en el que la frecuencia aumenta linealmente con el tiempo, mientras que para los inversores se utilizará un generador Simulink con frecuencia fija. En la última aplicación práctica se utiliza un generador de ondas cuadradas de Simulink. Se ejecutará en todas las simulaciones de la parte discreta con un tiempo de muestreo de alrededor de 1 μ s o incluso menor, ya que normalmente es el tiempo con el que se ejecutan las FPGA para tener una cuantificación apreciable de la señal.

3.2. Aplicación práctica 1: Simulación de filtro de paso bajo.

En esta sección se considera un filtro de primer orden, es decir, un filtro de paso bajo RC. Primero se analiza el sistema (circuito) que consta de un generador de voltaje alterno ($v_{in}(t)$), y dos elementos pasivos, una resistencia (R) y un condensador (C), tal como se muestra en la figura 3.1. Aplicando la Ley de Kirchhoff de Voltaje se obtiene una ecuación con respecto al voltaje de salida a través del capacitor:

$$v_{out}(t) = -Ri_R(t) + v_{in}(t)$$

$$v_{out}(t) = -Ri_c(t) + v_{in}(t)$$

$$v_{out}(t) = -R \left[C \frac{dv_c(t)}{dt} \right] + v_{in}(t)$$

$$v_{out}(t) = -RC \frac{dv_{out}(t)}{dt} + v_{in}(t)$$

Aplicando la transformada de Laplace, queda:

$$V_{out}(s) = -RCsV_{out}(s) + V_{in}(s)$$

$$V_{out}(s) + RCsV_{out}(s) = V_{in}(s)$$

$$V_{out}(s)(1 + RCs) = V_{in}(s)$$

Se asigna a $\tau = RC$ y se obtiene como función de transferencia,

$$W(s) = \frac{V_{out}(s)}{V_{in}(s)} = \frac{1}{(1 + \tau s)}$$

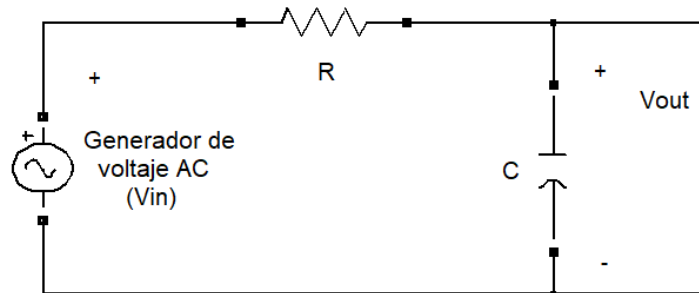


Figura 3. 1: Circuito esquemático de un filtro pasa bajo RC.
Elaborado por: Autor.

Esta función de transferencia corresponde a un sistema de primer orden y se utilizará para diseñar el esquema discreto que será implementado en System Generator de Xilinx disponible en la librería de Simulink. Obsérvese que el módulo de la función de transferencia; hay un punto de ruptura en,

$$\tilde{\omega} = \left| \frac{1}{\tau} \right|$$

y tiene un ancho de banda (a -3dB) de,

$$B_3 = \frac{1}{2\pi\tau}$$

En la figura 3.2 se muestra el diagrama de Bode (solamente frecuencia) con valores de las variables que serán posteriormente analizadas para valores asignados en la resistencia y capacitor. Para simular el filtro pasa bajo de

primer orden, se utiliza el software MatLab/Simulink. Para tener un uso más conveniente, pero sobre todo más eficiente de las variables que se utilizarán en la simulación, se crea un archivo script (tipo m), que permite ser leído por MatLab e interactuar con la simulación a través del “Workspace”. El código script define el valor de la resistencia ($R = 10\text{ k}\Omega$), del condensador ($C = 1\text{ }\mu\text{F}$) y de la constante τ (véase el código script).

```
%% Componente práctico - Hugo Vega %%
%% Práctica 1: Filtro pasa bajo %%
R = 1e4; ... Ohm
C = 1e-6; ... Faradios
tau = R*C; ... constante de tiempo

% Cálculo de la FT en tiempo continuo
s = tf('s');
W = 1/(1+tau*s);
figure()
bode(W), grid; ... graficar la FT
```

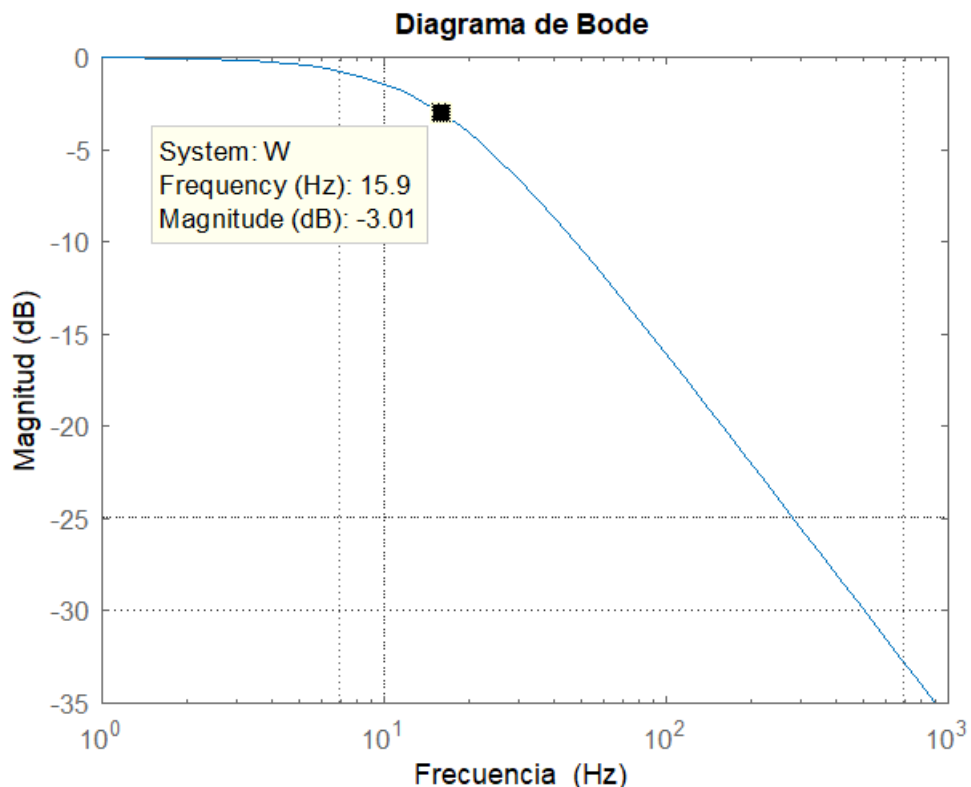


Figura 3. 2: Diagrama de Bode real (rojo) y asintótico (azul).
Elaborado por: Autor.

En Simulink, se impone un solucionador de pasos fijos (tipo Euler) como tipo de solucionador. Para diseñar el filtro pasa bajo a través del entorno

System Generator de Xilinx se toman primero las ecuaciones previamente calculadas que deben ser discretizadas, esto es posible con el uso de la transformada bilineal, que permite trabajar en transformadas Zeta de tiempo discreto. La transformada de bilineal dice que,

$$s = \frac{2z - 1}{Tz + 1}$$

Sustituyendo s en la función de transferencia del filtro pasa bajo, se obtiene,

$$W(z) = \frac{1}{1 + \tau \left(\frac{2z-1}{Tz+1} \right)} = \frac{z + 1}{\left(1 + \frac{2\tau}{T} \right) z + \left(1 - \frac{2\tau}{T} \right)}$$

luego:

$$W(z) = \frac{T}{2\tau + T} \cdot \frac{1 + z^{-1}}{1 + \left(\frac{T-2\tau}{T+2\tau} \right) z^{-1}} = b_0 \frac{1 + b_1 z^{-1}}{1 + a_0 z^{-1}}$$

donde,

$$b_0 = \frac{T}{2\tau + T}$$

$$b_1 = 1;$$

$$a_0 = \frac{T - 2\tau}{T + 2\tau}.$$

Se calculan b_0, b_1 e a_0 para poder insertar los valores en los bloques del entorno System Generator de Xilinx. Para elegir un valor de período de muestreo adecuado, es necesario cumplir dos condiciones, que $f_c > 2 f_m$, es decir, que la frecuencia de muestreo sea al menos dos veces superior a la frecuencia máxima del espectro de la señal que se va a muestrear; ya que la frecuencia de Nyquist $f_N \gg B_3$ es mucho mayor que la banda de paso y esto implica que el tiempo de muestreo sea un orden de magnitud menor que la banda de paso.

Las herramientas que se utilizan para construir el modelo son las que suman o restan (AddSub), las que multiplican constantes (CMux) y las que

actúan como retardo en la transformada Z (Delay). Pero para que el sistema discreto funcione, se requiere los tres bloques fundamentales Gateway In, Gateway Out y el System Generator. El sistema obtenido está formado por una señal de entrada que se multiplica por una constante (b_0) la señal entra en un sistema de retroalimentación no unitario, se multiplica por el dispositivo de retardo y en la sección de bucle cerrado se multiplica por una constante (a_0) después de lo cual la señal se multiplica por otro retardador de bucle abierto multiplicado por otra constante (b_1), tal como se muestra en la figura 3.3.

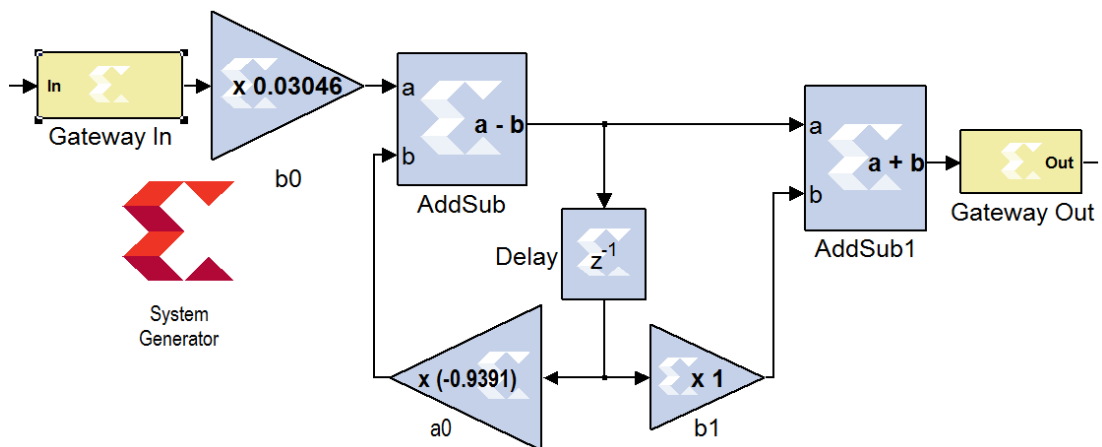


Figura 3. 3: Diagrama de bloques del filtro de tiempo discreto.
Elaborado por: Autor.

La figura 3.4 muestra la señal discretizada mediante el entorno System Generator de Xilinx, superpuesta a la señal filtrada obtenida en tiempo continuo gracias a los bloques presentes en la librería Simulink. Se puede obtener una señal de tiempo discreta muy aceptable. Cuanto más se vaya a reducir el tiempo de muestreo, mayor será la precisión de la señal, pero esto implica tener un instrumento real que luego sea capaz de trabajar en ese momento.

Se puede ver que para valores de frecuencia de la señal de entrada mucho más bajos que B_3 , el sistema no induce ninguna atenuación significativa ni variación de fase significativa, mientras que, a medida que aumenta la frecuencia de entrada, la atenuación y la variación de fase se vuelven no despreciables. En particular, a medida que la frecuencia diverge, el filtro tiende a "bloquear" completamente la señal de entrada.

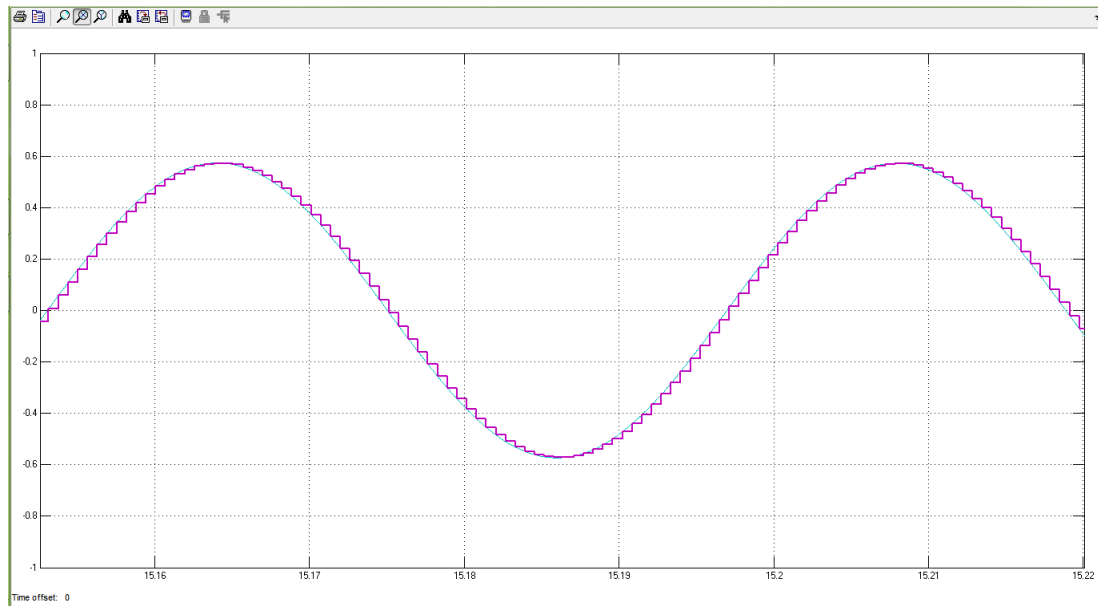


Figura 3. 4: La senoide azul es la se\u00f1al filtrada en tiempo continuo, la p\u00farpura es la discretizada.

Elaborado por: Autor.

Si se utiliza un valor de frecuencia de muestreo igual a $f_c = 100 \text{ Hz}$ siempre que tenga valores de frecuencia de se\u00f1al de entrada bajos, no se tendr\u00e1 problemas de muestreo de se\u00f1al, mientras que tan pronto como la frecuencia comience a aumentar, los problemas comenzar\u00e1n a surgir. Esto se debe a que ya no se cumplen todas las condiciones, en particular la de tener una frecuencia de Nyquist mucho mayor que la banda de paso del sistema de bucle cerrado. Sin embargo, debe recordarse que los dispositivos reales con un per\u00edodo de muestreo muy peque\u00f1o tienen un costo elevado.

3.3. Aplicaci\u00f3n pr\u00e1ctica 2: Simulaci\u00f3n de filtro de paso alto.

En esta secci\u00f3n se realiza un filtro RC pasa alto (se considera un filtro de primer orden) de manera similar a la aplicaci\u00f3n pr\u00e1ctica 1. El sistema que consta de un generador de voltaje alterna ($v_{in}(t)$), un condensador (C) y una resistencia (R), tal como se muestra en la figura 3.5. Para un filtro pasa alto se considera el voltaje de salida a trav\u00e9s de la resistencia:

$$v_{out}(t) = Ri_c(t) = R \left[C \frac{dv_c(t)}{dt} \right]$$

$$v_{out}(t) = RC \frac{dv_c(t)}{dt}$$

se sabe que $\tau = RC$, entonces la función de transferencia,

$$V_{out} = \frac{\tau s * V_{in}}{(1 + \tau s)}$$

y se obtiene como función de transferencia,

$$W(s) = \frac{V_{out}(s)}{V_{in}(s)} = \frac{\tau s}{(1 + \tau s)}$$

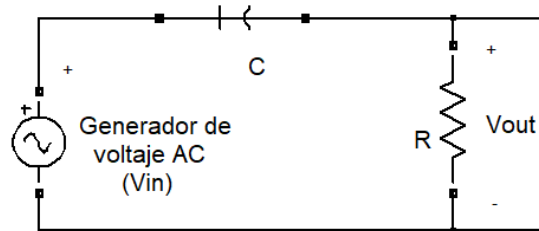


Figura 3. 5: Circuito esquemático de un filtro pasa alto RC.
Elaborado por: Autor.

En la figura 3.2 se muestra el diagrama de Bode (solamente frecuencia). Se observa en el módulo de la función de transferencia, que hay un punto de ruptura en $\tilde{\omega} = |\frac{1}{\tau}|$ y tiene una banda de paso (a -3dB) de $B_3 = \frac{1}{2\pi*\tau}$ para una frecuencia de 15.9 Hz.

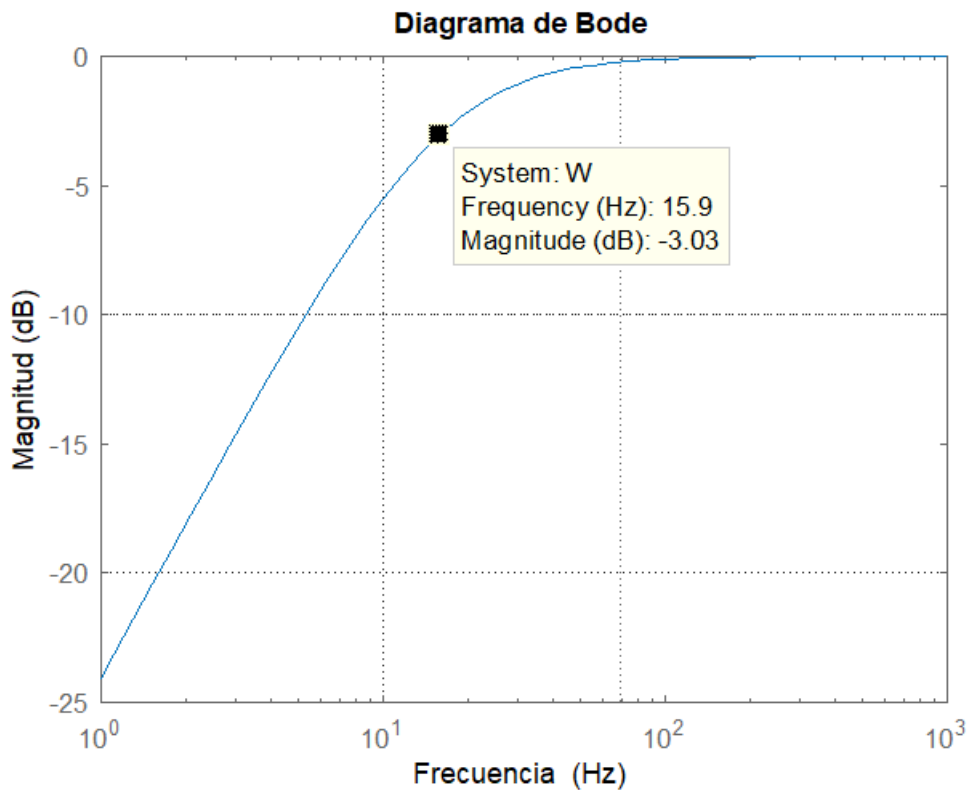


Figura 3. 6: Diagrama de Bode real (rojo) y asintótico (azul).
Elaborado por: Autor.

Para simular el filtro previamente analizado en tiempo real, se utiliza el software Simulink. Para tener un uso más conveniente, pero sobre todo más eficiente de las variables que se utilizarán en la simulación, se crea un archivo script en MatLab. El archivo define el valor de resistencia ($R = 10\text{ k}\Omega$), el condensador ($C = 1\text{ }\mu\text{F}$), y la constante τ (ver código A). En Simulink, se impone un solucionador de pasos fijos (tipo Euler) como tipo de solucionador.

```
%% Componente práctico - Hugo Vega %%
%% Práctica 2: Filtro pasa alto %%
R = 1e4; ... Ohm
C = 1e-6; ... Faradios
tau = R*C; ... constante de tiempo
% Cálculo de la FT en tiempo continuo
s = tf('s');
W = (tau*s)/(1+tau*s); ... FT continua
figure()
bode(W), grid;
```

Para diseñar el filtro se utiliza el entorno System Generator de Xilinx, primero retomando las fórmulas previamente calculadas y como en la aplicación práctica 1 (ver sección 3.2) se discretiza la función de transferencia mediante la transformada bilineal, y está dada por,

$$W(z) = \frac{2\tau}{T + 2\tau_1 + \left(\frac{T-2\tau}{T+2\tau}\right)z^{-1}} = b_0 \frac{1 + b_1z^{-1}}{1 + a_0z^{-1}}$$

donde,

$$b_0 = \frac{2\tau}{T + 2\tau}$$

$$b_1 = -1$$

$$a_0 = \frac{T - 2\tau}{T + 2\tau}$$

De manera similar al filtro pasa bajo, las consideraciones anteriores se aplican tanto a la elección del tiempo de muestreo como a la creación del modelo en System Generator de Xilinx. La figura 3.7 muestra el diseño del circuito RC equivalente a un filtro pasa alto. La figura 3.8 muestra el resultado obtenido de la simulación en tiempo real del filtro pasa alto. El filtro logra atenuar las frecuencias por debajo de su frecuencia de corte bastante bien

$f_t = 15.91 \text{ Hz}$. En cuanto al período de muestreo, surgen los mismos problemas encontrados en el análisis del filtro de paso bajo.

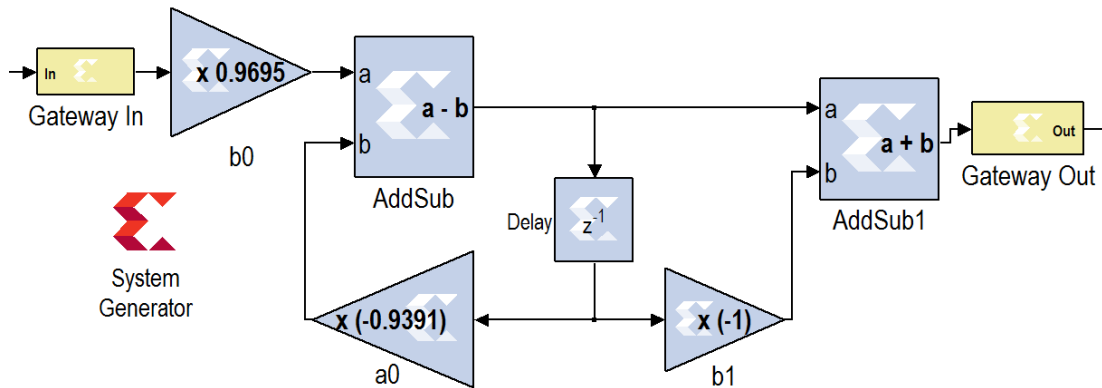


Figura 3. 7: Diagrama de bloques del filtro pasa alto discretizado.
Elaborado por: Autor.

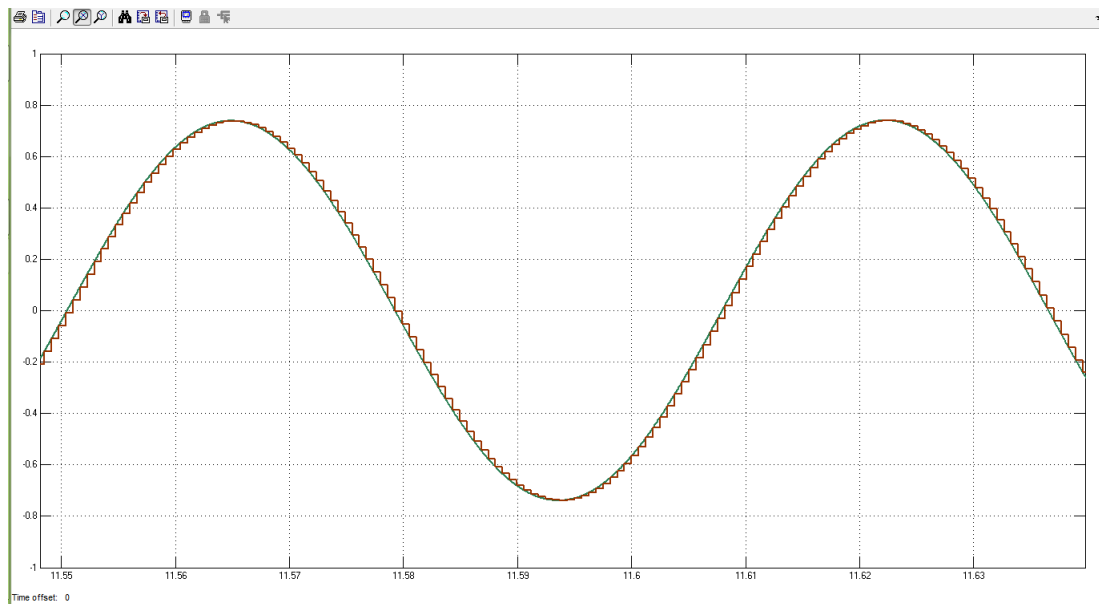


Figura 3. 8: La sinusoide verde es la señal filtrada en tiempo continuo, la roja es discretizada.

Elaborado por: Autor.

3.4. Aplicación práctica 3: Simulación de inversor de medio puente.

Esta configuración se denomina inversor de medio puente; consta de dos interruptores conectados en serie (compuestos por dispositivos activos como BJT, IGBT), cada uno de los cuales está equipado con un diodo antiparalelo para que el interruptor sea unidireccional en voltaje y bidireccional en corriente. La carga está conectada entre los puntos intermedios de las dos ramas. La serie de interruptores se conecta en paralelo a la serie de dos

condensadores iguales, cuya capacidad debe ser alta para que el voltaje central permanezca casi constante e igual,

$$\frac{V_{dc}}{2} = E$$

La figura 3.9 muestra el circuito esquemático del inversor de medio puente con carga óhmica-inductiva. Esta estructura permite obtener solo dos niveles de tensión de salida, que, gracias a la presencia de los diodos antiparalelos en los interruptores, son independientes de la dirección de la corriente.

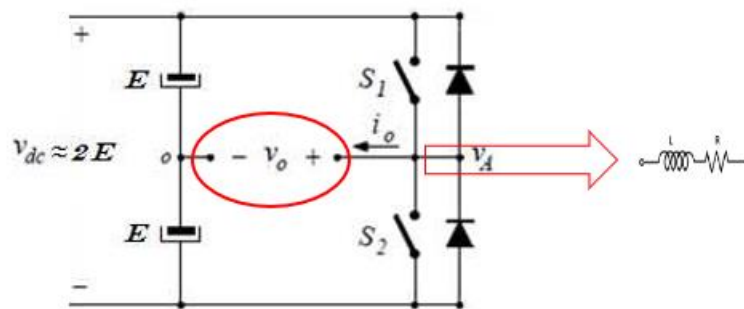


Figura 3. 9: Circuito esquemático del inversor de medio puente con carga óhmica-inductiva.

Elaborado por: Autor.

El primer nivel se puede obtener con la configuración del interruptor S_1 cerrada y el interruptor S_2 abierta, y la tensión de salida $v_{out}(t)$ aplicada a la carga es positiva e igual a $\frac{V_{dc}}{2}$, mientras que, con el segundo nivel, dado por el interruptor S_1 abierto y el interruptor S_2 cerrado, en consecuencia, la salida de voltaje conduce a un valor negativo igual a $-\frac{V_{dc}}{2}$.

Los retrasos y las imprecisiones en los comandos pueden hacer que la conducción de los dos interruptores se superponga, lo que provocaría un cortocircuito casi siempre destructivo entre la fuente de alimentación en el lado de CC. Para superar este tipo de problema, existe la inserción de un tiempo muerto ("dead time") para garantizar que cada interruptor (S_1 y S_2) de rama esté realmente abierto cuando el otro se cierre. Actualmente las principales técnicas de modulación que permiten regular la frecuencia de la tensión de

salida pueden ser aquellas con modulación de onda cuadrada o aquellas con ancho de pulso (PWM).

Para poder crear una simulación clara y con el objetivo de poder construir un sistema en el que se pueda ver la funcionalidad real de la interconexión entre Simulink y System Generator de Xilinx, se decidió hacer algunas simplificaciones en el diseño del inversor de medio puente. Se considera los dos interruptores conectados en serie como dos interruptores simples que se abren y se cierran, sin tener en cuenta el "tiempo muerto" indicado anteriormente.

Se observa un inversor de medio puente que tiene una carga óhmico-inductiva, con la intención de ir a ver la corriente en la salida de la carga, y luego considerar que:

- $\delta(t)$ cuando S_1 cerrado y S_2 abierto, se tiene que:

$$v_L(t) = L \frac{di_{out}}{dt} = \frac{V_{DC}}{2} - i_{out}R$$

- $[1 - \delta(t)]$ cuando S_1 abierto y S_2 cerrado, se tiene que:

$$v_L(t) = L \frac{di_{out}}{dt} = -\frac{V_{DC}}{2} - i_{out}R$$

Igualando las últimas dos ecuaciones se tiene que:

$$\begin{aligned} v_L(t) = L \frac{di_{out}}{dt} &= \left(\frac{V_{DC}}{2} - i_{out}R \right) \delta(t) + \left(-\frac{V_{DC}}{2} - i_{out}R \right) [1 - \delta(t)] = \\ &= \frac{V_{DC}}{2} \delta(t) - i_{out}R \delta(t) - \frac{V_{DC}}{2} - i_{out}R + \frac{V_{DC}}{2} \delta(t) + i_{out}R \delta(t) \\ &= V_{DC} \delta(t) - \frac{V_{DC}}{2} - i_{out}R. \end{aligned}$$

Por tanto,

$$L \frac{di_{out}}{dt} + i_{out}R = V_{DC} \delta(t) - \frac{V_{DC}}{2}$$

En la figura 3.10 se muestra el diagrama de bloques para proceder a discretizar la función de transferencia del filtro pasa alto.

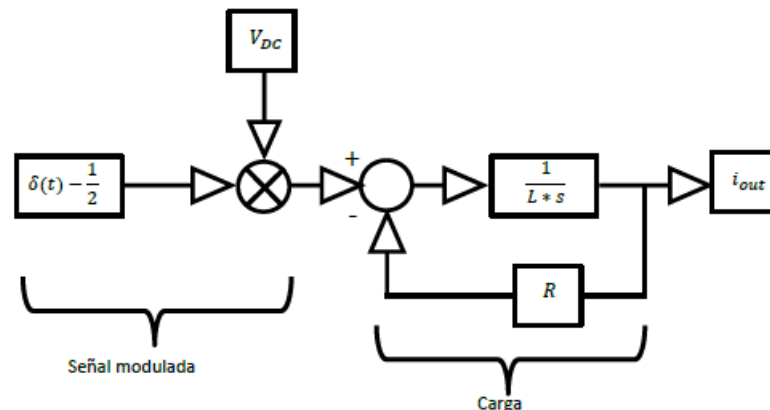


Figura 3. 10: Diagrama de bloques del inversor de medio puente.
Elaborado por: Autor.

Para crear la señal modulada lista para ser posteriormente filtrada por la carga, se intenta reproducir el método de suboscilación sinusoidal en el que las inversiones de voltaje de fase de un inversor se hacen coincidir con las intersecciones de dos triples de señales de diferente frecuencia. La señal de menor frecuencia se denomina moduladora (función de modulación) que en el caso bajo análisis es una señal sinusoidal con frecuencia igual a la fundamental $f = 50\text{Hz}$ y amplitud 1; el que tiene la frecuencia más alta se llama portadora, que es una señal triangular periódica con frecuencia de conmutación $f_{ws} = 20\text{kHz}$ y valores de campo de oscilación entre -1 y 1.

Se ilustra en la siguiente página el código en script de Matlab. El propósito de la modulación de suboscilación es obtener un voltaje que, aunque varía entre $\frac{V_{dc}}{2}$ y $-\frac{V_{dc}}{2}$, tiene un espectro de baja frecuencia idéntico al de la moduladora. Esta técnica es parte de la familia de técnicas de modulación por ancho de pulso (PWM). Una vez que se ha obtenido la señal moduladora, la atención debe dirigirse a la carga. Se observa su función de transferencia es,

$$P(s) = \frac{1}{Ls + R};$$

Como valores de la carga óhmico-inductiva se da a la resistencia $R = 10\ \Omega$ y a la inductancia $L = 0.2\text{mH}$.


```

%% Componente práctico - Hugo Vega %%
%% Práctica 3: Inversor de medio puente %%
fs = 500; ... Hz (frecuencia fundamental);
ws = 2*pi*fs; ... [rad/sec];
fws = 20*10^3; ... frecuencia de conmutación;
Tws = 1/fws; ... tiempo de conmutación;
Tm = Tws/2;
L = 2e-4; ... inductancia
R = 10; ... resistencia
Vdc = 1000;
tau = L/R;
s = tf('s');
P = 1/(R+L*s);
% Valores para la discretización
T = Tws/20;
b0 = (T)/(T*R+2*L);
a0 = (T*R-2*L)/(T*R+2*L);
b1 = 1;

```

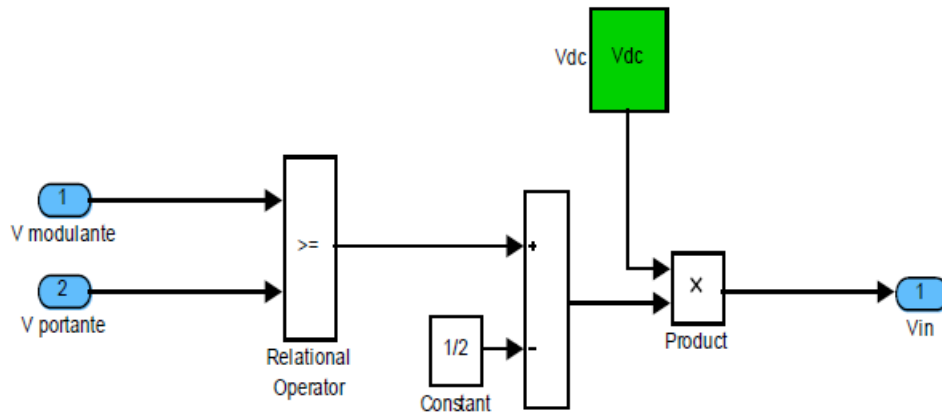


Figura 3. 11: Representación del diagrama de bloques de Simulink de la señal moduladora.

Elaborado por: Autor.

En la simulación en Simulink (véase la figura 3.12) se ha establecido una resolución de tiempo variable y como tipo de solucionador "ode45 (Dormand-Prince)". Se impone una vigésima parte del período de conmutación "Tamaño máximo de paso" y la duración de la simulación varía de 0 a 0.5 s. Con los paquetes System Generator es posible discretizar la carga partiendo de su función de transferencia a través de la transformada bilineal, si

$$s = \frac{2z - 1}{Tz + 1}$$

entonces,

$$W(z) = \frac{1}{\frac{2z-1}{Tz+1} * L + R} = \frac{T(z+1)}{2L(z-1) + TR(z+1)} = \frac{T(z+1)}{2Lz - 2L + TRz + TR}$$

$$= \frac{T(z+1)}{z(2L + TR) + (-2L + TR)} = \frac{T}{TR + 2L} * \frac{1 + z^{-1}}{1 + \left(\frac{TR - 2L}{TR + 2L}\right)z^{-1}}$$

$$W(z) = b_0 \frac{1 + b_1 z^{-1}}{1 + a_0 z^{-1}}$$

Teniendo como valores las variables:

$$b_0 = \frac{T}{TR + 2L}$$

$$b_1 = 1$$

$$a_0 = \frac{TR - 2L}{TR + 2L}$$

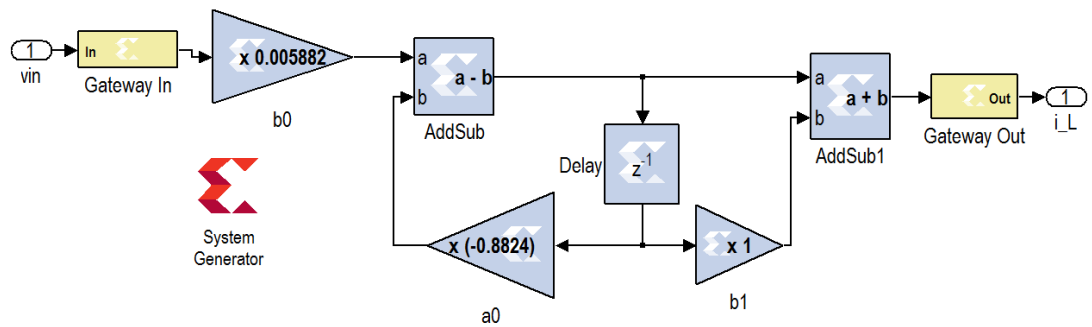


Figura 3. 12: Diagrama de bloques de la carga en System Generator de Xilinx.
Elaborado por: Autor.

Para elegir un valor de período de muestreo adecuado, es necesario cumplir dos condiciones, que $f_c > 2f_m$, es decir, que la frecuencia de muestreo sea al menos dos veces superior a la frecuencia máxima del espectro de la señal que se va a muestrear; ya que la frecuencia de Nyquist es mucho mayor que la banda de paso ($f_N \gg B_3$) y esto significa que el tiempo de muestreo es un orden de magnitud menor que la banda de paso. En el caso considerado, se toma un tiempo de muestreo igual a $T = \frac{1}{f_c}$ con $f_c = (f_{ws} * 20) \text{ Hz}$ que seguramente satisface todas las condiciones. El código de script en MatLab se muestra en la página anterior.

La figura 3.13 muestra la gráfica correspondiente a la corriente de salida para inversores de medio puente. Se puede notar que la señal discretizada

tiene una excelente aproximación a la señal en tiempo continuo, lo que demuestra un 99% de confiabilidad del algoritmo de programación propuesto.

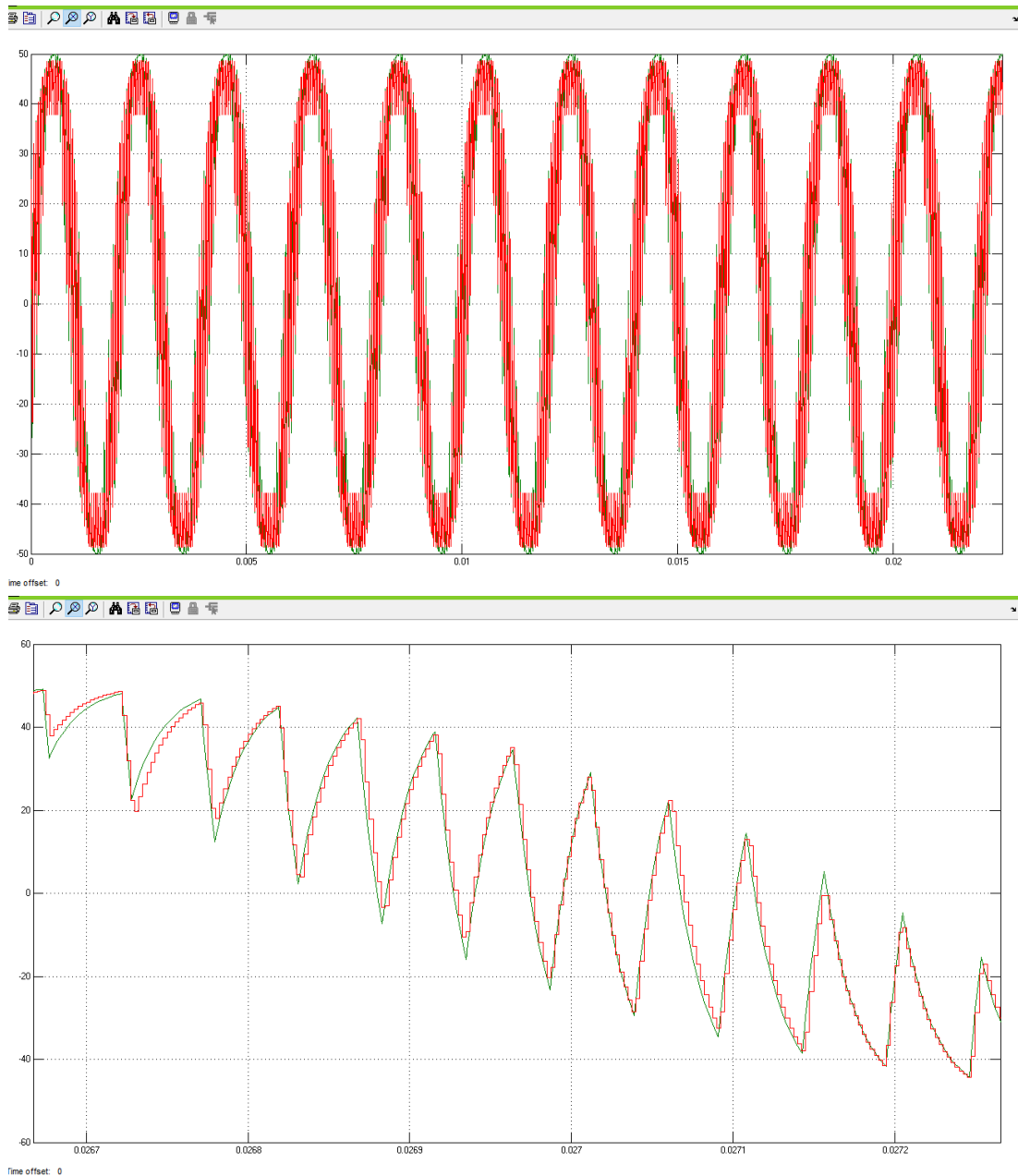


Figura 3. 13: Diagrama de corriente de salida para inversores de medio puente.
Elaborado por: Autor.

3.5. Aplicación práctica 4: Simulación de inversor de puente.

Se trata de un esquema de puente, formado por la conexión en serie de dos inversores de medio puente (véase la figura 3.14) donde la tensión total de salida es $v_0(t) = v_{A0}(t) - v_{B0}(t)$ y se tiene como señal modulada una señal que va de V_{DC} a $-V_{DC}$. También en esta aplicación se realizan algunas simplificaciones, como las de los inversores de medio puente.

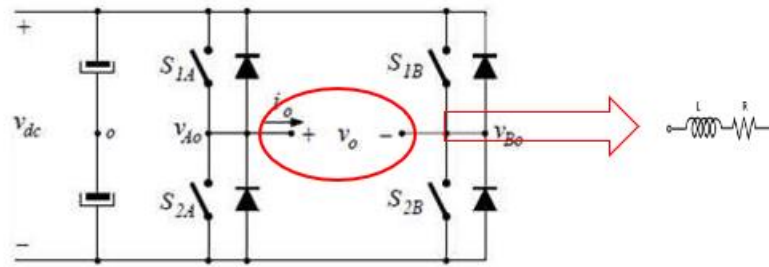


Figura 3. 14: Circuito esquemático de inversor de puente.
Elaborado por: Autor.

Para crear la señal modulada que será posteriormente filtrada por la carga, se utiliza la técnica de modulación PWM, en la que los interruptores diagonalmente opuestos (S_{1A} y S_{2B}) y (S_{1B} y S_{2A}) de las dos ramas (véase la figura 3.14) se controlan, respectivamente, como pares de interruptores 1 y 2. La señal de menor frecuencia se denomina moduladora (función de modulación) que en el caso en consideración utiliza una señal sinusoidal con una frecuencia igual a la fundamental de $f = 50 \text{ Hz}$ y amplitud 1; el que tiene la frecuencia más alta se llama portadora, que no es más que una señal periódica triangular con frecuencia de conmutación $f_{ws} = 20 \text{ kHz}$ y valores de campo de oscilación de -1 a 1. Para un inversor de puente con carga óhmico-inductiva, y luego considerar que:

- $\delta(t)$ cuando $S_{1A} - S_{2B}$ cerrado y $S_{2A} - S_{1B}$ abierto

$$v_L(t) = L \frac{di_{out}}{dt} V_{DC} - i_{out} * R;$$

- $[1 - \delta(t)]$ cuando $S_{1A} - S_{2B}$ abierto y $S_{2A} - S_{1B}$ cerrado

$$v_L(t) = L \frac{di_{out}}{dt} V_{DC} - i_{out} * R;$$

se hace la igualdad de las dos ecuaciones

$$\begin{aligned} v_L(t) = L \frac{di_{out}}{dt} &= (V_{DC} - i_{out} * R) * \delta(t) + (-V_{DC} - i_{out} * R) * [1 - \delta(t)] = \\ &= V_{DC} \delta(t) - i_{out} * R \delta(t) - V_{DC} - i_{out} * R + V_{DC} \delta(t) + i_{out} * R \delta(t) \\ &= 2 * V_{DC} \delta(t) - V_{DC} - i_{out} * R. \end{aligned}$$

La figura 3.15 muestra el diagrama esquemático que representa a la señal modulada y carga para el inversor de puente. Las figuras 3.16 y 3.17

muestran los bloques implementados en Simulink para la señal moduladora y de la carga a discretizar, respectivamente.

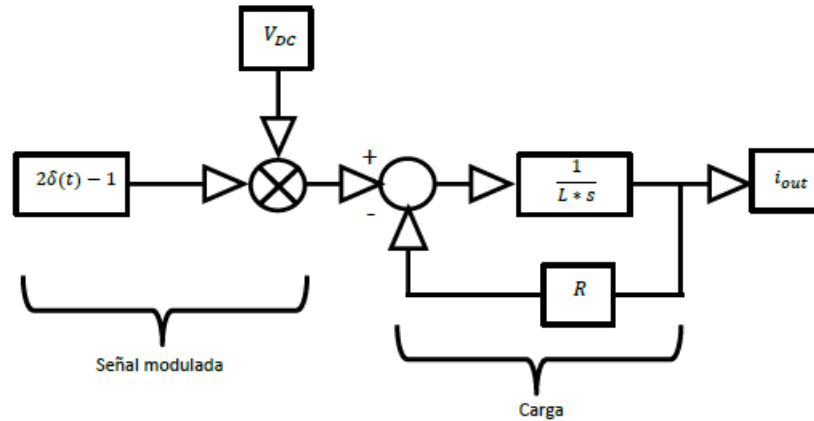


Figura 3. 15: Diagrama de bloques del inversor puente.
Elaborado por: Autor.

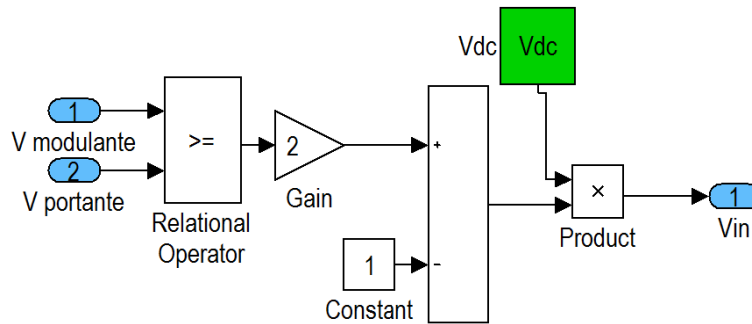


Figura 3. 16: Representación del diagrama de bloques de Simulink de la señal moduladora.
Elaborado por: Autor.

La parte de la carga es absorbida íntegramente por la analizada en el inversor de medio puente ya que no sufre variaciones tal como se muestra en el diagrama de bloques de la figura 3.17.

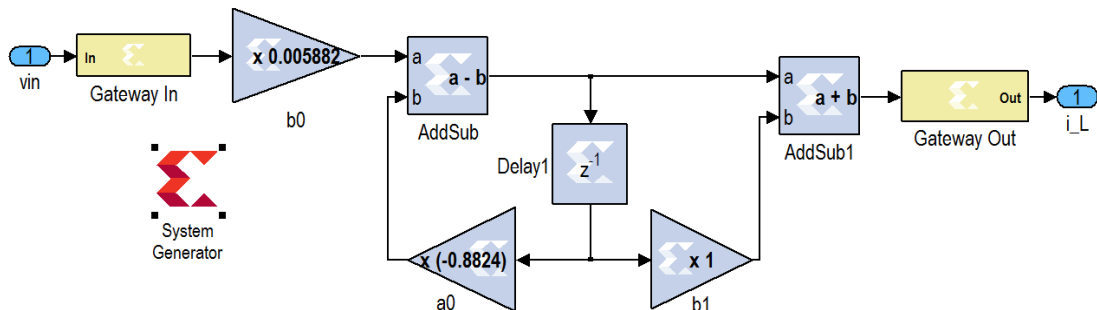


Figura 3. 17: Representación del diagrama de bloques en Simulink.
Elaborado por: Autor.

La figura 3.18 muestra la tendencia de la corriente de salida del inversor de puente. Se puede ver que trabajando con voltaje de entrada constante es

posible obtener una corriente alterna en la salida muy cercana a una señal sinusoidal, si se incrementa el valor de la inductancia se logra obtener una señal más sinusoidal. Con el inversor de puente es posible tener un valor de voltaje de salida doble (100 V de amplitud) que el del inversor de medio puente (50 V de amplitud) (véase la figura 3.13 y compararla con la figura 3.18).

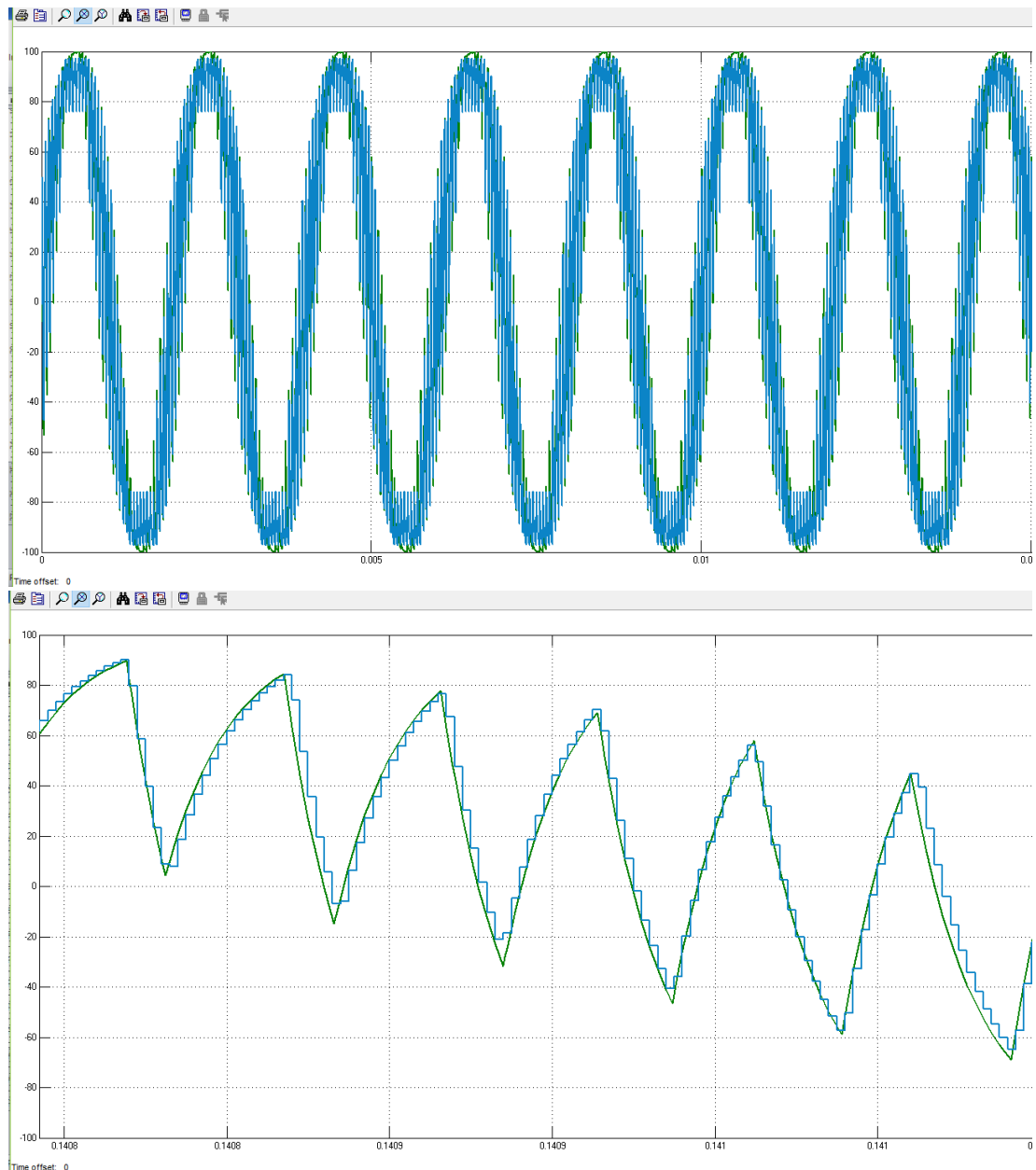


Figura 3. 18: Diagrama de corriente de salida para inversores de puente.
Elaborado por: Autor.

3.6. Aplicación práctica 5: Simulación de Control de retroalimentación de un inversor de puente.

En este último ejemplo se enfocará en observar la realización de un sistema más completo, compuesto por un regulador PI (Proporcional-Integral)

y un inversor monofásico, teniendo la parte de la carga construida a partir de los bloques System Generator de Xilinx, que trabajaba en tiempo discreto. El sistema simularía un convertidor de control de corriente con retroalimentación unitaria.

Antes de ver el diagrama de bloques teórico del sistema, se requiere hacer algunas hipótesis simplificadoras; se supone que tiene una señal inicial ideal, en este caso un generador de corriente que tiene una señal de paso en el instante de tiempo de 0.15 s, y que va de 0 a 2 A. También se supone que el regulador PI no tiene el error de estado estable del lazo y para que sea estable, se tiene un peso que es al menos 20 veces menor que la frecuencia de muestreo y un margen de fase que puede estar entre 50 y 70.

Se eligió un regulador PI porque garantiza una acción reguladora con buenas características, manteniendo la complejidad de su implementación digital en niveles aceptables. También se asume que tanto el sistema como el regulador no tienen retardos adicionales (relacionados con filtros de señal, muestreo, tiempos de procesamiento). La figura 3.19 muestra el diagrama de bloques teórico del modelo del regulador PI a diseñarse en Simulink.

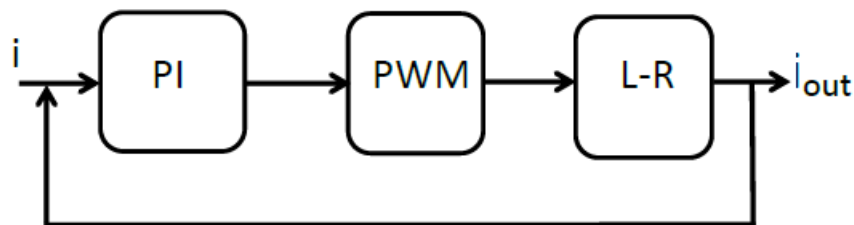


Figura 3. 19: Esquema teórico del modelo a crear en Simulink.
Elaborado por: Autor.

A través de Simulink fue posible realizar el siguiente diagrama de bloques con retroalimentación uniforme. El PI se obtiene de la transformada del sistema a controlar, que no es más que una ecuación del filtro de primer orden:

$$P(s) = \frac{1}{Ls + R}$$

del cual se deriva $M = \frac{1}{|P(j\omega_t)|}$; $e \varphi = m_f - (\angle(P(s) - 180)$;

con lo cual podemos obtener $K_p = \cos(\varphi)M$; $K_I = -\text{sen}(\varphi) M \omega_t$;

Obteniendo como ecuación del regulador $PI C(s) = K_p + \frac{K_I}{s}$;

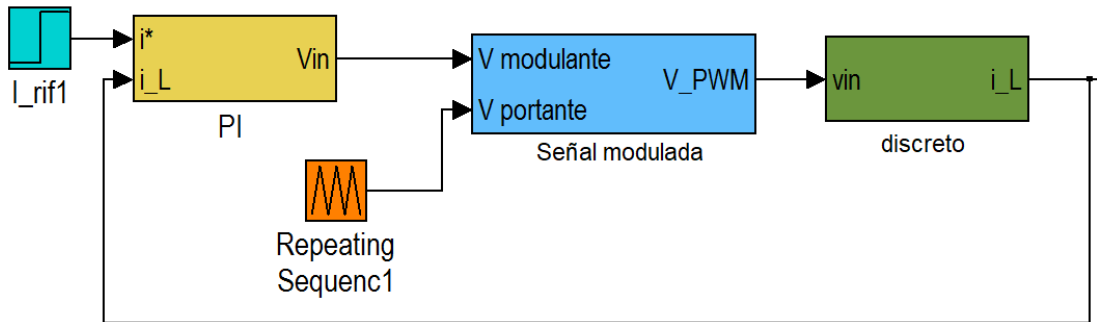


Figura 3. 20: Diseño del filtro FIR paso de banda usando Simulink.
Elaborado por: Autor.

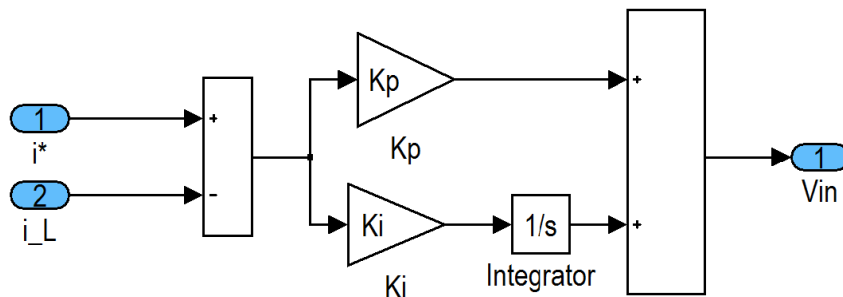


Figura 3. 21: Controlador PI.
Elaborado por: Autor.

La parte del inversor puente se analizó en el ejemplo anterior. Cabe recordar que para no tener saturación en el PWM al comparar la señal moduladora con la señal portadora (triangular), es necesario ajustar la señal moduladora multiplicándola por una ganancia muy pequeña (se eligió $2V_{DC}$ como el valor) y también en la parte de discretización del filtro LR fue necesario agregar un bloque de retardo que va a retardo de un ciclo de reloj para que sea posible realizar todas las operaciones lógicas.

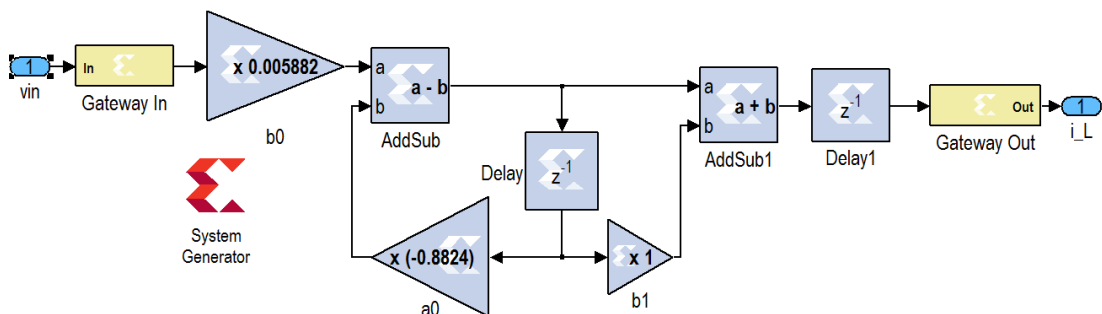


Figura 3. 22: Carga L-R.
Elaborado por: Autor.

A partir de una señal de entrada continua, se obtuvo una señal alterna en la salida que, gracias al controlador PI, se logró acelerar la respuesta del sistema.

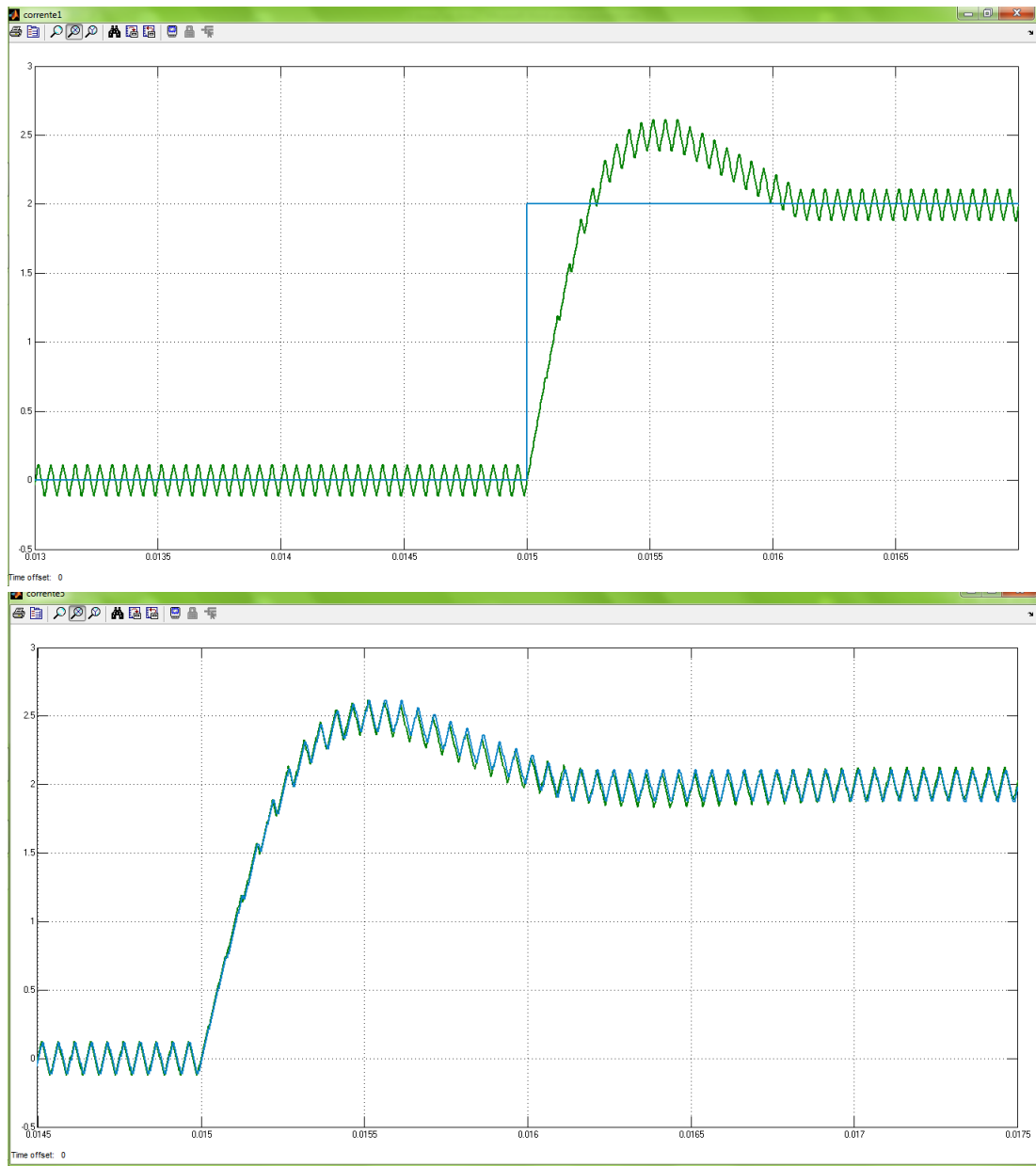


Figura 3. 23: Señal de salida discreta.
Elaborado por: Autor.

Conclusiones.

- La simulación en tiempo real es una alternativa válida a la simulación en la que es necesario tener el prototipo ya construido para poder probarlo. Poder trabajar en un banco de pruebas con componentes virtuales sin duda tiene un impacto muy positivo en la fase de creación de prototipos, ya que permite probar en paralelo tanto la parte de hardware como de software de un sistema.
- La electrónica de potencia se orienta hacia la simulación en tiempo real utilizando diversas soluciones para plataformas Hardware-in-the-Loop, principalmente por necesidades del mercado. De esta forma se logró reducir la duración de las pruebas, manteniendo un alto nivel de seguridad durante las pruebas, todo ello con un coste de simulación y pruebas muy asequible tanto para empresas como para centros de investigación.
- El entorno de desarrollo de Xilinx System Generator, propuesto en el capítulo 2, encaja muy bien en su uso para simulación en tiempo real y, en consecuencia, para Hardware en bucle.
- En las aplicaciones propuestas se puede ver cómo es posible, a partir de la representación matemática de los diagramas dinámicos de algunos componentes de la electrónica de potencia, derivar los componentes del modelado, que son fundamentales para realizar pruebas para una simulación en tiempo real satisfactoria.

Recomendaciones

- Incentivar a los estudiantes en el modelado matemático y simulación de la electrónica de potencia en específico los inversores que son muy utilizados en sistemas de paneles solares fotovoltaicos.
- Análisis comparativo de convertidores Buck y Boost utilizando la tarjeta programable FPGA mediante entorno de simulación Xilinx.

Bibliografía

- Aguirre D., N. (2013). *Implementación de un sistema de detección de señales de tráfico mediante visión artificial basado en FPGA* [Proyecto Fin de Carrera, Universidad de Sevilla]. <https://biblus.us.es/bibing/proyectos/abreproy/12112/>
- Doménech A., G., Garcerán H., V., Hinojosa J., J., López A., J. A., Martínez-Cabeza de Vaca A., J., Villó P., I., & Zapata P., J. (2019). *Circuitos y funciones electrónicas* (Universidad Politécnica de Cartagena, UPCT). CRAI Biblioteca.
- Ellis, G. (2012). Model Development and Verification. En *Control System Design Guide* (pp. 261–282). Elsevier. <https://doi.org/10.1016/B978-0-12-385920-4.00013-8>
- Fuente Alba, E. (2011). *La importancia de la Electrónica de Potencia—Electricidad*. <https://www.revistaei.cl/reportajes/la-importancia-de-la-electronica-de-potencia/>
- González, M. L. (2015). *Dispositivos electrónicos*. D - Editorial de la Universidad Nacional de La Plata. <http://public.ebookcentral.proquest.com/choice/publicfullrecord.aspx?p=4499427>
- Rodríguez Molina, A. (2018). *Diseño, fabricación y validación de fuentes de alimentación* [Info:eu-repo/semantics/bachelorThesis, E.T.S.I. Diseño Industrial (UPM)]. <https://oa.upm.es/49659/>
- The MathWorks, Inc. (2016, septiembre). *MATLAB - El lenguaje del cálculo técnico—MathWorks España*.

http://es.mathworks.com/products/matlab/index.html?s_tid=gn_loc_dr

op



DECLARACIÓN Y AUTORIZACIÓN

Yo, **Vega Freire, Hugo Miguel** con C.C: # 070486225-9 autor del Trabajo de Titulación: **Implementación de aplicaciones prácticas en electrónica de potencia utilizando los entornos de simulación Xilinx y Simulink** previo a la obtención del título de **INGENIERO EN ELÉCTRICO MECÁNICO** en la Universidad Católica de Santiago de Guayaquil.

1.- Declaro tener pleno conocimiento de la obligación que tienen las instituciones de educación superior, de conformidad con el Artículo 144 de la Ley Orgánica de Educación Superior, de entregar a la SENESCYT en formato digital una copia del referido trabajo de titulación para que sea integrado al Sistema Nacional de Información de la Educación Superior del Ecuador para su difusión pública respetando los derechos de autor.

2.- Autorizo a la SENESCYT a tener una copia del referido trabajo de titulación, con el propósito de generar un repositorio que democratice la información, respetando las políticas de propiedad intelectual vigentes.

Guayaquil, 20 de septiembre de 2021

f. Hugo Vega

Nombre: Vega Freire, Hugo Miguel

C.C: 070486225-9

REPOSITORIO NACIONAL EN CIENCIA Y TECNOLOGÍA

FICHA DE REGISTRO DE TESIS/TRABAJO DE TITULACIÓN

TÍTULO Y SUBTÍTULO:	Implementación de aplicaciones prácticas en electrónica de potencia utilizando los entornos de simulación Xilinx y Simulink		
AUTOR(ES)	Vega Freire, Hugo Miguel		
REVISOR(ES)/TUTOR(ES)	M. Sc. Romero Paz, Manuel de Jesús		
INSTITUCIÓN:	Universidad Católica de Santiago de Guayaquil		
FACULTAD:	Facultad de Educación Técnica para el Desarrollo		
CARRERA:	Ingeniería en Eléctrico Mecánica		
TÍTULO OBTENIDO:	Ingeniero en Eléctrico Mecánico		
FECHA DE PUBLICACIÓN:	20 de septiembre de 2021	No. DE PÁGINAS:	40
ÁREAS TEMÁTICAS:	Electrónica Analógica, Electrónica de Potencia y Simulación de Circuitos		
PALABRAS CLAVES/ KEYWORDS:	Potencia, Electrónica, Simulación, Discreto, Semiconductores, Inversores		
RESUMEN/ABSTRACT (150-250 palabras):			
<p>En el presente documento se desarrolla el componente práctico del examen complejo denominado "Implementación de aplicaciones prácticas en electrónica de potencia utilizando los entornos de simulación Xilinx y Simulink". En el campo de la electrónica, los sistemas eléctricos y demás, durante la fase de creación de prototipos de un sistema, las simulaciones son necesarias para depurar el código creado y probar los resultados obtenidos como si el sistema funcionara en un entorno real. En este componente práctico se trata de la simulación en tiempo real, siendo uno de los métodos más utilizados en los sistemas HIL y cómo se implementa en la electrónica de potencia. Se estudiaron tres modelos de simulación en tiempo real: continuo, discreto e híbrido. El entorno Xilinx System Generator, es un entorno de desarrollo que permite el diseño de modelos para simulaciones híbridas en tiempo real. Se intentará comprender cómo está estructurado y se darán algunos ejemplos en el campo de la electrónica de potencia.</p>			
ADJUNTO PDF:	<input checked="" type="checkbox"/> SI	<input type="checkbox"/> NO	
CONTACTO CON AUTOR/ES:	Teléfono: +593-9-87261076	E-mail: hugomiguelvegaf@hotmail.com	
CONTACTO CON LA INSTITUCIÓN: COORDINADOR DEL PROCESO DE UTE	Nombre: Palacios Meléndez Edwin Fernando		
	Teléfono: +593-9-67608298		
	E-mail: edwin.palacios@cu.ucsg.edu.ec		
SECCIÓN PARA USO DE BIBLIOTECA			
Nº. DE REGISTRO (en base a datos):			
Nº. DE CLASIFICACIÓN:			
DIRECCIÓN URL (tesis en la web):			