

UNIVERSIDAD SAN FRANCISCO DE QUITO USFQ

Colegio de Ciencias e Ingenierías

Diseño de un Simulador de Tiempo Real para Circuitos Lineales

Santiago Andrés Granda Bravo

Carlos Gonzalo López López

Ingeniería Eléctrica y Electrónica

Trabajo de fin de carrera presentado como requisito
para la obtención del título de
Ingeniero Electrónico

Quito, 13 de mayo de 2020

UNIVERSIDAD SAN FRANCISCO DE QUITO USFQ

Colegio de Ciencias e Ingenierías

**HOJA DE CALIFICACIÓN
DE TRABAJO DE FIN DE CARRERA**

Diseño de un Simulador de Tiempo Real para Circuitos Lineales

**Santiago Andrés Granda Bravo
Carlos Gonzalo López López**

Nombre del profesor, Título académico

Alberto Sánchez, PhD.

Quito, 13 de mayo de 2020

DERECHOS DE AUTOR

Por medio del presente documento certifico que he leído todas las Políticas y Manuales de la Universidad San Francisco de Quito USFQ, incluyendo la Política de Propiedad Intelectual USFQ, y estoy de acuerdo con su contenido, por lo que los derechos de propiedad intelectual del presente trabajo quedan sujetos a lo dispuesto en esas Políticas.

Asimismo, autorizo a la USFQ para que realice la digitalización y publicación de este trabajo en el repositorio virtual, de conformidad a lo dispuesto en el Art. 144 de la Ley Orgánica de Educación Superior.

Nombres y apellidos: Santiago Andrés Granda Bravo

Código: 00124310

Cédula de identidad: 1720742426

Lugar y fecha: Quito, 13 de mayo de 2020

Nombres y apellidos: Carlos Gonzalo López López

Código: 00122777

Cédula de identidad: 1716495930

Lugar y fecha: Quito, 13 de mayo de 2020

ACLARACIÓN PARA PUBLICACIÓN

Nota: El presente trabajo, en su totalidad o cualquiera de sus partes, no debe ser considerado como una publicación, incluso a pesar de estar disponible sin restricciones a través de un repositorio institucional. Esta declaración se alinea con las prácticas y recomendaciones presentadas por el Committee on Publication Ethics COPE descritas por Barbour et al. (2017) Discussion document on best practice for issues around theses publishing, disponible en <http://bit.ly/COPETHeses>.

UNPUBLISHED DOCUMENT

Note: The following capstone project is available through Universidad San Francisco de Quito USFQ institutional repository. Nonetheless, this project – in whole or in part – should not be considered a publication. This statement follows the recommendations presented by the Committee on Publication Ethics COPE described by Barbour et al. (2017) Discussion document on best practice for issues around theses publishing available on <http://bit.ly/COPETHeses>.

RESUMEN

El siguiente documento presenta el desarrollo de un circuito simulador en tiempo real de circuitos lineales de hasta diez nodos. El documento inicia con una introducción a los conceptos básicos y técnicas de análisis para circuitos lineales y su descripción en Verilog. El trabajo además presenta la síntesis y su simulación y un análisis comparativo con los resultados obtenidos con PSpice.

Palabras clave: Tiempo Real, simulación en tiempo real, overruns, circuitos lineales, Algoritmo MNA, método numérico de Euler.

ABSTRACT

The following document presents the design and development of a real-time simulator for linear circuits of up to ten nodes. The document presents the required concepts related to linear circuit analysis and its associated numerical methods. Then, a conceptual design and its implementation in Verilog is presented. Finally, pre- and post-synthesis simulation results are presented and compared with PSpice simulation results.

Key words: Real-Time, Real-Time simulation, overruns, linear circuits, MNA algorithm, Euler method.

TABLA DE CONTENIDO

1.	INTRODUCCIÓN	10
2.	ANTECEDENTES	13
	2.1 Análisis de circuitos lineales	13
	2.2 Método de Euler	14
3.	DISEÑO CONCEPTUAL	16
	Introducción al diseño conceptual	16
	3.1 Unidad de procesamiento de información	17
	3.2 Unidad de control	22
4.	DESCRIPCIÓN EN VERILOG HDL Y SÍNTESIS	24
5.	CASOS DE ESTUDIO Y RESULTADOS	27
	5.1 Casos de estudio	27
	5.1.1 Circuito RC.....	27
	5.1.2 Circuito de 4 nodos.....	28
	5.1.3 Circuito de 10 nodos.....	30
	5.2 Resultados.....	32
	5.2.1 Caso de estudio 1: circuito RC.	33
	5.2.2 Caso de estudio 2: circuito de 4 nodos.	34
	5.2.3 Caso de estudio 3: circuito de 10 nodos.	35
6.	DISCUSIÓN	38
7.	CONCLUSIONES	44
8.	REFERENCIAS BIBLIOGRÁFICAS	45
9.	ANEXO A: Reporte de Utilización de Recursos de Síntesis	46

ÍNDICE DE TABLAS

Tabla 1. Valores RMSE obtenidos a partir de los valores experimentales y teóricos. ...	41
--	----

ÍNDICE DE FIGURAS

Figura 1. Ejemplo de circuito de general para aplicar MNA.....	14
Figura 2. Unidad de procesamiento de información.....	18
Figura 3. Diseño conceptual del módulo Unidad de Control.	23
Figura 4. Reporte de utilización de recursos: Slice Logic.....	25
Figura 5. Reporte de utilización de recursos: Primitives.....	25
Figura 6. Reporte de utilización de recursos: Black Boxes.....	26
Figura 7. Diagrama esquemático circuito RC.	27
Figura 8. Diagrama esquemático circuito de 4 nodos.	29
Figura 9. Diagrama esquemático circuito de 10 nodos.	30
Figura 10. Gráfico teórico-experimental de voltaje del caso de estudio 1.	33
Figura 11. Gráficos experimentales-teóricos de voltajes y corrientes: caso de estudio 2.	34
Figura 12. Gráficos experimentales-teóricos de voltajes del nodo 2 al 5: caso de estudio 3.	35
Figura 13. Gráficos experimentales-teóricos de voltajes del nodo 6 al 9: caso de estudio 3.	36
Figura 14. Gráficos experimentales-teóricos del voltaje en el nodo 10 y corrientes: caso de estudio 3.....	37
Figura 15. Análisis de tiempo de simulación.	39

1. INTRODUCCIÓN

Los programas de simulación son ampliamente utilizados para realizar distintos estudios, desde el comportamiento de circuitos integrados hasta estudios de dispositivos eléctricos de potencia (Matar, et al., 2005). Como explican Matar, et al. (2005), los estudios realizados en programas simuladores poseen una gran desventaja que radica en la no operación en tiempo real, esto se debe a que los programas son ejecutados a manera de software sobre un CPU (Unidad de procesamiento central) para realizar los cálculos de los modelos. Esto causa que el tiempo efectivo en obtener una solución corresponda con la verdadera velocidad con la que responde el circuito. Cabe destacar que la realización de simulaciones de circuitos o sistemas mediante un software es una herramienta muy útil debido a su versatilidad; sin embargo, es en realidad una tecnología que se empezó a desarrollar en los años 70 con el advenimiento de los computadores y en particular de los computadores personales. Antes de esto, el estudio de circuitos y sistemas se realizaban utilizando computadores analógicos, los cuales reproducían la respuesta de sistemas mediante la resolución de ecuaciones diferenciales lineales y no lineales utilizando amplificadores operacionales.

En la actualidad, debido al extensivo desarrollo de tecnologías de hardware digital, y su rápido y relativamente barato prototipaje e implementación mediante plataformas como los FPGA (Matriz de Puertas Lógicas Programable en Campo), es posible realizar simulaciones en tiempo real de la respuesta de circuitos y sistemas tan grandes y complejos como un sistema de distribución o transmisión.

De acuerdo con la explicación de Bélanger, et al. “Una simulación es la representación de las operaciones o características de un sistema a través del uso u operación de otro.” (p. 37). Para el caso de la simulación en tiempo real, éstas son un tipo de simulación digital en la cual el tiempo de simulación es discretizado de forma

constante permitiendo así que el tiempo de simulación avance a la par o incluso más rápido que la respuesta real del sistema (Bélanger, et al., 2005).

En el caso del sistema a diseñar se plantea una descripción de hardware que implemente un circuito simulador en tiempo real para resolver circuitos lineales usando un método numérico que se ejecuta en tiempo real. Con la guía de los conceptos presentados previamente, se implementa un método numérico iterativo que computa un resultado durante un intervalo de tiempo fijo (que coincide con el avance real del tiempo) y en cada nuevo paso definido de tiempo se calcula un nuevo valor numérico y así sucesivamente hasta cumplir con la condición final definida en el diseño.

Para el desarrollo de este proyecto se han definido un conjunto de especificaciones límites. En particular, este sistema permite la simulación en tiempo real de circuitos lineales con un máximo de diez nodos. Por este motivo el Hardware fue diseñado para resolver sistemas de ecuaciones diferenciales con un máximo de dieciséis incógnitas. Así mismo, este trabajo inicia con la presentación de conceptos de análisis nodal los cuales permitan extraer de forma algorítmica las ecuaciones diferenciales que describen el comportamiento de los circuitos lineales.

La resolución de las ecuaciones se realiza utilizando el método de integración numérica de Euler. Luego, se propone la realización conceptual mediante diagrama de bloques del sistema digital empleando los conceptos de modularidad, regularidad y jerarquía. Esta realización conceptual conduce a la descripción en hardware de cada uno de los módulos utilizando Verilog. A continuación, se realiza la síntesis, la que permite interpretar la descripción en HDL (Lenguaje de Descripción de Hardware) y su realización a nivel de RTL (Lenguaje de Transferencia de Registros) o de FPGA.

Una vez culminada la síntesis se analizan los recursos que se utilizan en caso de que se desee implementar en un FPGA, y se realiza la simulación digital para comprobar el correcto funcionamiento del diseño.

Como casos de estudio se analiza el comportamiento del hardware para el análisis de la respuesta de circuitos ante una excitación DC (corriente directa), de los siguientes ejemplos: un circuito RC (resistencia en serie con capacitor), un circuito con cuatro nodos y finalmente un circuito de diez nodos; se realiza la comparación de las curvas de respuesta con los resultados producidos en PSpice. Se analiza también las restricciones de tiempo de los resultados computados entre cada iteración.

En las siguientes secciones se presenta de forma detallada cada punto mencionado. Cabe señalar que el proceso de descripción de hardware, síntesis y simulación se realizó en Vivado de Xilinx.

2. ANTECEDENTES

2.1 Análisis de circuitos lineales

Para un estudio general de circuitos RLC (resistencias, inductancias y capacitancias) es conveniente utilizar el método denominado Análisis Nodal Modificado (MNA). Dicho método nodal permite describir matemáticamente el circuito en estudio, logrando obtener de esta forma los valores de voltaje y corriente en los puntos deseados. De manera general el análisis nodal modificado procura describir un circuito mediante variables de estado con la siguiente forma.

$$M \dot{x}(t) = -Gx(t) + Pu(t) \quad (1)$$

$$Y(t) = Qx(t) \quad (2)$$

En donde

$$x(t) = \begin{bmatrix} V(t) \\ I(t) \end{bmatrix} \quad M = \begin{bmatrix} C & 0 \\ 0 & L \end{bmatrix} \quad G = \begin{bmatrix} G_1 & W^T \\ -W & 0 \end{bmatrix}$$

La matriz $x(t)$ está compuesta por I y V que son las incógnitas dependientes del tiempo. La matriz M se forma con los valores de los capacitores y los inductores del circuito, finalmente la matriz G contiene los elementos de conductancia $G_1 = \frac{1}{R}$, y la matriz de adyacencia W la cual se obtiene siguiendo la siguiente regla (Qin, Z., & Cheng, C. K., 2009, p.22).

$$W = [a_{ij}]_{n \times b} \quad \text{donde } a_{ij} = \begin{cases} 1 & \text{si } e_j \text{ empieza desde } v_i \\ -1 & \text{si } e_j \text{ termina en } v_i \\ 0 & \text{otro caso} \end{cases}$$

A continuación, se presenta un circuito con cuatro nodos que sirve de ejemplo para aplicar este método.

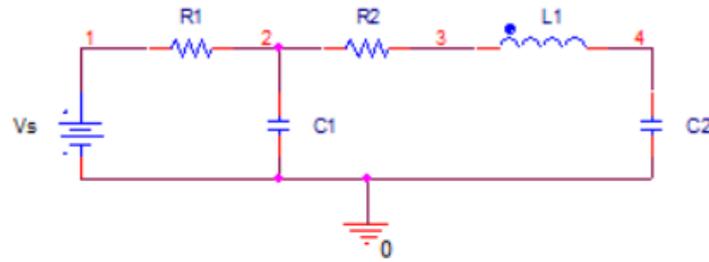


Figura 1. Ejemplo de circuito de general para aplicar MNA.

Según Qin., et al (2009), se debe realizar los siguientes pasos para obtener las ecuaciones:

Paso 1: escoger como nodo de referencia a la tierra.

Paso 2: crear variables para las corrientes de los inductores, definiendo una dirección.

Paso 3: para los N nodos se debe establecer ecuaciones usando KCL, ley de corrientes de Kirchhoff, que representan las corrientes de rama de elementos RC en términos de voltajes y variables de corrientes que se definieron en el paso 2.

Paso 4: establecer la ecuación de la rama de las inductancias en forma diferencial, $v_L = L \frac{di_L}{dt}$, usando las variables predefinidas de corriente y voltaje.

Aplicando los pasos mencionados se obtiene el siguiente sistema de ecuaciones diferenciales con forma de la ecuación (1).

$$\begin{bmatrix} C_1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & C_2 & 0 \\ 0 & 0 & 0 & L_1 \end{bmatrix} \begin{bmatrix} \dot{v}_2 \\ \dot{v}_3 \\ \dot{v}_4 \\ \dot{i}_1 \end{bmatrix} = - \begin{bmatrix} \frac{1}{R_1} + \frac{1}{R_2} & -\frac{1}{R_2} & 0 & 0 \\ -\frac{1}{R_2} & \frac{1}{R_2} & 0 & 1 \\ 0 & 0 & 0 & -1 \\ 0 & -1 & 1 & 0 \end{bmatrix} \begin{bmatrix} v_2 \\ v_3 \\ v_4 \\ i_1 \end{bmatrix} + \begin{bmatrix} \frac{1}{R_1} v_s \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

2.2 Método de Euler

Como se observó al usar MNA se genera un sistema de ecuaciones diferenciales por lo tanto es necesario utilizar un método numérico que permita resolver dicho sistema.

El método de Euler presenta una complejidad de implementación y uso aceptable en comparación con otros métodos y sus resultados se aproximan bastante bien a un modelo de referencia según la literatura comúnmente disponible.

El método de Euler es conocido porque permite resolver ecuaciones diferenciales dado un valor de condición inicial, debido a que éste es un método de integración numérica. Para proporcionar una solución numérica con un valor de condición inicial se decide a criterio propio el intervalo en que se desea obtener la solución. Al intervalo se lo divide en longitudes H y usando el valor de condición inicial para la primera iteración se calcula el resto de las soluciones (Chapra, S. C., & Canale, R. P., 2010, p.708). La expresión general del método de Euler para una sucesión de aproximaciones se muestra a continuación.

$$\begin{aligned}
 y_1 &= y_0 + Hf(x_0, y_0) \\
 y_2 &= y_1 + Hf(x_1, y_1) \\
 &\cdot \\
 &\cdot \\
 &\cdot \\
 y_{n+1} &= y_n + H f(x_n, y_n)
 \end{aligned} \tag{3}$$

Considerando las expresiones matemáticas del análisis por MNA y aplicando el método de Euler para resolver sistemas de ecuaciones diferenciales se obtuvo la siguiente expresión iterativa que permite la solución de circuitos lineales.

$$\overrightarrow{x(t+H)} = \overrightarrow{x(t)} + H * \overrightarrow{M^{-1}} * (-\overrightarrow{G} * \overrightarrow{x(t)} + \overrightarrow{PU}) \tag{4}$$

3. DISEÑO CONCEPTUAL

Introducción al diseño conceptual

Para conceptualizar, comprender y diseñar correctamente el Circuito Simulador en Tiempo Real para circuitos lineales; es menester realizar un diseño conceptual en primer lugar que permita concebir todas las operaciones y funciones matemáticas que deben ser ejecutadas, para posteriormente diseñar todos los módulos que implementen dichas operaciones y resolver de esta forma los sistemas de ecuaciones diferenciales que describan los circuitos lineales que se deseen simular. Conociendo las funciones que deben ser implementadas por cada módulo y siguiendo las guías de diseño digital que presentan Harris & Harris (2016), se procede a dividir el módulo principal de mayor jerarquía en módulos más pequeños, los cuales pueden ser replicados y reutilizados para implementar en este caso un método numérico que resuelva ecuaciones diferenciales de primer orden. Los submódulos que componen el módulo de mayor jerarquía tienen funciones, entradas y salidas bien definidas, siendo así que al momento de interconectar todos los submódulos el flujo de datos no se pierde evitando obtener resultados erróneos. De esta manera, la realización del proyecto busca abstraer y reducir su complejidad siguiendo los principios de Modularidad, Regularidad y Jerarquía.

El diseño conceptual presenta y define todos los módulos que van a formar parte del módulo de mayor jerarquía que permitirá resolver las ecuaciones diferenciales que describen a los circuitos lineales aplicando el método numérico de Euler. En la conceptualización del diseño se muestran todos los módulos que componen al simulador, definiendo sus respectivas entradas y salidas. La solución propuesta obedece a una arquitectura con dos componentes principales: flujo de información (datapath) y control.

3.1 Unidad de procesamiento de información

La Figura 2 muestra el diagrama conceptual del flujo de información que se propone. Las funciones de cada módulo y las operaciones que se realizan en cada etapa del diseño conceptual se explican inmediatamente.

La Figura 2 muestra el diseño conceptual que se siguió rigurosamente para describir el Hardware que implemente el método numérico vectorial de Euler. Se puede observar que las dos entradas que rigen el funcionamiento del circuito son las señales de reset y de clk (señal de reloj de la tarjeta en la que se desee implementar). Las dos señales entran a un módulo divisor de frecuencia en el cual la señal de reloj de entrada de 50 [MHz] (frecuencia definida) se divide hasta que alcance una frecuencia que permita obtener los resultados finales de cada iteración en un tiempo que no sobrepase el periodo de tiempo establecido como nuestro H y tiene un valor de 1.66667×10^{-5} [s], este periodo se toma como la unidad de tiempo para la cual se diseñó el simulador. El valor de H representa el intervalo con el que se va a muestrear unidad real de tiempo. La nueva señal de reloj se conecta a una máquina de estados finitos, (FSM), que permite extraer los datos de las matrices almacenados en memoria no volátil (NVM), con los nombres correspondientes a cada una de las matrices generadas por el Algoritmo MNA.

Utilizando MNA y el método presentado en el texto de Qin & Cheng (2009), se obtienen las matrices M que contiene los valores de capacitancias e inductancias, G con los valores de conductancias y vectores de adyacencia, y el vector de constantes PU, del circuito que se desee simular. Los datos de las matrices se almacenaron columna a columna en memoria NVM. Todas las operaciones de extracción de datos en las memorias suceden de forma simultánea debido a la configuración del diseño porque están dispuestos en paralelo como se puede ver en la Figura 2. Para obtener todos los datos de las matrices en paralelo y poder realizar las operaciones matriciales necesarias, las salidas de las memorias NVM se conectaron a demultiplexores de 1 entrada y 16 salidas. Estos demultiplexores obtienen y ordenan la información de entrada en cada salida del módulo de acuerdo con la señal de selección, misma que se genera en la FSM de lectura de datos.

Debido a que las ecuaciones poseen 16 incógnitas, las matrices que contienen los datos de los circuitos son matrices cuadradas de 16 filas y 16 columnas. Se inserta un registro en todas las salidas de los demultiplexores, este proceso se realiza por cada pieza de información correspondiente a las columnas de cada matriz y de esta forma los elementos de las matrices almacenadas en memorias se mantienen constantes en los registros y se encuentran listos para que se realicen las operaciones matriciales correspondientes.

Los elementos de la matriz M a la salida de los registros pasan a ser invertidos en el módulo M_{inv} , el cual realiza la operación $1/M$ de 32 bits usando un bloque de propiedad intelectual IP propio de Xilinx, que ejecuta la inversión de la matriz en punto flotante.

Cuando la inversión de la matriz M se completa pasa a multiplicarse por el valor de H constante o tiempo de muestreo que se definió para este problema y se mencionó previamente de 1.66667×10^{-5} . Para realizar la multiplicación se almacenó el valor constante de H en una memoria que se observa en la Figura 2 que tiene el nombre de “ROM constante H ”. La multiplicación se realiza en el módulo que tiene el nombre “ $H * M_{inv}$ ”, el cual utiliza un bloque IP que realiza operaciones matemáticas en punto flotante, para este caso la multiplicación de 2 variables de 32 bits en punto flotante y obtiene los resultados de la multiplicación de $H * M_{inv}$ en 32 bits. Los datos de la matriz G se almacenan en memoria NVM.

Se observa en la Figura 2 el módulo con el nombre “ROM Matriz G ”, son extraídos y siguen el mismo proceso de ubicación de datos en paralelo y almacenado en registros. Una vez que se obtienen los datos en paralelo de la matriz G , estos se multiplican por menos uno, constante que se almacena en memoria NVM en el módulo “ROM Constante 1”, igualmente se utiliza el bloque IP para realizar operaciones

matemáticas en punto flotante, en el módulo que se llama “Multiplicación $-1 * G$ ” en la Figura 2.

El vector de condiciones iniciales es un vector de ceros $x(0) = x[0]$, que se almacena en una memoria NVM con el nombre “ROM $x(0)$ C. Iniciales” que guarda los datos de voltajes de nodos y corrientes de inductores en el tiempo 0. De la misma forma que las memorias anteriores esta información se almacena en paralelo en un demultiplexor y cada elemento del vector columna se conecta a un multiplexor de selección de la iteración en la que se encuentra el sistema. La información que sale de los multiplexores se envía a registros para ser almacenada. La información a la salida de cada registro pasa a ser multiplicada por el contenido almacenado en registros de la matriz $-G$, este paso sucede en el módulo “Multiplicación $-G * x(t)$ ” que se puede observar en la Figura 2, el resultado que se obtiene en este módulo es un vector columna de 16 filas, donde cada elemento está expresado en 32 bits. Finalmente, el vector PU se almacena en memoria NVM en el módulo “ROM Vector PU” que puede observarse en la Figura 2.

Del mismo modo que las memorias previas, la información del vector columna PU es procesada en paralelo a través de un demultiplexor y almacenados en registros, uno por cada fila que compone al vector columna PU. Los elementos del vector PU pasan a sumarse con los resultados del vector columna que se obtiene en la multiplicación de $-G * x(t)$.

Una vez que los datos han sido extraídos, paralelizado y computado, los resultados se encuentran listos para operar y resolver el método de Euler cumpliendo la ecuación siguiente:

$$\overrightarrow{x(t+H)} = \overrightarrow{x(t)} + H * \overrightarrow{M^{-1}} * (-\overrightarrow{G} * \overrightarrow{x(t)} + \overrightarrow{PU}) \quad (5)$$

La ecuación 5 muestra el método numérico de Euler. Donde $x(t)$ es el vector que almacena los voltajes de nodos de cada iteración, H es el valor constante del tiempo de muestreo del vector de tiempo y que tiene relación directa con la señal de reloj de la plataforma de hardware, M es la matriz diagonal que contiene los valores de capacitancias e inductancias, $-G$ es la matriz que contiene los valores de conductancias del circuito y los vectores de Adyacencia del circuito, y PU es el vector de constantes.

Con las operaciones que han sido mencionadas se tiene el flujo de datos de los módulos $H * M_{inv}$, $-G * x(t)$, $x(t)$ y PU . Todas las operaciones vectoriales y matriciales se hacen siguiendo los principios básicos de operaciones algebraicas vectoriales.

El resultado final es un vector de 16 filas y 1 columna, cada uno de 32 bits, que corresponden a los voltajes de nodos y corrientes de inductores que se plantearon en las ecuaciones diferenciales que describen al circuito. Los 16 resultados de la iteración son almacenados en 16 registros. Cuando estos resultados son computados se retroalimentan a las entradas del multiplexor de selección de iteración, y de esta forma el proceso se repite hasta completar 600 iteraciones, que son el número máximo de iteraciones para el cual se diseñó el simulador en tiempo real.

3.2 Unidad de control

En la Figura 3 se muestra el diseño conceptual de la unidad de control la misma que controla el flujo de información.

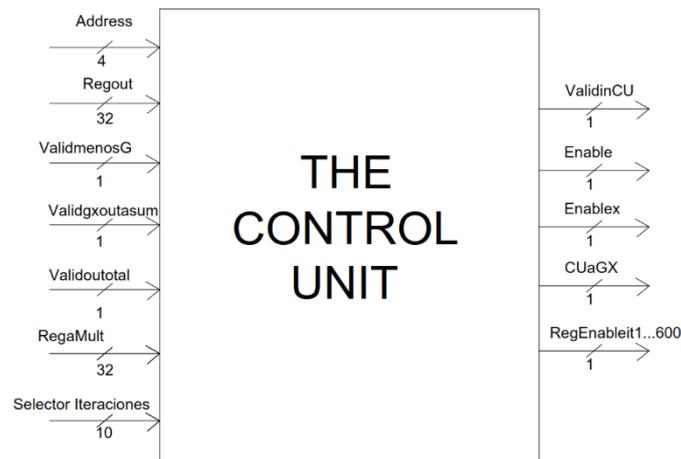


Figura 3. Diseño conceptual del módulo Unidad de Control.

La unidad de control cuenta con siete entradas que son: Address de los demultiplexores, la salida de los registros de la memoria M llamados Regout, la salida que indica la validez del dato del módulo IP Multiplicación $-1 * G$ y se llama ValidmenosG, la salida que indica la validez del dato del módulo IP Multiplicación $-G * x(t)$ que tiene el nombre de Validgxoutasum, la salida que indica la validez del dato del módulo IP Suma Final que se llama Validouttotal, la salida del primer registro del vector $x(t)$ de 32 bits, y la salida del módulo contador de iteraciones que genera la señal de selección de los multiplexores de selección de iteración. Las salidas generadas por la unidad de control son: ValidinCU señal que activa al módulo MInv y el módulo Multiplicación $-1 * G$, los enables de los registros de los datos de las matrices M y G con nombre Enable, los enables de los registros de los datos del vector $x(t)$ que tienen el nombre de Enablex, los enables de los registros finales de cada iteración RegEnableit1...600, y la señal de activación del módulo Multiplicación $-G * x(t)$ que tiene el nombre CUaGX.

El módulo unidad de control funciona de la siguiente forma, cuando el Address de los demultiplexores es diferente al número 15, la señal de Enable de los registros que contienen la información de las matrices M y G y la señal de Enablex de los registros con los elementos del vector $x(t)$ se mantienen apagadas, cuando el Address alcanza la última

dirección de datos del demultiplexor con el número 15, los registros mencionados escriben datos, una vez que los registros que almacenan la información de M han guardado sus datos, la señal de ValidinCU se activa, el módulo MInv y a su vez el módulo $-1 * G$ inician su funcionamiento. Cuando la señal ValidmenosG es igual a 1, la señal CUaGX se activa iniciando en ese momento las operaciones que corresponden al método numérico de Euler.

Para determinar que la iteración ha culminado la señal de Validouttotal se activa cambiando de 0 a 1. Para el proceso de escritura de los resultados en los registros finales correspondientes a cada iteración, la señal RegEnableit1...600 se activa con el valor numérico en el que se encuentra el selector de iteraciones.

Para iniciar la nueva iteración, la salida de los registros finales se realimenta al multiplexor de selección de iteraciones. Cuando la salida Regamult toma el valor de la última iteración obtenida, la señal CUaGX se activa nuevamente y el proceso del método de Euler inicia nuevamente, y de esta forma se repite por las 600 iteraciones en las que se diseñó el Hardware que se muestra en la Figura 2.

4. DESCRIPCIÓN EN VERILOG HDL Y SÍNTESIS

Siguiendo el diseño conceptual se procede con el siguiente paso que es describir el hardware que implemente los módulos y posteriormente realizar la síntesis del diseño. En esta etapa se describió el hardware en Verilog de los módulos de memoria NVM de las constantes H y la constante -1, el módulo divisor de frecuencia, contador de iteraciones, registros, multiplexores y decodificadores. Los módulos que llevan a cabo las operaciones matemáticas en punto flotante se implementaron con el uso de bloques IP.

La síntesis, como lo explican Harris & Harris (2010), se entiende como: “Síntesis lógica es un proceso que transforma código HDL en un netlist que describe el hardware. El

sintetizador lógico realiza optimizaciones que reducen la cantidad de hardware requerido.” (p. 176). Dicho esto, después de la síntesis se pueden observar los RTL producidos por la descripción de hardware, y si se requiere de la implementación en un FPGA se puede obtener una idea del hardware que se va a implementar, después de haber descrito el mismo en HDL.

A continuación, se puede observar el reporte de utilización de recursos generado cuando el proceso de síntesis concluye.

```
1. Slice Logic
-----
```

Site Type	Used	Fixed	Available	Util%
Slice LUTs*	93606	0	17600	531.85
LUT as Logic	93606	0	17600	531.85
LUT as Memory	0	0	6000	0.00
Slice Registers	326995	0	35200	928.96
Register as Flip Flop	316650	0	35200	899.57
Register as Latch	10345	0	35200	29.39
F7 Muxes	40559	0	8800	460.90
F8 Muxes	18955	0	4400	430.80

Figura 4. Reporte de utilización de recursos: Slice Logic.

```
7. Primitives
-----
```

Ref Name	Used	Functional Category
FDCE	316643	Flop & Latch
LUT6	91121	LUT
MUXF7	40559	MuxFx
MUXF8	18955	MuxFx
LDCE	10345	Flop & Latch
CARRY4	1794	CarryLogic
LUT4	1781	LUT
LUT5	1087	LUT
LUT3	247	LUT
LUT2	206	LUT
BUFG	8	Clock
FDRE	6	Flop & Latch
OBUFF	2	IO
IBUF	2	IO
OBUF	1	IO
FDPE	1	Flop & Latch

Figura 5. Reporte de utilización de recursos: Primitives.

8. Black Boxes

```

-----
+-----+-----+
|           Ref Name           | Used |
+-----+-----+
| Sumador_floating_point_0_0   | 272 |
| menosGxX_floating_point_0_0  | 256 |
| MenosG_floating_point_0_0    | 256 |
| Multiplicacionpaso3_floating_point_0_0 | 16 |
| Inversion_floating_point_0_0 | 16 |
| HxMinv_floating_point_0_0    | 16 |
| MemoriasG_blk_mem_gen_0_0    | 1 |
| MemoriasG_G9_0               | 1 |
| MemoriasG_G8_0               | 1 |
| MemoriasG_G7_0               | 1 |
| MemoriasG_G6_0               | 1 |
| MemoriasG_G5_0               | 1 |
| MemoriasG_G4_0               | 1 |
| MemoriasG_G3_0               | 1 |
| MemoriasG_G2_0               | 1 |
| MemoriasG_G1_2               | 1 |
| MemoriasG_G15_0              | 1 |
| MemoriasG_G14_0              | 1 |
| MemoriasG_G10_3              | 1 |
| MemoriasG_G10_2              | 1 |
| MemoriasG_G10_1              | 1 |
| MemoriasG_G10_0              | 1 |
| MemoriaP_blk_mem_gen_0_0     | 1 |
| MemoriaC_blk_mem_gen_0_0     | 1 |
+-----+-----+

```

Figura 6. Reporte de utilización de recursos: Black Boxes.

En la Figura 4, Figura 5, y Figura 6 se muestran los resultados desplegados en el reporte de utilización de recursos generado post-síntesis. En el reporte de utilización de recursos como lo explica Xilinx (2019), se descompone la utilización de recursos que ocupa el diseño, y los organiza dependiendo del recurso. En la Figura 4 se puede observar los recursos utilizados por el diseño después de haberse corrido la síntesis. En la sección de “Slice Logic” se muestran los recursos que utilizados por el diseño. Estos pueden ser usados por el sintetizador lógico para construir funciones lógicas de cualquier tipo (Xilinx, 2019). Como se puede ver en la Figura 4 el diseño usa: Look Up Tables (LUTS), registros, y multiplexores. Finalmente, en la Figura 5 se observan también los bloques primitivos (bloques básicos utilizados por el sintetizador lógico para la implementación en el FPGA), y el uso específico de cada uno de estos bloques primitivos. Se puede observar en la Figura 6 los black boxes (bloques IP) usados en el diseño, específicamente para realizar las operaciones matemáticas en punto flotante y las memorias en las que se

almacenaron los datos de las matrices que contienen la información de los circuitos que se presentan en la siguiente sección de casos de estudio. Los otros reportes generados en el reporte de utilización se incluyen en la sección de anexos.

5. CASOS DE ESTUDIO Y RESULTADOS

5.1 Casos de estudio

En esta sección se presentan los ejemplos que se decidieron estudiar para observar el comportamiento en simulación del simulador de circuitos lineales. En las siguientes subsecciones se presentan ejemplos para circuitos de un nodo, cuatro nodos y una corriente de inductor. Finalmente, se muestra un ejemplo de un circuito de diez nodos que cuenta con tres inductores con sus respectivas corrientes.

5.1.1 Circuito RC.

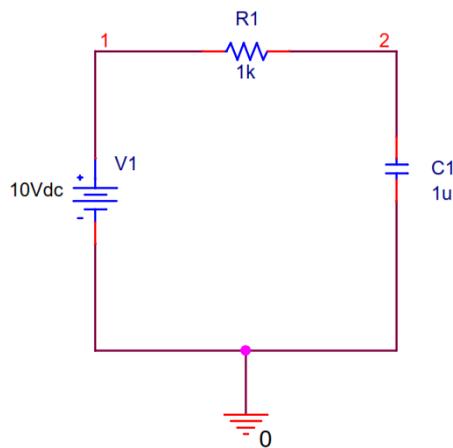


Figura 7. Diagrama esquemático circuito RC.

La ecuación diferencial que describe el comportamiento del circuito de la Figura 7 es:

$$C \cdot \dot{v}_2 = -\frac{1}{R} \cdot v_2 + \frac{v_s}{R} \quad (6)$$

La ecuación número 6 muestra el comportamiento del voltaje en el nodo 2 o voltaje en el capacitor con respecto al tiempo. Donde C representa el valor de capacitancia

del circuito en [F], R es el valor de resistencia del circuito en [Ω] y v_s representa el voltaje de la fuente en [V]. Sustituyendo los valores de resistencia, capacitancia y el valor de la fuente en la ecuación se puede almacenar los datos de las matrices C, G y el vector PU, para ser cargados en los módulos de memoria del simulador y así obtener los datos de voltaje en el nodo 2 con el transcurso del tiempo de muestreo H.

$$\text{Matriz } M = 1 \times 10^{-6}$$

$$\text{Matriz } G = 1 \times 10^{-3}$$

$$\text{Vector } PU = 0.01$$

Con estos valores se obtiene la siguiente ecuación:

$$1 \times 10^{-6} \cdot \dot{v}_2 = -1 \times 10^{-3} \cdot v_2 + 0.01$$

Estos parámetros del circuito se almacenan en las memorias del simulador y se procede a obtener los resultados que se pueden observar en la sección de resultados.

5.1.2 Circuito de 4 nodos.

En este circuito se analiza un ejemplo de circuito que posee cuatro nodos, donde tres nodos tienen sus voltajes desconocidos y un inductor en el que se analiza la corriente que circula por éste. En la figura que se muestra a continuación se indica el siguiente caso de estudio usado para la simulación. En este circuito se presentan las ecuaciones diferenciales en su forma matricial que fueron almacenadas en las respectivas memorias correspondientes a cada matriz.

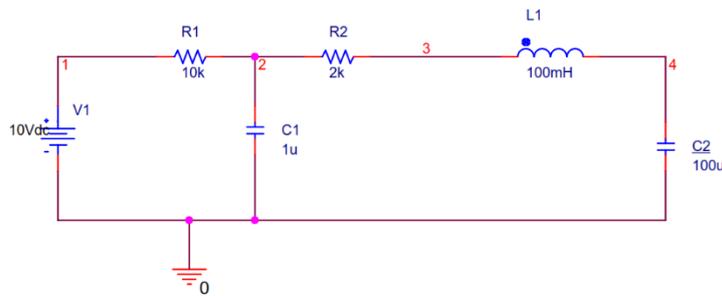


Figura 8. Diagrama esquemático circuito de 4 nodos.

En la Figura 8 se presenta el circuito usado para la extracción de ecuaciones diferenciales y simulación. Se observa que el circuito posee cuatro nodos, en los cuales el voltaje en los nodos 2, 3 y 4 es una incógnita y en el inductor L1 se extrae la corriente que circula por el elemento. A partir del circuito se procedieron a extraer las ecuaciones, usando el algoritmo MNA, las cuales se presentan en forma matricial a continuación.

$$\begin{pmatrix} 1 \times 10^{-6} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 100 \times 10^{-6} & 0 \\ 0 & 0 & 0 & 100 \times 10^{-3} \end{pmatrix} * \begin{pmatrix} v_2 \\ v_3 \\ v_4 \\ i_1 \end{pmatrix} \\ = - \begin{pmatrix} 6 \times 10^{-4} & -5 \times 10^{-4} & 0 & 0 \\ -5 \times 10^{-4} & 5 \times 10^{-4} & 0 & 1 \\ 0 & 0 & 0 & -1 \\ 0 & -1 & 1 & 0 \end{pmatrix} * \begin{pmatrix} v_2 \\ v_3 \\ v_4 \\ i_1 \end{pmatrix} + \begin{pmatrix} 1 \times 10^{-4} \\ 0 \\ 0 \\ 0 \end{pmatrix} * 10$$

El sistema de ecuaciones matriciales muestra y describe el comportamiento del circuito de la Figura 8. En el lado izquierdo de la ecuación se observa la matriz M que se almacena en el diseño en la memoria M, seguido del igual se observa la matriz G que se almacena en la memoria G y finalmente el vector PU que se almacena en la memoria PU. Una vez que estos datos han sido guardados en las memorias se procede a hacer la simulación y extraer resultados del sistema de ecuaciones. Los resultados de la simulación se pueden observar en la sección siguiente.

5.1.3 Circuito de 10 nodos.

En este circuito se analiza un ejemplo que cuenta con 9 voltajes de nodo como variables, 3 inductores que representan 3 incógnitas corrientes de inductores, lo que significa que el sistema de ecuaciones diferenciales para este caso tiene en total 12 ecuaciones diferenciales. En la siguiente figura se puede observar el circuito analizado para este caso de estudio y posteriormente se muestra el sistema de ecuaciones diferencial en su representación matricial.

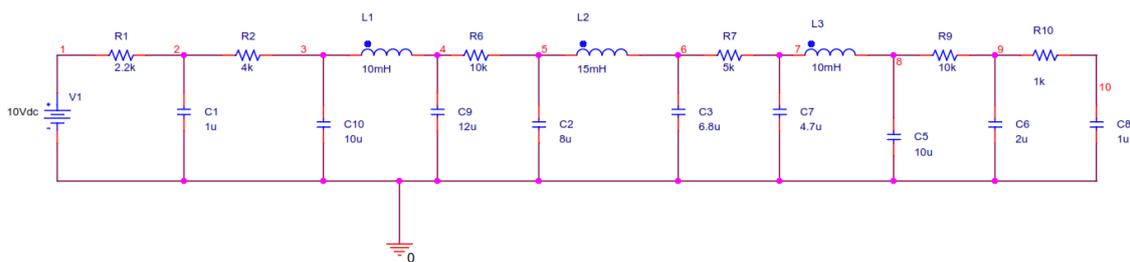


Figura 9. Diagrama esquemático circuito de 10 nodos.

La Figura 9 muestra el ejemplo usado para este caso de estudio. Se puede observar que el ejemplo cuenta con 10 nodos, en los cuales en 9 de estos nodos el voltaje es una incógnita. Se puede observar también que el ejemplo tiene 3 inductores en los cuales es necesario encontrar la corriente que circula por cada uno de ellos. Con esta explicación a continuación se muestran las matrices extraídas del circuito que posteriormente serán guardadas en las memorias M, G y PU. En la siguiente página se observan las matrices obtenidas para el circuito de la Figura 9.

Matriz $M =$

$$\begin{pmatrix} 1E-6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1E-5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1.2E-5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 8E-6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 6.8E-6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 4.7E-6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1E-5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2E-6 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1E-6 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 10E-3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 15E-3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 10E-3 & 0 \end{pmatrix}$$

Matriz $G =$

$$\begin{pmatrix} 7.5E-4 & -2.5E-4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2.5E-4 & 2.5E-4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1E-4 & -1E-4 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & -1E-4 & 1E-4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 2E-4 & -2E-4 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & -2E-4 & 2E-4 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1E-4 & -1E-4 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1E-4 & 1.1E-3 & -1E-3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1E-3 & 1E-3 & 0 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$\text{Vector } PU = \begin{pmatrix} 4.54E - 3 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Una vez que se han extraído las matrices que representan a las ecuaciones diferenciales, se procede a guardarlas en cada una de las memorias correspondientes. Cuando los datos de las matrices han sido almacenados en las memorias, se procede con la simulación y extracción de resultados, los mismos que se encuentran en la sección de resultados.

5.2 Resultados

En esta sección se presentan los resultados que se obtuvieron de cada uno de los casos de estudio que se presentaron anteriormente. Para todos los ejemplos los datos de las matrices se convirtieron a representación binaria de punto flotante usando el estándar IEEE-754 (Hough, 1981). A continuación, se presentan los resultados de simulación de cada uno de los casos de estudio. Estos resultados se presentan gráficamente donde se puede observar el comportamiento de cada uno con respecto al tiempo, en las gráficas se realiza una comparación con los resultados teóricos extraídos de la simulación circuital de PSpice. Para realizar el estudio y análisis del error que demuestren la validez de los datos para cada caso estudiado se utiliza el método de RMSE (Raíz del error cuadrático medio) que se encuentra en la sección siguiente de discusión.

5.2.1 Caso de estudio 1: circuito RC.

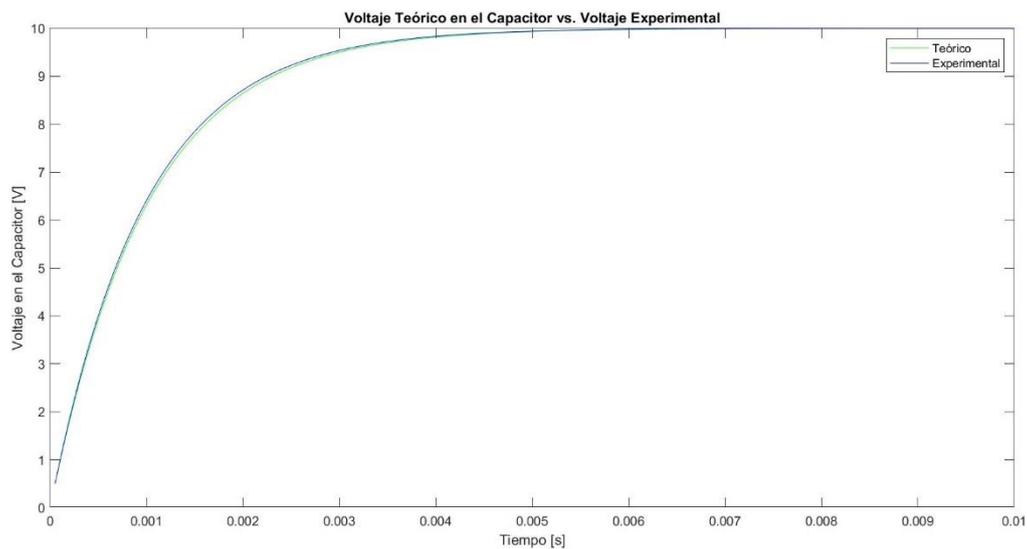


Figura 10. Gráfico teórico-experimental de voltaje del caso de estudio 1.

La Figura 10 muestra los resultados obtenidos en la simulación comparados con los datos teóricos, estos resultados se graficaron en el programa MATLAB. De color azul se pueden observar los resultados experimentales obtenidos por el circuito simulador diseñado, después de que las 600 iteraciones hayan transcurrido. El tiempo que demora el circuito simulador en extraer todos sus resultados es un tiempo aproximado de 600 veces el tiempo de muestreo o H definido en el diseño. Se puede observar en la Figura 10 que la curva de voltaje experimental en el capacitor tiene valores muy aproximados a los que se presentan en la curva teórica. Se observa también que el voltaje en el nodo alcanza su estado estable tomando un valor de voltaje máximo de 10 [V].

5.2.2 Caso de estudio 2: circuito de 4 nodos.

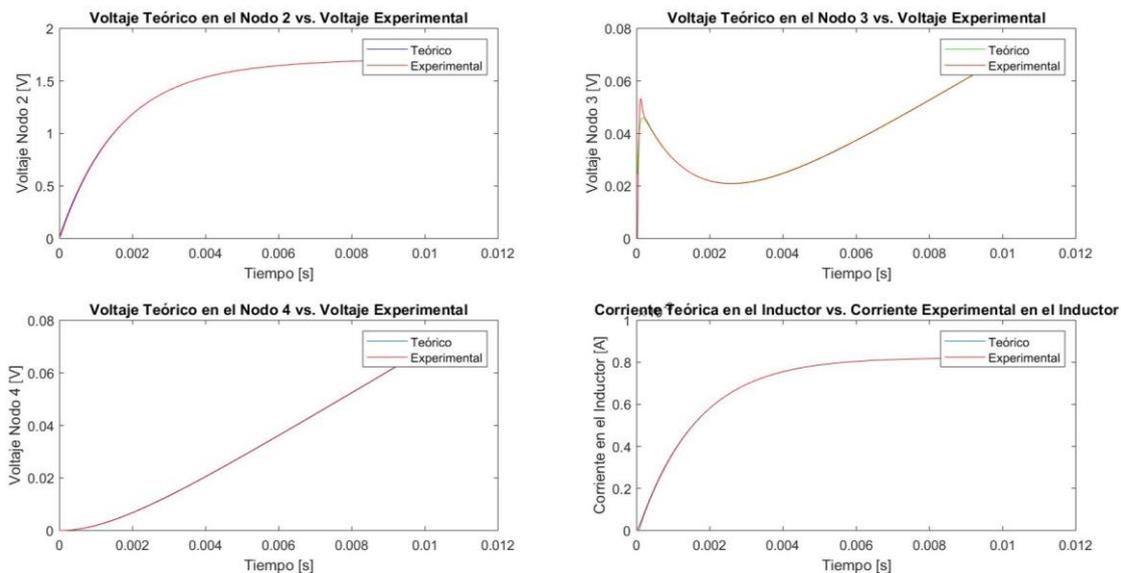


Figura 11. Gráficos experimentales-teóricos de voltajes y corrientes: caso de estudio 2.

En la Figura 11 se muestran los gráficos de los resultados experimentales-teóricos de los voltajes y corrientes para el circuito de cuatro nodos. Los datos experimentales, extraídos del circuito diseñado, son de color rojo y los datos teóricos, extraídos de PSpice, son de color verde o azul.

En el gráfico del voltaje en el nodo 2 se observa que ambas curvas poseen la misma tendencia lo que genera un error mínimo. También se aprecia que el voltaje alcanza su carga máxima en aproximadamente 1.7 [V].

En el gráfico del voltaje en el nodo 3 se observa una sobre elongación durante su etapa inicial del estado transitorio. A medida que transcurre el tiempo, los valores experimentales alcanzan la estabilización y siguen la misma tendencia de los valores teóricos. Debido a que el simulador fue diseñado para circuitos con una constante del tiempo de aproximadamente diez veces el valor de H, no se puede observar el momento en el que el voltaje alcanza la estabilidad, motivo por el cual se observa únicamente el comportamiento transitorio donde el voltaje en el nodo mantiene una tendencia creciente.

En el gráfico del voltaje en el nodo 4 no se puede observar el comportamiento estable del circuito debido a que la constante de tiempo es menor que para la que se diseñó el circuito. En las 600 iteraciones realizadas se puede observar que el comportamiento del voltaje de ambos resultados conserva el comportamiento creciente.

En el gráfico de la corriente se aprecia que tanto el comportamiento de la corriente teórica como experimental tienen la misma tendencia. Para este caso se observa que la corriente se estabiliza y alcanza su valor máximo de corriente en aproximadamente 8×10^{-4} [A].

5.2.3 Caso de estudio 3: circuito de 10 nodos.

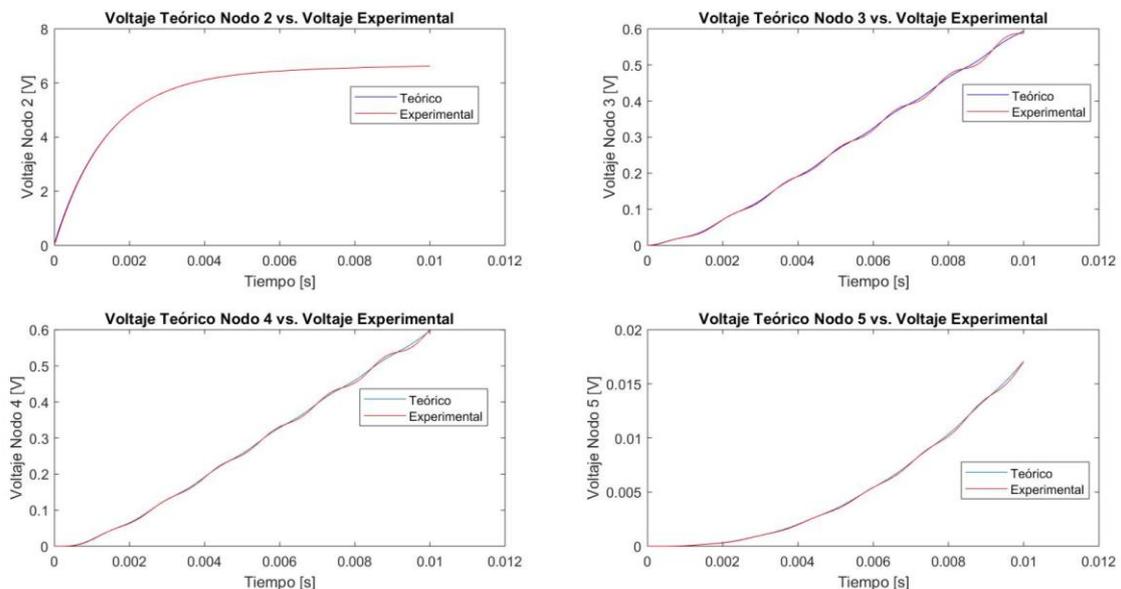


Figura 12. Gráficos experimentales-teóricos de voltajes del nodo 2 al 5: caso de estudio 3.

En la Figura 12 se muestran los resultados experimentales-teóricos de los voltajes en los nodos dos al cinco para el circuito de diez nodos. Los datos experimentales, extraídos del circuito diseñado, son de color rojo y los datos teóricos, extraídos de PSpice, son de color azul.

En el gráfico de voltaje en el nodo 2 se observa que los datos experimentales y teóricos se asemejan entre sí. Además, se aprecia que el voltaje alcanza su estado estable

donde su máximo es aproximadamente 6.5 [V]. En el gráfico de voltaje en el nodo 3 se observa que los datos experimentales siguen la misma tendencia del voltaje teórico con leves oscilaciones. El voltaje en este nodo no alcanza su estado estable por lo que se observa únicamente el comportamiento creciente del voltaje.

En el gráfico del voltaje en el nodo 4 se aprecia que ambas curvas poseen la misma tendencia mostrando así un comportamiento creciente debido a que en la simulación no alcanza el estado estable del voltaje en el nodo mencionado. Los resultados experimentales presentan cierta oscilación con respecto a los valores teóricos, esta oscilación es causada por el error de cálculo propio del método numérico implementado.

En el gráfico del voltaje en el nodo 5 éste tiene un comportamiento similar, con tendencia creciente en la figura no se logra apreciar el estado estable. Se puede ver que los datos experimentales se ajustan correctamente a los datos teóricos mostrando una leve oscilación en sus valores de voltaje.

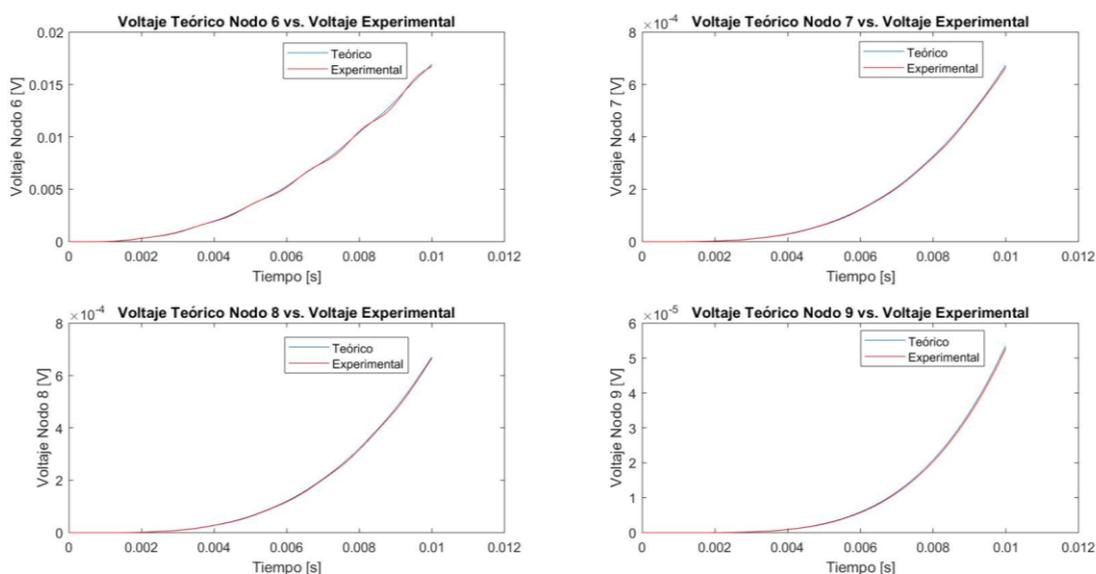


Figura 13. Gráficos experimentales-teóricos de voltajes del nodo 6 al 9: caso de estudio 3.

En la Figura 13 se muestran los resultados experimentales-teóricos de los voltajes en los nodos del seis al nueve, para el circuito de diez nodos. Los datos experimentales extraídos del circuito diseñado son de color rojo y los datos teóricos extraídos de PSpice, son de color azul.

En los gráficos de los diferentes nodos el voltaje sigue una tendencia creciente y durante las seiscientas iteraciones realizadas no se aprecia el estado estable del voltaje. Sin embargo, se aprecia que los datos experimentales tienen una buena tendencia y poseen una gran similitud con los datos teóricos.

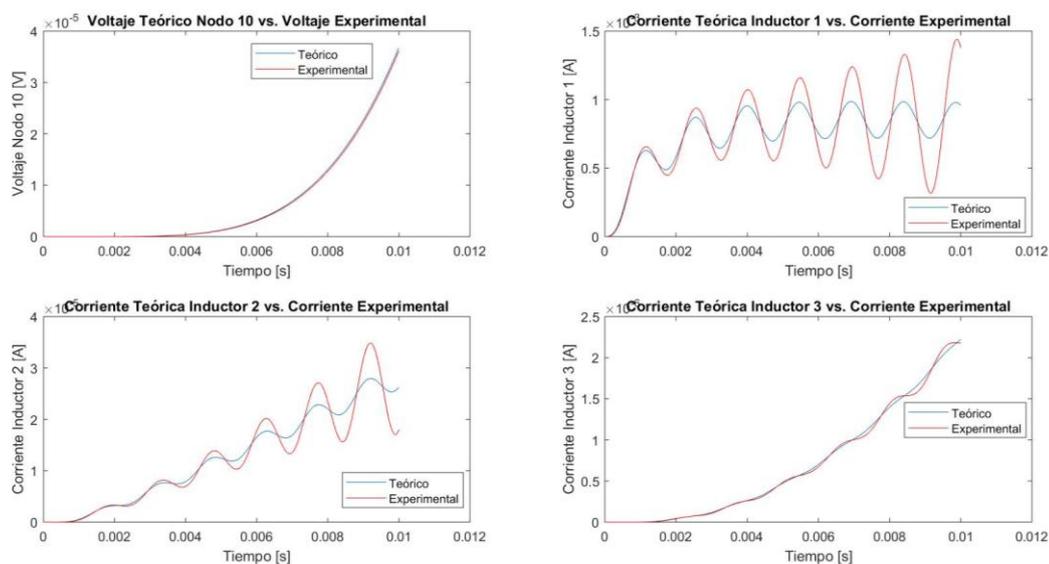


Figura 14. Gráficos experimentales-teóricos del voltaje en el nodo 10 y corrientes: caso de estudio 3.

En la Figura 14 se muestran los gráficos de los resultados experimentales-teóricos de los voltajes en el nodo diez y de las tres corrientes. Los datos experimentales, extraídos del circuito diseñado, son de color rojo y los datos teóricos, extraídos de PSpice, son de color verde o azul.

En el gráfico del voltaje en el nodo 10, el voltaje experimental y teórico tienen un comportamiento creciente donde no se puede observar el estado estable del mismo.

En el gráfico de la corriente 1 se puede comentar que ambas corrientes tienen un estado transitorio agresivo en el cual se puede apreciar una sobre elongación que oscila para ambos casos entre 0.5 y 1.5 [mA]. Para el caso de la corriente se observa que los resultados experimentales presentan un mayor error contra los valores teóricos, pero la corriente estimada sigue la misma tendencia de la teórica, esta diferencia entre ambos conjuntos de datos se debe al error de cálculo propio del método numérico de Euler.

En el gráfico de la corriente 2, los valores experimentales y teóricos mantienen una tendencia creciente que presenta un cierto error entre los resultados. La corriente en el inductor se mantiene en el estado transitorio sin llegar al estado estable donde se pueda apreciar el valor de asentamiento de la corriente.

El gráfico de la corriente 3 tiene un comportamiento creciente durante las seiscientas iteraciones completadas, por lo cual no se puede ver el comportamiento en estado estable de ésta. Se puede mencionar también que la corriente experimental tiene un comportamiento semejante al teórico presentando oscilaciones durante el transcurso del tiempo. La corriente en este inductor se encuentra en el orden de los [μ A].

6. DISCUSIÓN

El análisis del tiempo de simulación es muy importante, por este motivo en esta sección se analiza y se discute la veracidad de los resultados con respecto al tiempo, como lo explican Bélanger, et al. (2010), si las operaciones requeridas para completar la simulación no se completan durante el tiempo de muestreo fijo establecido, la simulación en tiempo real se considera como errónea generando lo que se conoce como overruns. Lo que significa que la simulación en tiempo real ejecuta las operaciones necesarias para el sistema a simular precisamente con tiempos que se asemejan a su contraparte teórica sin exceder el paso en el tiempo establecido. Debido a que el hardware fue diseñado para que

todas las operaciones se realicen en paralelo, el tiempo que toma simular circuitos de 1 nodo o de 10 nodos es el mismo.

A continuación, se muestran los resultados de tiempo total de simulación y la longitud del paso de tiempo entre cada iteración, para posteriormente discutir los resultados y la calidad de estos.



Figura 15. Análisis de tiempo de simulación.

Como se mencionó anteriormente la frecuencia original del reloj es 50 [MHz], la cual experimentalmente se fue dividiendo hasta alcanzar un periodo de tiempo entre iteraciones que no sobrepase el tiempo de muestreo teórico establecido de 1.66667×10^{-5} [s] o 0.016667 [ms]. Para obtener los resultados sin incurrir en overruns, experimentalmente se observó que la frecuencia del reloj se puede dividir un máximo de 8 veces, de esta forma se obtuvieron los resultados de tiempo que se observan en la Figura 15. Teóricamente transcurridas 600 iteraciones el tiempo total debería tener un valor aproximado de 10 [ms], mientras que en la Figura 15, en el marcador de color amarillo se observa que el tiempo total de simulación fue de 9.833 [ms]. Con estos resultados se puede acotar que la diferencia entre el tiempo total de simulación teórico y experimental

es 1.67×10^{-4} [s]. Este resultado muestra un tiempo total de simulación aceptable y se puede mencionar que después de 600 iteraciones no se ha incurrido en overruns o se ha excedido el tiempo de simulación. En la parte inferior de la Figura 15 se observan los resultados generados por el contador de iteraciones. El tiempo de muestreo teórico tiene un valor de 0.016667 [ms], en los marcadores azules se observa los valores del paso de tiempo entre iteraciones para las muestras tomadas en la Figura 15; los resultados de la muestran que el tiempo de muestreo experimental entre iteraciones tiene un valor de 0.01638 [ms]. En consecuencia, con estos resultados se puede mencionar que entre el tiempo de muestreo teórico y experimental existe una diferencia de 2.867×10^{-7} [s]. Este resultado indica que el tiempo de muestreo experimental no excede el tiempo definido teóricamente. Finalmente, con estas comparaciones entre resultados teóricos y experimentales se observa que el diseño cumple con las restricciones de tiempo, computando los resultados en un tiempo menor que el establecido en su contraparte teórica demostrando de esta forma la validez de nuestros resultados.

Al observar los datos arrojados por la simulación de los diferentes casos de estudio, se apreció de forma gráfica que la mayoría de los resultados experimentales se aproximan de forma precisa a los valores de referencia obtenidos en PSpice. Sin embargo, es de interés conocer de forma más exacta el error existente entre los valores de voltaje y corriente experimentales y teóricos en cada caso de estudio, y probar así la validez del diseño. Para analizar el error generado por los resultados se aplicó el método raíz del error cuadrático medio o RMSE. La expresión de dicho error está dada por la siguiente expresión en donde \hat{y}_i es el valor experimental y y_i el valor de referencia (T. Chai and R. R. Draxler, 2014).

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2} \quad (7)$$

Caso de estudio 1		RMSE
Voltaje en circuito RC	V_1	0.0271
Caso de estudio 2		
Voltajes en circuito de 4 nodos	V_2	0.0038
	V_3	0.0014
	V_4	1.3093e-4
Corriente en circuito de 4 nodos	I_1	2.1149e-6
Caso de estudio 3		
Voltajes en circuito de 10 nodos	V_2	0.0134
	V_3	0.0037
	V_4	0.0032
	V_5	9.1172e-05
	V_6	8.9523e-05
	V_7	3.1122e-06
	V_8	3.0351e-08
	V_9	3.3416e-07
	V_{10}	2.5473e-07
	Corriente en circuito de 10 nodos	I_1
I_2		2.5298e-06
I_3		2.7269e-08

Tabla 1. Valores RMSE obtenidos a partir de los valores experimentales y teóricos.

En la Tabla 1 se muestran los cálculos de la raíz del error cuadrático medio analizado para los tres casos de estudio. Estos errores son producidos por dos factores. El

primer factor es el método numérico el cual produce un error de propagación, es decir en cada iteración se acumula un porcentaje de error. El segundo factor que afecta a los resultados son el número de iteraciones realizadas, mientras más iteraciones se realicen el error será menor debido a que se reducirá la tolerancia. La tolerancia obtenida para las seiscientas iteraciones realizadas se encuentra en el rango de 4×10^{-4} y 4×10^{-6} para los voltajes, y de 1.5×10^{-5} a 4.02×10^{-10} para las corrientes. Por otro lado, se observa que los valores de RMSE de las corrientes son menores debido al orden de magnitud en el que se encuentran los valores de corriente. Debido a las oscilaciones obtenidas en las corrientes, se observa gráficamente que los valores experimentales no se ajustan en su totalidad a los valores teóricos.

Las oscilaciones observadas en las corrientes son causadas por el error de propagación propio del método numérico, error que incrementa entre cada valor. De acuerdo con las observaciones realizadas por Zidan, et al. (2011), el método de Euler produce resultados menos precisos en comparación con otros métodos como es el método de Runge-Kutta de cuarto orden (RK4). La ventaja del método de Euler es la simplicidad de sus cálculos. Por otro lado, el método RK4 produce la aproximación numérica más precisa, pero los cálculos que se deben realizar para computar los resultados son más extensos y complejos, y existe un incremento en los recursos requeridos para implementar este método numérico.

Consecutivamente, analizando los valores de RMSE y las observaciones visuales de los resultados se determinó que existe una buena aproximación de valores experimentales comparados con el modelo de referencia.

Dicho esto, el error de cálculo podría disminuir cambiando el método numérico de Euler por uno más preciso como el método RK4. Sin embargo, el tiempo requerido por este método para computar los resultados incrementa. Otra técnica que permite

mejorar los resultados utilizando el método de Euler es disminuir el tiempo de muestreo entre cada iteración, pero este procedimiento causa un incremento en el número de recursos utilizados.

Finalmente, analizando los distintos factores que producen error en los resultados, se determinó que existe una buena aproximación entre los valores experimentales obtenidos comparados con el modelo de referencia proporcionado por PSpice.

7. CONCLUSIONES

A pesar de que teóricamente el método numérico para resolver ecuaciones diferenciales de Euler posee un alto error de propagación (en comparación con otros métodos), la calidad de los resultados obtenidos experimentalmente posee gran semejanza con los resultados del modelo teórico en el cual se basó el proyecto. Los resultados muestran una alta validez comprobada por el análisis realizado con RMSE, a pesar del alto error propio del método numérico implementado.

Disminuir el valor de H , incrementa la fidelidad de los resultados que se pueden obtener, debido a que el número de muestras tomadas de la señal incrementa. Esto genera un aumento del número de iteraciones causando un crecimiento de los recursos utilizados en la medida que el número de iteraciones lo haga.

Como se mencionó anteriormente es muy importante que los resultados de la simulación experimental respeten las restricciones de tiempo establecidas teóricamente. Esto sucede si los valores generados en cada iteración no exceden el tiempo de muestreo fijado en el diseño. De esta forma se puede concluir que el simulador no solo garantiza la calidad de los resultados, sino que trabaja en tiempo real cumpliendo con las características que permiten al circuito trabajar en tiempo real.

Finalmente, el diseño que se realizó al ser una primera aproximación no tomó en cuenta optimizaciones en el hardware. Por este motivo todo el hardware fue diseñado para que las operaciones se realicen en paralelo incrementando así los recursos utilizados por el sintetizador lógico como se pudo observar en la sección de reportes de síntesis. La ventaja de tener un diseño con todos sus módulos en paralelo es que existe un incremento en la tasa de procesamiento de datos y reducción en el tiempo de procesamiento.

8. REFERENCIAS BIBLIOGRÁFICAS

- Bélanger, J., Venne, P., & Paquin, J. N. (2010). The what, where and why of real-time simulation. *Planet Rt*, 1(1), 25-29.
- Chai, T., & Draxler, R. R. (2014). Root mean square error (RMSE) or mean absolute error (MAE)?. *GMDD*, 7(1), 1525-1534.
- Chapra, S. C., & Canale, R. P. (2010). *Numerical methods for engineers*. Boston: McGraw-Hill Higher Education.
- Harris, D., & Harris, S. (2010). *Digital design and computer architecture*. Morgan Kaufmann.
- Levinson, N. (1946). The Wiener (root mean square) error criterion in filter design and prediction. *Journal of Mathematics and Physics*, 25(1-4), 261-278.
- Liu, C., Ma, R., Bai, H., Gechter, F., & Gao, F. (2018). A new approach for FPGA-based real-time simulation of power electronic system with no simulation latency in subsystem partitioning. *International Journal of Electrical Power & Energy Systems*, 99, 650-658.
- Matar, M., Abdel-Rahman, M., & Soliman, A. M. (2005, June). FPGA-based real-time digital simulation. In *Int. Conf. on Power Systems Transients*.
- Qin, Z., & Cheng, C. K. (2009). *Symbolic analysis and reduction of VLSI circuits*. Springer Science & Business Media.
- Xilinx. (2019). *Vivado Design Suite User Guide: Design Analysis and Closure Techniques*. UG906 (v2019.2).
- Zidan, M. A., Radwan, A. G., & Salama, K. N. (2011, December). The effect of numerical techniques on differential equation based chaotic generators. In *ICM 2011 Proceeding* (pp. 1-4). IEEE.

9. ANEXO A: Reporte de Utilización de Recursos de Síntesis

1.1 Summary of Registers by Type

```
-----
```

Total	Clock Enable	Synchronous	Asynchronous
0	-	-	-
0	-	-	Set
0	-	-	Reset
0	-	Set	-
0	-	Reset	-
0	Yes	-	-
1	Yes	-	Set
326988	Yes	-	Reset
0	Yes	Set	-
6	Yes	Reset	-

```
-----
```

2. Memory

```
-----
```

Site Type	Used	Fixed	Available	Util%
Block RAM Tile	0	0	60	0.00
RAMB36/FIFO*	0	0	60	0.00
RAMB18	0	0	120	0.00

```
-----
```

3. DSP

```
-----
```

Site Type	Used	Fixed	Available	Util%
DSPs	0	0	80	0.00

```
-----
```

4. IO and GT Specific

```

-----
+-----+-----+-----+-----+
| Site Type | Used | Fixed | Available | Util% |
+-----+-----+-----+-----+
| Bonded IOB | 5 | 0 | 100 | 5.00 |
| Bonded IPADs | 0 | 0 | 2 | 0.00 |
| Bonded IOPADs | 0 | 0 | 130 | 0.00 |
| PHY_CONTROL | 0 | 0 | 2 | 0.00 |
| PHASER_REF | 0 | 0 | 2 | 0.00 |
| OUT_FIFO | 0 | 0 | 8 | 0.00 |
| IN_FIFO | 0 | 0 | 8 | 0.00 |
| IDELAYCTRL | 0 | 0 | 2 | 0.00 |
| IBUFDS | 0 | 0 | 96 | 0.00 |
| PHASER_OUT/PHASER_OUT_PHY | 0 | 0 | 8 | 0.00 |
| PHASER_IN/PHASER_IN_PHY | 0 | 0 | 8 | 0.00 |
| IDELAYE2/IDELAYE2_FINEDELAY | 0 | 0 | 100 | 0.00 |
| ILOGIC | 0 | 0 | 100 | 0.00 |
| OLOGIC | 0 | 0 | 100 | 0.00 |
+-----+-----+-----+-----+

```

5. Clocking

```

-----
+-----+-----+-----+-----+
| Site Type | Used | Fixed | Available | Util% |
+-----+-----+-----+-----+
| BUFGCTRL | 8 | 0 | 32 | 25.00 |
| BUFIO | 0 | 0 | 8 | 0.00 |
| MMCME2_ADV | 0 | 0 | 2 | 0.00 |
| PLLE2_ADV | 0 | 0 | 2 | 0.00 |
| BUFMRCE | 0 | 0 | 4 | 0.00 |
| BUFHCE | 0 | 0 | 48 | 0.00 |
| BUFR | 0 | 0 | 8 | 0.00 |
+-----+-----+-----+-----+

```

6. Specific Feature

```

-----
+-----+-----+-----+-----+
| Site Type | Used | Fixed | Available | Util% |
+-----+-----+-----+-----+
| BSCANE2 | 0 | 0 | 4 | 0.00 |
| CAPTUREE2 | 0 | 0 | 1 | 0.00 |
| DNA_PORT | 0 | 0 | 1 | 0.00 |
| EFUSE_USR | 0 | 0 | 1 | 0.00 |
| FRAME_ECCE2 | 0 | 0 | 1 | 0.00 |
| ICAPE2 | 0 | 0 | 2 | 0.00 |
| STARTUPE2 | 0 | 0 | 1 | 0.00 |
| XADC | 0 | 0 | 1 | 0.00 |
+-----+-----+-----+-----+

```