Evaluación de Transitorios Electromagnéticos en Tiempo Acelerado mediante la Transformada Numérica de Laplace en un FPGA

Evaluating Electromagnetic Transients in Accelerated-Time through the Numerical Laplace Transform in an FPGA.

J. R. Zuluaga[1], J. L. Naredo[2], C. H. Rodríguez[3] L. J. Castañón[4], M. G. Vega[2].

[1]jzuluaga@gdl.cinvestav.mx / [2]jlnaredo@gdl.cinvestav.mx / [3]chrodrigue@gdl.cinvestav.mx / [5]ldcastanon@gdl.cinvestav.mx

Recibido: octubre 17, 2019 / Aceptado: enero 23, 2020 / Publicado: septiembre 18, 2020

Resumen. Se reporta la implementación de la Transformada Numérica de Laplace Inversa (TNLI) en un Arreglo de Compuertas Programable por Efecto de Campo (FPGA, por sus siglas en inglés) para realizar cálculos de transitorios electromagnéticos en redes de energía eléctricas en tiempo acelerado; es decir, varias veces más rápido que el tiempo real. Se utiliza una tarjeta de desarrollo FPGA Virtex 6xc6vlx240t-1ff1156 de tecnología Xilinx®. Se presentan 2 casos de aplicación. El primero consiste en la inversión numérica de una función de Laplace cuya solución en el dominio del tiempo es conocida analíticamente. El segundo caso corresponde a la evaluación de la respuesta transitoria de una línea de transmisión monofásica con parámetros RLGC dependientes de la frecuencia. Los resultados se validan con el programa comercial PSCAD/ EMTDC®. Estos muestran la viabilidad de la herramienta propuesta para analizar transitorios de redes eléctricas en tiempo acelerado

Palabras clave: Transitorios Electromagnéticos, Coordinación de Aislamientos, Transformada Numérica de Laplace (TNLI), FPGAs, Simulación en Tiempo Real, Simulación en Tiempo acelerado.

Abstract. This paper reports the implementation of the Inverse Laplace Numerical Transform (TNLI) in a Field Programmable Gate Array (FPGA) for calculating power-system electromagnetic transients in accelerated time; that is, several times faster than real time. A Virtex 6 xc6vlx240t-1ff1156 FPGA development card from Xilinx® technology is used for this purpose. Two application cases are presented here. The first one is the numerical inversion of a Laplace test-function whose time-domain solution is known analytically. The second case corresponds to the evaluation of the transient response of a single-phase transmission line with frequency-dependent RLGC parameters. This response is validated with the commercial-grade program PSCAD/EMTDC®. Finally, the performance attained by the reported TNLI-FPGA implementation demonstrates its ability for calculating power-system transients in accelerated time.

Key Words: Electromagnetic Transients, Insulation Coordination, Numerical Laplace Transform, FPGAs, Real-Time Simulation, Accelerated-Time Simulation.

1. INTRODUCCIÓN

La Transformada de Laplace es una herramienta poderosa para el análisis dinámico de sistemas eléctricos. A través de esta transformada, las ecuaciones integro-diferenciales que describen tales sistemas se transforman en relaciones algebraicas cuya solución puede obtenerse fácilmente. Sin embargo, estas soluciones todavía deben ser llevadas al dominio del tiempo aplicando la transformada inversa de Laplace. En la mayoría de los casos, esto suele resultar muy difícil e incluso imposible de realizar analíticamente. Por este motivo se suele recurrir a formas de inversión numérica que en lo sucesivo se denominarán genéricamente como la Transformada Numérica de Laplace (TNL) [1], [2], [3], y [4].

Muchas de las aplicaciones de la TNL requieren un uso intensivo de recursos de cómputo, como son el tiempo de procesador (CPU) y la cantidad de memoria. Tales aplicaciones, además, cada vez suelen demandar menores tiempos de cómputo. Un caso de este tipo es el de los estudios estadísticos de Transitorios Electromagnéticos (TEMs) por maniobra en sistemas de transmisión de energía eléctrica. Entre otras aplicaciones, éstos son requeridos para determinar y coordinar niveles de aislamientos. En ellos se deben realizar simulaciones de miles de operaciones de interruptores para determinar la distribución

ISSN: 1665-5745 www.e-gnosis.udg.mx

estadística de sobre tensiones y sobre corrientes. Con las herramientas convencionales, el tiempo de cómputo de estos estudios suele ser excesivo y costoso [5]. El objetivo de este artículo es proponer y presentar una implementación de la Transformada Numérica de Laplace Inversa (TNLI) en un procesador del tipo arreglo de compuertas programable por efecto de campo, o FPGA (por sus siglas en inglés), así como el de demostrar la capacidad de dicha implementación para calcular TEMs en tiempo acelerado; es decir, varias veces más rápido que en tiempo real [10]. Para este objetivo se adopta la técnica de la TNLI basada en el truncamiento mediante ventana de datos, [1], [2], [4], [6], [7]. El FPGA utilizado es el Virtex 6 xc6vlx240t-1ff1156 de Xilinix®. Una implementación previa de la TNLI en un FPGA, quizá la única, fue reportada por Yonemoto et al., en el 2002 [11]. En ésta se utilizó la técnica de la TNLI de Durbin [12] cuya eficiencia numérica es 50 % inferior que la que aquí se propone; adicionalmente, no se logró en [11] la reproducción de un transitorio en tiempo real.

2. EVALUACIÓN NUMÉRICA DE LA TRANSFORMADA INVERSA DE LAPLACE

Sea *f(t)* una función del tiempo "*t*" que representa una señal física. Su transformada de Laplace *F(s)* está definida por la siguiente función [13]:

$$F(s)=\int_0^{\infty} f(t)e^{-(s)t}dt\,, \tag{1}$$

donde $s=c+j\omega$ es la variable compleja de Laplace. Su parte imaginaria ω es la frecuencia angular y su parte real "*c"* es la constante de amortiguamiento [1], [2], y [3]. Por otro lado, la transformada inversa de Laplace para funciones reales viene dada por la siguiente expresión [1]:

$$f(t)=\frac{e^{ct}}{\pi}\mathrm{Re}\left\{\int_0^{\infty} F(c+j\omega)e^{j\omega t}d\omega\right\}. \tag{2}$$

El tratamiento numérico de (2) requiere truncar los límites de integración, así como discretizar las variables ω y t [2]. En este trabajo se adopta la técnica de inversión de la transformada de Laplace (TNLI) basada en el truncamiento de ω mediante ventanas de datos [2].

2.1. Tratamiento numérico de la Transformada Inversa de Laplace

El truncamiento del rango de integración en (2) se puede representar mediante la multiplicación de $F(c+j\omega)$ por una ventana rectangular $\sigma_r(\omega)$, la cual vale cero fuera de dicho rango:

$$f(t)\approx \hat{f}(t)=\frac{e^{ct}}{\pi}\mathrm{Re}\left\{\int_0^{\infty} F(c+j\omega)\sigma_r(\omega)e^{j\omega t}d\omega\right\} \tag{3}$$

donde $\hat{f}(t)$ representa una aproximación a $f(t)$ y $\sigma_r(\omega)$ es la ventana rectangular.

El truncamiento con las ventanas rectagulares introduce errores conocidos como de Gibbs de hasta 9.6 % [2]. La ventana de Von Hann (o Hanning) $\sigma_{hn}(\omega)$ resulta altamente conveniente para reducir dichos errores y ha sido adoptada extensamente en el análisis de TEMs en sistemas de energía eléctrica [3]:

$$\sigma_{hn}(\omega)=\begin{cases}\frac{1}{2}\left[1+\cos(\pi\omega/\Omega)\right], & 0\le\omega\le\Omega\\ 0, & \omega<0,\ \omega>\Omega\end{cases}. \tag{4}$$

ISSN: 1665-5745

2.2. Discretización

La variable continua *t* se suele discretizar en *N* pasos uniformes de tamaño Δt, donde $\Delta t=T/N$ y *T* representa el tiempo de observación. Cabe mencionar que, de acuerdo con el ancho de banda "**Ω**" en (4), Δt debe cumplir con el criterio de la tasa de muestreo de Nyquist [2], [6], [7]. Por su parte, $F(c+j\omega)$ se puede discretizar empleando un muestreo uniforme, ya sea regular o impar [7]. De esta forma, se producen 4 métodos para la TNLI: **1)** el regular bilateral (RB) **2)** el impar bilateral (IB) **3)** el regular unilateral (RU) y **4)** el impar unilateral (IU). En este artículo se hace énfasis sobre el muestreo RU, el cual se representa en la figura 1. La siguiente expresión denota la transformada inversa de Laplace con éste método [14]:

$$\hat{f}_n = \frac{e^{cn\Delta t}}{\Delta t}\,\mathrm{Re}\left\{\frac{1}{N}\sum_{k=0}^{N-1}\left[F_k\sigma_{hn,k}e^{\frac{j2\pi kn}{N}}\right]\right\} \tag{5}$$

donde $\sigma_{hn,k}$ y F_k representan las *k*-ésimas muestras de $\sigma_{hn}(\omega)$ y $F(c+j\omega)$, respectivamente; es decir, $\sigma_{hn}(k\Delta\omega)$ *y* $F(k\Delta\omega)$.

La sumatoria en (5) se puede calcular mediante el algoritmo de la Transformada Rápida de Fourier Inversa (IFFT, por sus siglas en inglés); simbólicamente:

$$\overline{f} = \overline{K} \otimes \mathrm{Re}\left\{IFFT\left(\overline{F} \otimes \overline{\sigma}\right)\right\} \tag{6}$$

donde $\overline{f}$, $\overline{K}$, $\overline{F}$ y $\overline{\sigma}$ son vectores de *N* muestras y $\otimes$ representa el producto de Hadamard; esto es, el producto entre dos vectores de dimensión *N* que resultan en otro vector de la misma dimensión y cuyos elementos son el producto de los elementos correspondientes de los vectores que se multiplican. Nótese que $\overline{f}$ es el vector de la *N* muestras "$\tilde{f}_n$" de la señal en el dominio del tiempo obtenida tras la inversión, $\overline{F}$ es el vector de la *N* muestras "$\overline{F}_K$" de la función en el dominio de Laplace y $\overline{\sigma}$ es el vector de la *N* muestras $\sigma_{hn,k}$ de la función de ventana de datos. En cuanto a $\overline{K}$ en (6), éste es un vector de coeficientes que toma cuatro formas diferente dependiente del tipo de muestreo, el cual, para el muestreo RU viene dado por:

$$\overline{K}_{RU} = 2exp(c\overline{n}\Delta t)/\Delta t \tag{7}$$

3. IMPLEMENTACIÓN DE LA TNLI EN FPGA

En la actualidad los FPGAs se utilizan cada vez más en aplicaciones de sistemas eléctricos de potencia. Esto es debido principalmente a dos factores: 1) su arquitectura que permite cómputo en paralelo y 2) su alta velocidad de procesamiento que reemplaza en gran medida a los circuitos integrados de aplicación específica o ASICs, así como a los procesadores digitales de señales o DSPs [8], [9], [10].

En este artículo se reporta la implementación de la TNLI basada en ventaneo la cual se resume en la expresión general (6). Dicha implementación se ilustra en la Fig. 2 y los pasos correspondientes se enlistan a continuación:

1. Definir la representación numérica ya sea de punto fijo o de punto flotante, así como su número de bits.
2. Calcular el vector $\overline{K}$ fuera de línea y cargarlo en las memorias EPROM del FPGA.
3. Adecuar y cargar el algoritmo de la IFFT en el FPGA.
4. Calcular la ventana de Hanning con el algoritmo CORDIC.

ISSN: 1665-5745

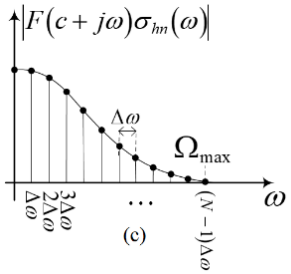


Figura 1. Discretización de *F(c+jω)* muestreo RU.

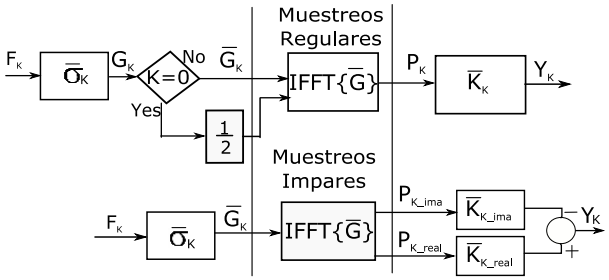


Figura 2. Diagrama de bloques de la TNLI.

3.1. Representación de variables en un FPGA

Los FPGAs están limitadas en cuanto a espacio disponible de memoria; por lo tanto, la implementación de un algoritmo en ellas plantea varios desafíos. Uno de estos es determinar el formato de representación numérica más apropiado para las variables [15]. Este formato en una computadora convencional suele ser en punto flotante a 32 bits (precisión simple) y a 64 bits (precisión doble); ésto, de acuerdo con el estándar IEEE 754-1985 para números en punto flotante [16]. En un FPGA se considera que dicha representación demanda una gran cantidad de recursos (look-up tables o memorias). El uso de representación en punto fijo, por otro lado, es mucho menos demandante en recursos de memoria; sin embargo, esto se logra a costa de una menor precisión [17].

En este artículo se adopta la representación de punto fijo con complemento a dos por las siguientes razones: su aritmética es simple, los IP-cores disponibles permiten formatos en punto fijo sin cambio alguno y la interconexión de todas las instancias del método es más sencilla.

Todos los valores que se almacenan en la EPROM del FPGA deben convertirse a representación de punto fijo y, de igual manera, los datos que se obtienen del FPGA deben ser convertidos de binario a representación real punto flotante.

3.2. Cálculo de los vectores K

De acuerdo con la expresión (7), los componentes del vector $\bar{K}$ dependen del valor que se dé al factor de amortiguamiento "*c*". Para este último se suele usar la siguiente expresión [6]:

$$c = -\ln(\varepsilon_r)/T \tag{8}$$

donde ε_r es el error deseado de solapamiento (aliasing) que suele especificarse entre 10^{-2} y 10^{-6} [2].

En el caso de una TNLI implementada en un FPGA, debido al número limitado de bits para representar los elementos del vector $\bar{K}$, el error obtenido ε_r suele no corresponder con el especificado en (8); sobre todo cuando para este último se elige un valor muy pequeño que resulta en magnitudes muy altas de los componentes de $\bar{K}$. Se recomienda entonces seleccionar un valor conservador para ε_r en (8), del orden de 10^{-2}, y ajustarlo mediante un proceso de prueba y error. Una vez seleccionado dicho valor, se procede a calcular el vector $\bar{K}$ para cada muestreo y guardarlo en memoria EPROM del FPGA.

3.3. Implementación de la IFFT

La FFT es quizá el algoritmo más utilizado en procesamiento digital de señales y en telecomunicaciones. Este ha sido implementado en distintos tipos de hardware especializado: DSPs, FPGAs, etc. El software ISE 13.4 de Xilinx® ofrece varias implementaciones del algoritmo del algoritmo IFFT, todas estas limitadas a un número *N* de muestras que sea potencia de 2; es decir, $N=2^m$ con *m* un entero entre 3 y 16. De todas las

ISSN: 1665-5745

arquitecturas disponibles para el algoritmo de la IFFT, en este artículo se seleccionó la denominada "pipelined, streaming I/O". Esta arquitectura permite el procesamiento continuo de datos y mejor velocidad, aunque, requiere de una mayor cantidad de recursos del FPGA [18]. El FPGA aquí usado ofrece libertad para elegir diferentes arquitecturas. La Fig. 3 representa el bloque (instancia) de la IFFT con sus respectivas entradas (izquierda) y salidas (derecha).

3.4. Implementación de la ventana de datos

La expresión (4) proporciona la forma más general de la ventana de Hanning que, para el muestreo RU, se puede reescribir de la siguiente manera:

$$\bar{\sigma}_{hn}(\omega) = 0.5 + 0.5\cos(\pi\bar{m}/N), \tag{9}$$

donde

$$\bar{m} = (0,1,2,\ldots,N-1), \tag{10}$$

El cálculo interno de las funciones trigonométricas en FPGAs usualmente se lleva a cabo mediante el algoritmo CORDIC. En las librerías de Xilinx® se cuenta con dos implementaciones que, al igual que en la FFT, a medida que se gana en velocidad se requiere más espacio de silicio. El objetivo de este trabajo es realizar cómputo rápido y por tanto se selecciona la arquitectura que proporciona la mayor velocidad. El funcionamiento en detalle de estos algoritmos puede consultarse en la referencia [19]. El diagrama de bloques de la implementación se muestra en la Fig. 4. Cabe decir que este bloque también se puede ejecutar fuera de línea y cargar sus resultados en la memoria EPROM con el fin de acelerar la velocidad de los cálculos.

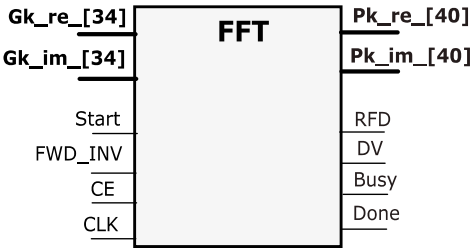


Figura. 3. Diagrama entradas/salidas de la FFT.

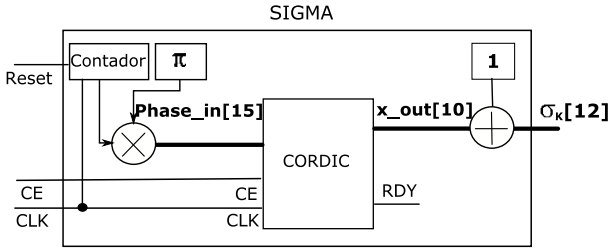


Figura 4. Diagrama de la función de ventaneo.

3.5. Diagrama de la implementación de la TNLI en una FPGA

La figura 5 resume el diseño final que incluye todas las etapas previamente descritas para el método propuesto: IFFT, ventana y memorias EPROM. Adicionalmente, este diagrama contiene bloques de contadores para acceder a las memorias del FPGA. Es importante resaltar que en los métodos con muestreo regular los elementos de $\bar{K}$ son reales, lo cual permite usar menos memoria, así como efectuar menos operaciones que en la implementación con los métodos que usan muestreo impar (ver figura 2). La tabla 1 presenta las variables en la implementación de la TNLI y proporciona el número de bits necesarios para representar a cada una de ellas. El número de bits de la variable $\bar{P}_k$ de la salida de la instancia IFFT (ver tabla 1) incrementa en proporción con el logaritmo base dos del número de muestras [18]. Para los casos de prueba en este artículo son necesarios 6 bits extra debido a que la TNLI implementada es de 64 muestras.

Para determinar el número de operaciones y el tiempo que tarda el algoritmo en realizar éstas, basta con contar el número de ciclos de reloj que toma el algoritmo y dividirlo por la frecuencia a la que conmuta dicho reloj. Esto se puede hacer mediante un contador controlado que se detenga una vez que la variable "Done" (terminado) de la instancia de la IFFT asuma el valor de "1".

ISSN: 1665-5745

4. CASOS DE ESTUDIO

4.1. Inversión de una función analítica

Considérese cómo primer caso de estudio la siguiente función en el dominio de Laplace:

$$F(s) = \frac{s \times exp\left(-0.018s\right)}{s^2 + (30\pi)^2} \tag{11}$$

que corresponde en el dominio del tiempo a una función coseno desplazada:

$$f(t) = cos\left[30\pi(t - 0.018)\right]u(t - 0.018) \tag{12}$$

La figura 6a ilustran el resultado de aplicar el algoritmo de la TNLI a (11). Los resultados con los muestreos no se incluyen aquí dado que son idénticos, pero con un error un poco mayor. En estas figuras se comparan los resultados obtenidos usando 3 valores de amortiguamiento c=30.7, 46.1 y 61.4; estos han sido calculados mediante (8) asignando niveles de error de 10%, 1% y 0.1%, respectivamente.

Puesto que la transformada inversa de (11) está dada en forma analítica por (12), es posible evaluar el error de los métodos implementados en forma precisa. Para ello se utiliza la siguiente expresión.

$$Error_k = \frac{\left| f_k - \hat{f}_k \right|}{\max(f)}; \quad k = 1,2,...N \tag{13}$$

donde f_k corresponde a una muestra de la señal analítica y $\hat{f}_k$ a esa misma muestra pero de la señal proveniente de la TNLI implementada en el FPGA. La Fig. 7 proporciona las gráficas de error obtenidas para la implementación propuesta. Se considera un factor del coeficiente de amortiguamiento de *c=61.4* que corresponde a un error de aliasing del 0.01%.

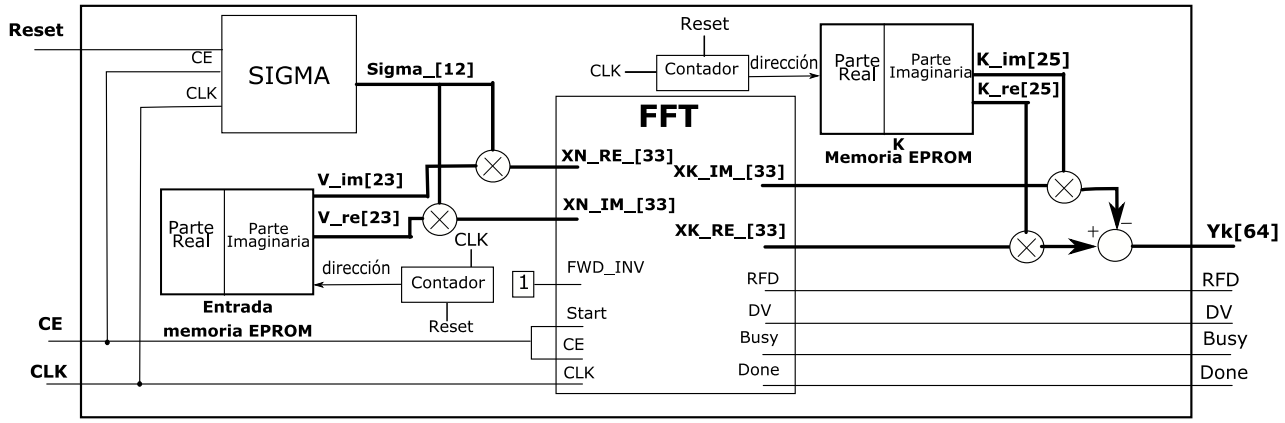


Figura 5. Diagrama de bloques de la TNLI.

4.2. Energización de una línea de transmisión monofásica

Considérese ahora, como segundo caso de estudio la energización de una línea aérea monofásica de longitud *L=240* km formada por un conductor ACSR Bluejay de radio r_c*=1.25* cm a una altura *h=12* m sobre un suelo conductor imperfecto con resistividad ρ_s*=100* Ω*m*. La figura 7a muestra la geometría transversal de la línea y proporciona las propiedades eléctricas de sus materiales, incluida la resistividad del aluminio del conductor ACSR $\rho_c=4.13\times10^{-8}$ Ω*m*. La figura 7b muestra un diagrama longitudinal de la línea junto con sus conexiones. En el extremo transmisor la línea es energizada con un escalón unitario de voltaje a través de una impedancia generalizada de fuente $Z_S=0.5+j0.03s$ Ω. En el extremo receptor la línea está

ISSN: 1665-5745

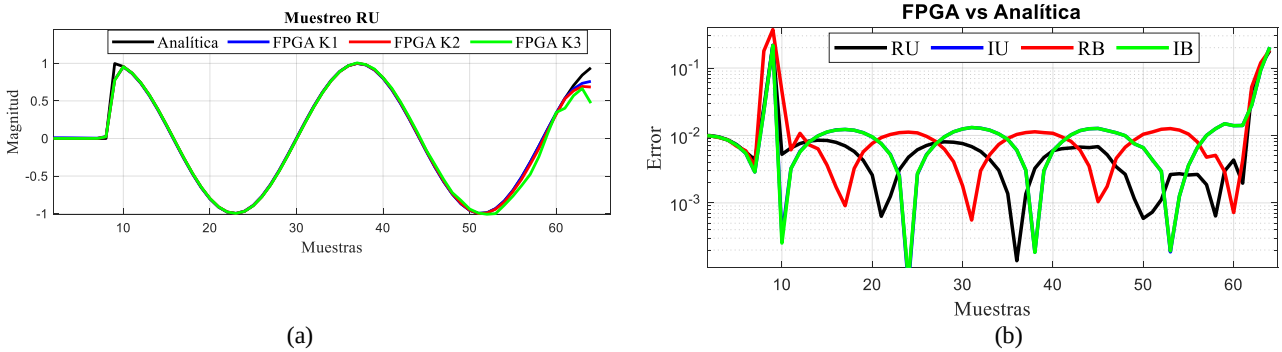
conectada a tierra mediante una carga $Z_L=1\times10^6$ Ω. En el dominio de Laplace el voltaje del extremo receptor $V_L(s)$ está dado por la siguiente expresión:

$$V_L(s) = H(s)V_s(s) \tag{14}$$

donde $V_S(s)=1/s$ es el voltaje inyectado en el extremo transmisor y $H(s)$ es la función de transferencia del sistema formado por la línea y las impedancias Z_S de fuente y Z_L de carga. Por el hecho de que $H(s)$ consiste en una composición no trivial de funciones trascendentales, su inversión analítica para obtener la forma de onda de voltaje de respuesta $v_L(t)$ resulta quizá imposible y, por lo tanto, se recurre a la TNLI.

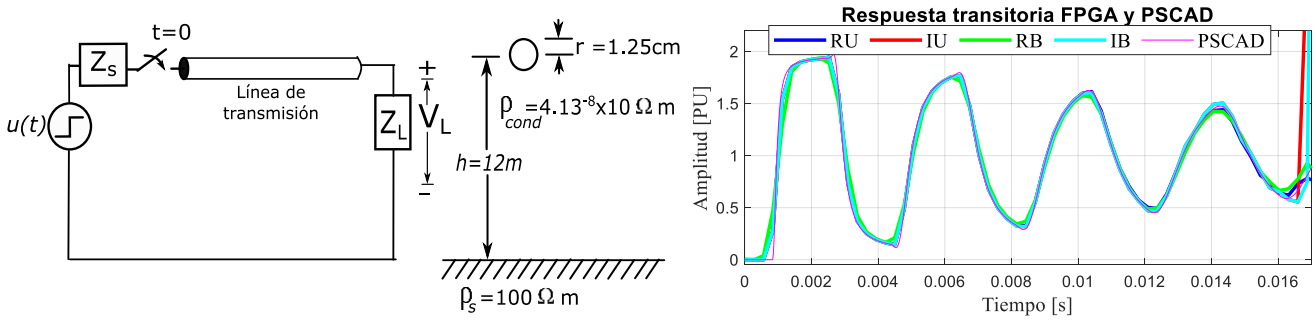
Tabla 1. Representación numérica de las variables para una TNLI de 64 muestras.

	Variables de la FPGA						
	Entrada F_k		Ventana	$G_k = H_k F_k$	$P_k = IFFT(G_k)$	K_k	$Y_k = P_k K_k$
	$\mathrm{Re}\{F_k\}$	$\mathrm{Im}\{F_k\}$	σ_k				
Bits enteros	1	1	1	2	8	22	30
Bits de signo	1	1	1	1	1	1	1
Bits fracionarios	21	21	10	31	31	2	33
Bits totales	23	23	12	34	40	25	64



(a) (b)

Figura 6. (a) Resultado de la TNLI con varios valores de amortiguamiento y (b) Error respecto a la solución analítica.



(a) (b) (c)

Figura 7. (a) Datos de la línea (b) Diagrama del circuito bajo estudio y (c) Respuesta transitoria de la línea con la TNLI-FPGA y PSCAD/EMTDC®.

Finalmente, la figura 7c muestra las formas de onda de respuesta $v_L(t)$ obtenidas mediante los cuatro métodos considerados de la TNLI ejecutados en el FPGA. A manera de validación, en dicha figura también se incluye la respuesta obtenida con el programa de grado comercial PSCAD/EMTDC®, versión 4 [20]. El tiempo de cómputo para el PSCAD es en promedio 0.7 s en una PC i7 con ambiente Windows 10. Este tiempo de ejecución debe contrastarse con los $3.55\mu s$ requeridos por la TNLI en el FPGA. El modelo que se usa en PSCAD es el "Frequency Dependent (Phase) Model'' con un ajuste racional entre 0.5 Hz y 1 MHz

ISSN: 1665-5745

usando máximo un orden de 20 polos tanto para el ajuste de la impedancia característica como para el ajuste de la función de propagación.

5. DISCUSIÓN

La tabla 2 resume la utilización de recursos lógicos dentro del FPGA. Estos recursos corresponden a ambos casos de estudio que se presentaron anteriormente. Los simuladores actuales de TEMs en tiempo real, como son el RTDS® y el OPAL-RT®, permiten simular casos de estudio con pasos de tiempo de Δt=20μs por cada muestra [8], [9], y [21]. Los métodos aquí reportados arrojan 64 muestras en 3.55μs lo cual corresponde a 360 veces más rápido que el tiempo real; sin embargo, para hacer una comparación justa con un simulador en tiempo real, aún es necesario simular una red más compleja y con más muestras.

El uso de muestreos impares incrementa el número de operaciones debido a la aritmética compleja involucrada en el cálculo del vector $\bar{K}$. Este sobrecosto no se ve reflejado en el tiempo de cómputo de los algoritmos debido a que el FPGA realiza cómputo en paralelo. Adicional a esto la variante regular unilateral (RU), también proporciona mayor precisión numérica como lo muestra la figura 6b. Debido a esto, el muestreo regular unilateral es considerado el de mejor rendimiento en un FPGA.

Tabla 2. Uso de recursos en el FPGA

Recursos Lógicos utilizados									
Registros Slice	Slice LUTs	LUT-FF pairs	bonded IOBs	Bloques	RAM/ FIFO	BUFG/ BUFGCTRLs	DSP48E1s	Ciclos de Reloj	Tiempo a 33 [Mhz]
5160 (1.711%)	5160 (1.711%)	5160 (1.711%)	5160 (1.711%)	5160 (1.711%)	5160 (1.711%)	5160 (1.711%)	5160 (1.711%)	5160 (1.711%)	5160 (1.711%)

6. CONCLUSIONES

La TNL es una herramienta muy valiosa para el análisis de sistemas diversos, concretamente para el análisis de TEMs en redes de suministro de energía eléctrica. En este artículo se ha propuesto realizarlos mediante la implementación de la técnica de inversión numérica de la transformada de Laplace basada en el truncamiento por ventana de datos en un hardware de tipo FPGA.

Las pruebas aquí reportadas han mostrado que la implementación propuesta puede lograr velocidades de cómputo del orden de 360 veces mayores a las de los actuales simuladores comerciales de transitorios electromagnéticos en tiempo real. El método aquí propuesto es entonces apropiado para aplicaciones de coordinación de aislamientos mediante estudios estadísticos. También es adecuado para la computación de TEMs en tiempo acelerado; es decir, varias veces más rápido que tiempo real [22].

Con respecto al método propuesto, aquí se han probado cuatro variantes para la TNLI basada en ventaneo. Aunque los tiempos de ejecución de estas variantes en el FPGA no presentan grandes diferencias, las del muestreo regular requieren menores recursos de hardware; es decir, de un área de silicio menor en alrededor de un 10% y una mejor precisión (ver figura 6b). Por esto, en este artículo se concluye que el método de discretización regular unilateral es la mejor opción para la implementación de TNLI en un FPGA.

Finalmente, los autores de este artículo solo han encontrado una implementación de la TNLI en FPGAs reportada en la literatura especializada. Esta fue introducida en el año 2003 por Yonemoto et al. [11]. El progreso subsecuente en la tecnología de los FPGAs, así como de los métodos de solución, han motivado la investigación realizada en el presente trabajo.

ISSN: 1665-5745

7. REFERENCIAS

1. Moreno, Pablo, and Abner Ramirez. "Implementation of the numerical Laplace transform: A review" IEEE Transactions on power delivery, vol 23 No 4 pp 2599-2609, (2008).
2. Naredo, José L., et al. "Frequency domain aspects of electromagnetic transient analysis of power systems.", in "Transient Analysis of Power Systems: Solution Techniques, Tools and Applications", editor: Martinez-Velasco, Juan A., John Wiley & Sons pp 39-70, (2014).
3. Cohen, Alan M. Numerical methods for Laplace transform inversion. Vol. 5. Springer Science & Business Media, 2007.
4. Gómez, Pablo, and Felipe A. Uribe. "The numerical Laplace transform: An accurate technique for analyzing electromagnetic transients on power system devices." International Journal of Electrical Power & Energy Systems 31.2-3 (2009): 116-123.
5. Mestas, Patricia, and Maria Cristina Tavares. "Relevant parameters in a statistical analysis—Application to transmission-line energization." IEEE Transactions on Power Delivery 29.6 (2014): 2605-2613.
6. Wedepohl, L. M. "Power system transients: Errors incurred in the numerical inversion of the Laplace transform." Proc. of the 26th Midwest Symposium on Circuits and Systems. 1983.
7. Wilcox, D. J. "Numerical Laplace transformation and inversion." International Journal of Electrical Engineering Education 15.3 (1978): 247-265.
8. Y. Chen and V. Dinavahi, "FPGA-Based Real-Time EMTP," in IEEE Transactions on Power Delivery, vol. 24, no. 2, pp. 892-902, April 2009. doi: 10.1109/TPWRD.2008.923392.
9. Chen, Yuan, and Venkata Dinavahi. "Digital hardware emulation of universal machine and universal line models for real-time electromagnetic transient simulation." IEEE Transactions on industrial electronics 59.2 (2012): 1300-1309.
10. Matar, M., & Iravani, R. (2013). The reconfigurable-hardware real-time and faster-than-real-time simulator for the analysis of electromagnetic transients in power systems. IEEE Trans. Power Deliv, 28(2), 619-627.
11. Yonemoto, Akihiro, Takashi Hisakado, and Kohshi Okumura. "An implementation of numerical inversion of Laplace transforms on FPGA." Circuits and Systems, 2003. ISCAS'03. Proceedings of the 2003 International Symposium on. Vol. 4. IEEE, 2003.
12. Durbin, F. "Numerical inversion of Laplace transforms: an efficient improvement to Dubner and Abate's method." The Computer Journal 17.4 (1974): 371-376.
13. L. J. Castañón, J. R. Zuluaga and J. L. Naredo, "Numerical laplace inversion methods for electromagnetic transient simulations," 2016 North American Power Symposium (NAPS), Denver, CO, 2016, pp. 1-6.
14. Zuluaga, J. R., Rodríguez, C. H., Castañón, L. J., & Naredo, J. L. (2017, October). Sampling approaches for the numerical laplace transform and its FPGA implementation. In Electrical Engineering, Computing Science and Automatic Control (CCE), 2017 14th International Conference on (pp. 1-6). IEEE.
15. Savich, Antony W., Medhat Moussa, and Shawki Areibi. "The impact of arithmetic representation on implementing MLP-BP on FPGAs: A study." IEEE transactions on neural networks 18.1 (2007): 240-252.
16. IEEE Computer Society Standards Committee. Working group of the Microprocessor Standards Subcommittee. "IEEE standard for binary floating-point arithmetic." Institute of Electrical and Electronic Engineers, 1985.
17. Holt, Jordan L., and Thomas E. Baker. "Back propagation simulations using limited precision calculations." Neural Networks, 1991. IJCNN-91-Seattle International Joint Conference on. Vol. 2. IEEE, 1991.
18. LogiCORE, I. P. (2011). Fast Fourier Transform v7. Xilinx Corporation.
19. LogiCORE, I. P. (2011). CORDIC v4.0. Xilinx Corporation.
20. Muller, Craig. "User's Guide on the Use of PSCAD." Manitoba, Canada: Manitoba HVDC Research Centre (2010).
21. TM.I Gieshrech "Small Time-sten « 2uSec) VSC Model for the Real Time Digital Simulator," In Power System Transients (IPST), 2005 International Conference on, Canada, June 2005.
22. Herbordt, Martin C., et al. "Achieving high performance with FPGA-based computing." Computer 40.3 (2007).

ISSN: 1665-5745 www.e-gnosis.udg.mx