Hardware design of computer arithmetic blocks for engineering laboratory sessions.

J. Vázquez, R. Carrasco, J. Ortegón, A. Castillo, M.-I. Rocha-Gaso, V. Cabañas

Abstract — **This work presents a set of laboratory sessions that can be carried out by students of Digital Design courses required in engineering carrers like Electronics, Communications, Mechatronics, etc. The purpose of these lab is that the students develop their skills and confidence in the design of arithmetic hardware blocks by presenting them specific problems. In this sense, this paper shows a design methodology to be performed in a laboratory for the design of arithmetic blocks which can be implemented in microcontrollers and FPGAs. More specifically, we present the block design of a number's multiplicative inverse ($1/x$), its square root ($\sqrt{x}$) and the square root of its inverse ($1/\sqrt{x}$). The completion of these exercises requires the application of the Newton-Raphson algorithm, polynomial approximations of functions, difference equations and digital design. Students of our institution completed the lab sessions and after analyzing the results of student surveys and classroom observations, we found out that completing these tasks significantly contributed to the students' training in the hardware design field.**

Keywords—**Computing arithmetics, hardware design, laboratory sessions, digital design, engineering.**

I. Introducción

HOY EN DIA, las nuevas estrategias de enseñanza-aprendizaje que se desarrollan en el salón de clase, exigen que el alumno se enfrente a situaciones que se asemejen a condiciones reales de trabajo como profesional. Por ejemplo, deben plantearse situaciones a resolver, considerando el uso de todos los conocimientos acumulados (o saberes), el trabajo colaborativo, y en su conjunto estimular y desarrollar diversas actitudesy habilidades acorde a lo deseado en un plan de estudios en particular. En este sentido, el aprendizaje basado en problemas y en proyectos es ideal para usarse en diversas áreas de la ingeniería, facilitando la implantación de la estrategia de enseñanza-aprendizaje en el campo de la ingeniería aplicada. Lo anterior, permite dirigir el aprendizaje al tratamiento de un determinado problema, de interés para el futuro profesionista, y así lograr potencializar el impacto del conocimiento al generar emociones y motivación durante el desarrollo de la actividad.

Actualmente diversos trabajos bajo este paradigma de aprendizaje han sido presentados de manea exitosa, demostrando que uno de los factores más importantes para conseguirlo es el diseño cuidadoso de ejercicios que lleven al estudiante a resolver problemas de más básicos a más complejos de forma progresiva [Ref]. Por ejemplo, los autores en [1] presentan un conjunto de prácticas de laboratorio para realizar un procesamiento básico de imágenes mediante FPGAs. En [2] se presenta un laboratorio virtual para procesamiento de imágenes que busca dar comprensión al alumno de cómo diversos algoritmos de restauración y post-procesamiento contribuyen a mejorar la calidad en una imagen distorsionada. Por otra parte, en [3] se propone un prototipo de tarjeta digital basada en FPGAs para la implementación de laboratorios virtuales. Este tipo de laboratorios permite que los estudiantes mejoren sus habilidades en diseño digital básico y orientarlos a la solución de problemas del mundo real. De manera similar en [4] se propone realizar diseño de módulos que posteriormente, en su conjunto, sirven para integrar un solo sistema. También se han reportado aplicaciones basadas en web, por ejemplo, en [5] se propone una herramienta de trabajo colaborativo para procesamiento de imágenes basadas en FPGAs, mientras que, ejemplos de proyectos de laboratorio para aplicaciones de cifrado y descifrado de imágenes utilizando procesadores embebidos es presentado en [6].

Por otra parte, enfocándonos al profesionista como ingeniero diseñador de hardware, comúnmente le tocará desarrollar aplicaciones donde la solución total y absoluta no estará disponible, por lo que su tarea consistirá en brindar un conjunto de soluciones que den respuesta a una problemática particular. Una gran variedad de algoritmos de procesamiento digital de señales utilizan diversas funciones matemáticas que comúnmente no están disponibles en un procesador, microcontrolador o FPGA. Por ejemplo, en [7] se presenta el algoritmo para implementar una descomposición QR vía rotaciones de Givens, donde es necesario el cálculo de una raíz cuadrada o el recíproco de una raíz, según sea el caso. Esto toma relevancia, cuando es necesario realizar el cálculo de una división y el factor de división no puede expresarse en potencias de dos, lo que evita se pueda implementar con registros de corrimiento. En consecuencia, el diseño de este tipo de bloques en dispositivos FPGAs se debe hacer desde cero o, en su

Dr. J. Vázquez Castillo, Dr. J. Ortegón Aguilar, MTI. Vladimir Cabañas Victoria. División de Ciencias e Ingeniería, Universidad Quintana Roo, unidad Chetumal, México. (e-mail: {jvazquez@uqroo.edu.mx, jortegon@uqroo.edu.mx, vdrakul@uqroo.edu.mx).
Dra. M. I. Rocha Gaso. División de Ingeniería, Universidad Quintana Roo, unidad Cancún, México. (e-mail: mrocha@uqroo.edu.mx).
Dr. R. Carrasco Alvarez. Centro Universitario de Cienas Exactas, Universidad Autónoma de Guadalajara, México. (e-mail: r.carrasco@academicos.udg.mx).
Dr. A. Castillo Atoche. Departamento de Ingeniería Mecatrónica, Universidad Autónoma de Yucatán, México. (e-mail: acastill@correo.uady.mx).

defecto, comprarse como un bloque propietario, representando un posible costo adicional. En este sentido, esta limitación de los dispositivos procesadores hace que el algoritmo bajo interés no pueda ser desarrollado e implementado por completo y, por lo tanto, no se pueda brindar una solución a un determinado problema.

En este artículo se presenta una metodología de diseño para bloques de cálculo aritmético orientado a microcontroladores o FPGAs. Específicamente, se presenta el diseño de bloques del inverso de un número ($\mathbf{1/x}$), su raíz cuadrada ($\sqrt{\boldsymbol{x}}$) y del recíproco de su raíz ($\mathbf{1/\sqrt{x}}$). Esta metodología se lleva a cabo en el curso de Procesamiento Digital de Señales en la Universidad de Quintana Roo, México; sin embargo, se espera que la metodología propuesta también pueda llevarse como prácticas en los laboratorios de otras Universidades en las asignaturas de Diseño Digital, Sistemas Embebidos, Diseño de Hardware, Microcontroladores, entre otros.

II. Contexto de la asignatura de procesamiento digital de señales

El diseño de bloques de cálculo aritmético está contemplado para llevarse a cabo durante el desarrollo de la asignatura de *Procesamiento Digital de Señales* que se imparte en el noveno cuatrimestre de la Ingeniería en Redes en la Universidad de Quintana Roo, México. La asignatura se compone de 5 unidades distribuidas de la siguiente forma: Unidad I Introducción a Señales y Sistemas, Unidad II Señales y Sistemas en TiempoDiscreto, Unidad III Transformada Z, Unidad IV Transformada Discreta de Fourier y Unidad V Diseño de Filtros Digitales.

Los saberes del alumno, que serán de vital importancia para el desarrollo de la metodología de diseño de los bloques de cálculo aritmético, comienzan a adquirirse en la Unidad 2 en el subtema *Sistemas discretos en el tiempo descritos mediante ecuaciones en diferencia*. En este subtema se analizan diversos sistemas discretos en el tiempo tanto recursivos como no recursivos, además de que, se discute la gran importancia de implementar algoritmos recursivos para la solución de funciones a través del cómputo aritmético. Es pertinente resaltar que en este punto el alumno ya ha cursado asignaturas de métodos numéricos, sistemas digitales, microcontroladores entre otras relacionadas al diseño digital.

III. Descripción de la metodología propuesta

Con la finalidad de lograr el objetivo de este artículo, en esta sección se describe la metodología de diseño propuesta para el diseño de los bloques de cálculo aritmético. La propuesta consta de seis pasos, los cuales son descritos en las siguientes subsecciones.

A. Selección de la función a implementar como bloque de cómputo aritmético.

En este paso, el diseñador de hardware (HW) debe definir la función a ser aproximada mediante HW en base a los requerimientos del usuario. Para nuestro caso, y como trabajo en laboratorio, el alumno debe elegir entre funciones que pueden ser evaluadas de manera recursiva y que pueden calcularse mediante un algoritmo iterativo como lo es el algoritmo de Newton-Raphson (A-NR). Es decir, el alumno debe elegir entre diseñar el bloque $1/x, \sqrt{x}$, o $1/\sqrt{x}$.

B. Algoritmo de Newton-Raphson (A-NR)

En este paso, el alumno debe utilizar los conocimientos adquiridos en las unidades I y II de la asignatura para visualizar el bloque a diseñar en una versión algorítmica. Para nuestro caso, se utiliza el A-NR y asocia la implementación del bloque como un sistema de tiempo discreto, el cual será representado con ecuaciones en diferencia con coeficientes constantes.

En el método A-NR sirve para encontrar valores s que hacen cero una función $f(s)$ iterativamente. El A-NR es definido como sigue:

$$s_n = s_{n-1} - \frac{f(s_{n-1})}{f'(s_{n-1})} \tag{1}$$

donde s_n representará el valor a calcular en la iteración n, s_{n-1} es el cálculo de la iteración previa $n-1$, $f(s_{n-1})$ es una función construida a partir del valor s_{n-1} y $f'(s_{n-1})$ su derivada. La comprensión del algoritmo de NR ya fue abordado en la asignatura de Métodos numéricos (Matemáticas V) impartido en la Universidad de Quintana Roo y es un saber del alumno; sin embargo, un trabajo de investigación es indicado para reforzar el conocimiento adquirido.

Suponiendo que el alumno ha decidido implementar la función $s = 1/\sqrt{x}$ tenemos que, el algoritmo iterativo A-NR puede ser implementado como sigue:

Despejando x de $s = \frac{1}{\sqrt{x}}$, tenemos que:

$$x = \frac{1}{s^2} \tag{2}$$

Con la ayuda de (2), se define $f(s_{n-1})$ como:

$$f(s_{n-1}) = \frac{1}{{s_{n-1}}^2} - x = 0 \tag{3}$$

Derivando la ec. (3), se obtiene lo siguiente:

$$f'(s_{n-1}) = -\frac{2}{{s_{n-1}}^3} \tag{4}$$

Substituyendo (3) y (4) en (1) obtenemos la representación del A-NR para nuestra función objetivo:

$$s_n = s_{n-1} - \frac{\frac{1}{{s_{n-1}}^2} - x}{-\frac{2}{{s_{n-1}}^3}} = s_{n-1} - \frac{s_{n-1} - {s_{n-1}}^3 x}{-2}$$
$$= \frac{-2s_{n-1} - s_{n-1} + {s_{n-1}}^3 x}{-2} = \frac{2s_{n-1} + s_{n-1} - {s_{n-1}}^3 x}{2}$$

$$s_n = \frac{1}{2}(3s_{n-1} - {s_{n-1}}^3 x) \quad (5)$$

En este caso, el algoritmo recursivo puede ser representado como un sistema de ecuaciones en diferencia con coeficientes constantes, el cual puede ser fácilmente implementado como un filtro de respuesta infinita al impulso de la siguiente forma:

$$s(n) = \frac{1}{2}\,[3s(n-1) - s^3(n-1)\,x], \quad (6)$$

donde $s(n)$ es el valor calculado correspondiente al recíproco de la raíz de x y $s(n-1)$ es el valor calculado de x en la iteración previa del algoritmo. En el instante de tiempo $n = 0$, correspondiente al momento que el algoritmo inicia, el valor $s\,(-1)$ puede provenir de un generador de número aleatorios o ser un valor prestablecido cercano al valor que se busca. En nuestro caso, el primer valor del algoritmo será leído de una tabla que almacena las semillas para ciertos rangos del dominio de la función de interés.

Siguiendo el mismo procedimiento realizado previamente es posible encontrar las ecuaciones en diferencia para las funciones objetivos de $s = 1/x$ y $s = \sqrt{x}$. Sin embargo, estas funciones pueden ser obtenidas a partir del bloque que calcula $1/\sqrt{x}$ de la siguiente forma:

$$s = \frac{1}{\sqrt{x}} * \frac{1}{\sqrt{x}} = \frac{1}{x} \quad (7)$$

$$s = \frac{1}{\sqrt{x}} * x = \sqrt{x} \quad (8)$$

Es decir, con un multiplicador adicional y un multiplexor es posible tener un bloque completo que pueda realizar los tres cálculos aritméticos.

C. Generación de la semilla de cálculo del bloque de cómputo aritmético.

Al ser iterativo el A-NR, es indispensable contar con un bloque generador de semillas aleatorias o contar con un bloque que de un valor inicial a $s(-1)$ en el algoritmo. El bloque generador de semillas aleatorias puede lograse con algún esquema basado en registros de corrimiento (LFSR) [8] o por algún método sencillo basado en autómatas celulares con alguna regla de transición [9]. Sin embargo, el utilizar este tipo de bloques puede ocasionar que el A-NR no converja o no lo haga de manera rápida. Una solución más exacta, y de hecho más sencilla, es utilizar una tabla de datos, o look-up table (LUT), para almacenar algunas muestras de las funciones objetivo en las iteraciones del A-NR.

En nuestro caso de estudio, la cantidad de muestras relacionadas al dominio de $s = 1/\sqrt{x}$ y su valor calculado, dependerá del caso de estudio a realizar y de la exactitud de bit establecidos como requerimiento de usuario. Una LUT pequeña ayudará a cumplir con las restricciones de baja ocupación de área en HW; sin embargo, ocasionará que el resultado se obtenga después de varias iteraciones. Por otro lado, una LUT grande con muchas muestras almacenadas podrá lograr una convergencia rápida, pero con la restricción de que el bloque implementado en HW será también de área más grande.

Se sabe que, el A-NR tiene una convergencia cuadrática por lo que la exactitud de bit proporcionada por el evaluador polinomial para el valor de $s(-1)$ será doblada cuando pase a través del bloque del A-NR por cada iteración. Esto nos dice que no es indispensable contar con una resolución de LUT grande debido a que la implementación del A-NR alcanzará, con pocas iteraciones y buenos niveles de relación señal a ruido de cuantificación (SQNR del inglés signal-to-quantization-noise ratio).

D. Generación del modelo de oro del algoritmo elegido.

En esta etapa el alumno debe implementar el A-NR a través de un lenguaje computacional vía software (C++, Matlab, etc.), en el cual más adelante permita comparar los resultados de su algoritmo de alto nivel en punto flotante y punto fijo con los resultados obtenidos de su arquitectura implementada. Durante la implementación del modelo de oro, el alumno debe hacer un análisis profundo de la SQNR que le permita definir qué tan exacta será su implementación. Se recomienda tratar de lograr implementaciones con SQNRs mayores a 60 dBs. Así, la métrica para el análisis de ruido por cuantificación SQNR es definida como [10, p. 27]:

$$SQNR_{dB} = 10\log_{10}\left(\frac{\sum_{n=1}^{L} s(n)^2}{\sum_{n=1}^{L}(s(n)-\hat{s}(n))^2}\right), \quad (10)$$

donde L es el número de muestras del cual se compone el dominio de la función, $s(n)$ es la función evaluada en punto flotante y $\hat{s}(n)$ es la función evaluada en punto fijo con W_L bits de ancho de palabra, el cual se compone W_I bits enteros y $W_F = W_L$-W_I bits de parte fraccionaria.

E. Diseño de la arquitectura.

En esta etapa, el alumno realiza la propuesta de una arquitectura del bloque bajo diseño. El encargado de la asignatura asiste al alumno y define con él el proceso de diseño del bloque. El alumno diseña un bloque funcional correspondiente al bloque de cálculo aritmético elegido mediante el lenguaje de descripción de HW Verilog. El lenguaje de descripción de HW forma parte de los saberes del alumno ya que los adquiere en la asignatura de Diseño Digital impartido en cursos previos en la Universidad de Quintana Roo.

F. Verificación de la arquitectura diseñada.

Esta etapa es una de las más importantes antes de liberar un diseño. En este sentido, el alumno debe verificar su diseño implementado una cama de pruebas, la cual debe inyectar valores aleatorios al bloque diseñado y comparar los resultados con los generados con el modelo de oro previamente

implementado. Si los resultados coinciden con los resultados del cálculo en punto fijo en software *vs.* hardware, el diseño es aprobado. De lo contrario, el alumno debe realizar un análisis del diseño de tipo top-down hasta lograr detectar en qué nivel se encuentra su error para poder corregirlo.

IV. Casos de estudio y resultados: Diseño de bloques de cálculo aritmético

En esta sección se presenta el análisis de resultados obtenidos cuando la metodología propuesta es validada mediante el diseño de bloques de cómputo aritmético y mediante la propuesta de solución a un problema nuevo por parte del alumno. La validación experimental consistió en aplicar la metodología de diseño en una práctica de laboratorio y posteriormente reutilizar el HW diseñado en una segunda práctica de laboratorio para implementar un bloque de cómputo de mayor complejidad, el cual no está disponible como bloque convencional en microcontroladores o FPGAs.

A. Descripción de los laboratorios.

Práctica 1 (*4 horas*): Diseño de un bloque base:

Implementación del bloque de cómputo aritmético correspondiente a la función $s = 1/x$. El bloque diseñado deberá tener una SQNR de 60 dBs y no se tienen restricciones en el ancho de palabra de la arquitectura para alcanzar el desempeño de SQNR.

Esta práctica de laboratorio busca que el estudiante plantee el sistema recursivo resultante mediante la aplicación de la metodología propuesta. El alumno debe retomar los saberes del curso de diseño digital para proponer el diseño de un filtro de respuesta infinita al impulso (IIR) y realizar un análisis en punto fijo garantizando la SQNR establecida como requerimiento base. Así mismo, el estudiante tiene la libertad de plantear las características y técnica de la arquitectura del bloque generador de semillas. Una vez realizado lo anterior, el alumno debe proponer el diseño de la arquitectura y su camino de datos como resultado del estudio de un análisis de punto fijo.

Práctica 2 (*6 horas*): Diseño de bloques avanzados

Diseño de una arquitectura para calcular las funciones $g(x) = e^x$ y $g(x) = \ln(1 + x)$. El alumno basándose en el bloque de cómputo de la práctica uno, debe proceder a proponer una solución para el cálculo de las funciones requeridas mediante Series de Taylor. Las Series de Taylor nos dan la posibilidad de realizar diseño de bloques de cálculo adicionales haciendo uso de un módulo base $s = 1/x$ y etapas de multiplicación recursivas. En esta práctica de laboratorio el alumno debe realizar diversos análisis de punto fijo y precisión de bit ya que dependiendo del ancho del camino de datos de su diseño y del número de elementos de la serie que decida implementar, se puede lograr un determinado desempeño de SQNR.

V. Impacto de las prácticas de laboratorio

Las prácticas de laboratorio han sido aplicadas a un total de 38 estudiantes de dos generaciones (2014, 2015), lo cual nos ha permitido medir el grado de satisfacción de las prácticas de laboratorio y su impacto en el aprendizaje de la asignatura de Procesamiento Digital de Señales. En este sentido, se realizó un estudio estadístico mediante una encuesta a los equipos de trabajo formados para la realización de las prácticas de laboratorio. La Figura 1 muestra los resultados relacionados al nivel de complejidad de las dos prácticas de laboratorio en cuanto al diseño de los bloques de HW. Los resultados muestran que la mayoría de los estudiantes, el 68%, consideraron que la primera práctica de laboratorio fue de complejidad baja. De forma contraria, la segunda práctica de laboratorio fue considerada de nivel alto por un 53% de los estudiantes. Estos resultados eran de esperarse debido a que la segunda práctica de laboratorio involucra saberes que el alumno debe adquirir por su propia cuenta (en este caso Series de Taylor) y que lo pone en una situación muy similar a una condición de trabajo real como diseñador de HW.

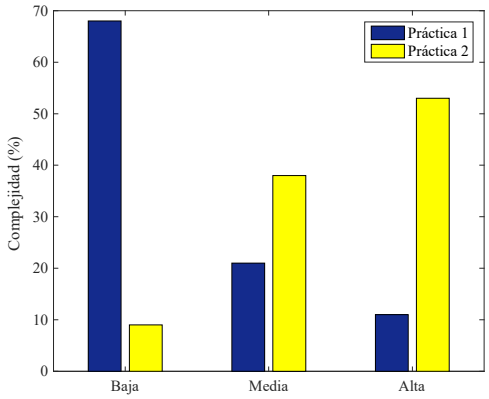


Figura 1. Evaluación de la complejidad computacional de las prácticas de laboratorio de cómputo aritmético.

En lo que respecta a los tiempos para realizar las prácticas de laboratorio, la Figura 2 nos da una idea general del tiempo necesario para finalizar la Práctica 1. A pesar de que la gran mayoría expresó que el nivel de complejidad fue bajo, la mayor parte de los alumnos requirieron de 2.5 horas a 4 horas para finalizar el diseño de la arquitectura requerida. Por otra parte, el tiempo de dedicación para realizar la práctica 2 de laboratorio (ver Figura 3) arrojó que un 54% de los alumnos requirió de hasta 7 horas para finalizar la práctica. Sin embargo, una gran población de estudiantes, el 44%, necesitó más de 7 horas para poder completarla.

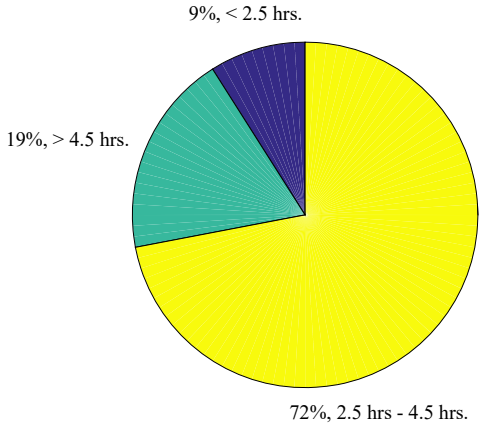


Figura 2. Evaluación del tiempo requerido para realizar la Práctica 1 del laboratorio.

Por otra parte, el grado de aceptación de la practicas de laboratorio fue expresado por los alumnos de manera explícita arrojando que un 92 % de ellos indicaron que las prácticas de laboratorio contribuyen a su aprendizaje y comprensión del tema "Sistemas discretos en el tiempo descritos mediante ecuaciones en diferencia" además de que contribuye al aprendizaje y diseño de filtros IIR.

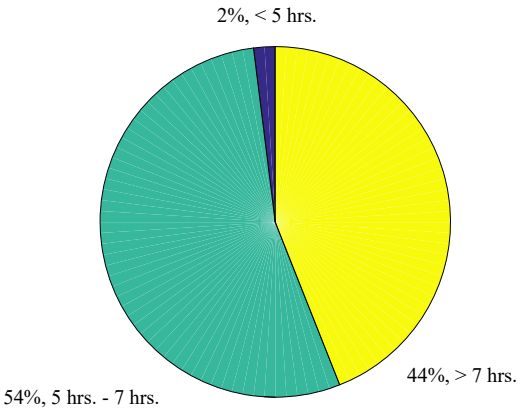


Figura 3. Evaluación del tiempo requerido para realizar la Práctica 2 del laboratorio.

VI. Conclusiones

La enseñanza de sistemas embebidos suele ser muy compleja, puesto que requiere conocimientos muy variados y el cambio en las tecnologías es acelerado. Por ello, es esencial el diseño de prácticas de laboratorio que faciliten al alumno el aprendizaje en este campo, dándole confianza y reforzando sus conocimientos.

En este trabajo se presentó una metodología para diseñar bloques de cálculo aritmético orientado a microcontroladores o FPGAs. La metodología de diseño fue aplicada a alumnos de la asignatura de Procesamiento Digital de Señales para diseñar diversos bloques de cómputo. La finalidad de la metodología aplicada busca mejorar las habilidades de los alumnos en la solución de problemas de diseño enfrentándolo a nuevas situaciones como diseñador de HW. La realización de los laboratorios planteados, refuerza y pone en práctica sus saberes relacionados a matemáticas, diseño electrónico, arquitectura de sistemas digitales (microcontroladores) y métodos numéricos. La aceptación por parte de los estudiantes es alta, ya que consideran que contribuyen a su aprendizaje y los lleva a enfrentarse a situaciones similares a las del campo laboral, donde pueden aprovechar conocimientos y desarrollos previos en la solución de una problemática particular.

Referencias

[1] A. Castillo Atoche, J. Vazquez Castillo, J. Ortegon Aguilar, y C. Rodriguez Cruz, "Laboratory Projects for Engineering Students with FPGA", *IEEE Lat. Am. Trans.*, vol. 6, núm. 2, pp. 130–136, jun. 2008.
[2] A. Castillo Atoche, J. Ortegon Aguilar, J. Vazquez Castillo, y J. Rivera, "Virtual Laboratory for Digital Image Processing", *Lat. Am. Trans. IEEE Rev. IEEE Am. Lat.*, vol. 12, núm. 6, pp. 1176–1181, sep. 2014.
[3] D. Garijo Mangas y R. Senhadji Navarro, "ccLAB: A Tool for Remote Verification of FPGA-based Circuits", *IEEE Lat. Am. Trans.*, vol. 14, núm. 3, pp. 1115–1121, mar. 2016.
[4] G. E. Blanco Silva, J. Rodriguez Resendiz, E. Gorrostieta Hurtado, J. C. Pedraza Ortega, y J. M. Ramos Arreguin, "Didactic Platform for Image Processing Experiments based on Digital Design", *IEEE Lat. Am. Trans.*, vol. 13, núm. 10, pp. 3398–3404, oct. 2015.
[5] I. Garcia, E. Guzmán-Ramírez, y C. Pacheco, "CoLFDImaP: A web-based tool for teaching of FPGA-based digital image processing in undergraduate courses", *Comput. Appl. Eng. Educ.*, vol. 23, núm. 1, pp. 92–108, ene. 2015.
[6] A. Kumar, S. Fernando, y R. C. Panicker, "Project-Based Learning in Embedded Systems Education Using an FPGA Platform", *IEEE Trans. Educ.*, vol. 56, núm. 4, pp. 407–415, nov. 2013.
[7] L. Canche Santos, A. Castillo Atoche, J. Vazquez Castillo, O. Longoria-Gándara, R. Carrasco Alvarez, y J. Ortegon Aguilar, "An improved hardware design for matrix inverse based on systolic array QR decomposition and piecewise polynomial approximation", 2015, pp. 1–6.
[8] S. L. Anderson, "Random Number Generators on Vector Supercomputers and other Advanced Architectures", *SIAM Rev.*, vol. 32, núm. 2, pp. 221–251, jun. 1990.
[9] P. D. Hortensius, R. D. McLeod, y H. C. Card, "Parallel random number generation for VLSI systems using cellular automata", *IEEE Trans. Comput.*, vol. 38, núm. 10, pp. 1466–1473, oct. 1989.
[10] J. G. Proakis y D. G. Manolakis, *Tratamiento digital de señales*. Madrid: Prentice-Hall, 1997.

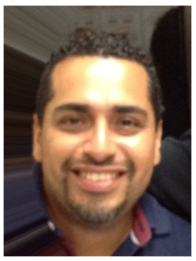
Javier Vázquez Castillo es Doctor en Ciencias por el Centro de Investigación y de Estudios Avanzados del I.P.N. (2014) en México con la especialidad en telecomunicaciones, recibió el grado de Maestro en Ciencias por la misma institución en 2002 e Ingeniero en Electrónica con especialidad en sistemas digitales por el Instituto Tecnológico de Mérida, México (2000). Actualmente es profesor de la Universidad de Quintana Roo. Sus áreas de interés son: diseño en microelectrónica digital, tratamiento digital de señales, cómputo aritmético y diseño de dispositivos para telecomunicaciones.

Roberto Carrasco Alvarez nació en la ciudad de México en 1981. Cursó sus estudios universitarios en el Instituto Tecnológico de Morelia obteniendo el título en ingeniero electrónico en 2004. Llevo a cabo sus estudios de maestría y doctorado en el Centro de Investigación y Estudios Avanzados del Instituto Politécnico Nacional (CINVESTAV), en el área de telecomunicaciones, obteniendo los grados en los años 2006 y 2010 respectivamente. Ha sido profesor asistente en el CINVESTAV y profesor de asignatura en el Instituto Tecnológico de Estudios Superiores de Occidente (ITESO). Actualmente es profesor investigador en el Centro Universitario de Ciencias Exactas e Ingenierías de la Universidad de Guadalajara. Sus líneas de investigación incluyen el procesamiento digital de señales y las comunicaciones digitales.

Jaime Silverio Ortegón Aguilar es Doctor en Ciencias por el Centro de Investigación y de Estudios Avanzados del I.P.N. (2007) con la especialidad en Ingeniería Eléctrica, Maestro en Ciencias por el mismo centro (2002) en la especialidad en Ciencias de la Computación, Licenciado en Ciencias de la Computación por la Facultad de Matemáticas de la Universidad Autónoma de Yucatán, México (2000). Actualmente es profesor de la Universidad de Quintana Roo. Sus áreas de interés son: tratamiento digital de señales, visión por computadora y diseño de dispositivos para telecomunicaciones.

Alejandro Arturo Castillo Atoche obtuvo el grado Doctor en Ciencias (2010) y de Maestro en Ciencias (2002) por el Centro de Investigación y de Estudios Avanzados del I.P.N. en México, y es Ingeniero en Electrónica del Instituto Tecnológico de Mérida, México (2000). Actualmente es profesor en la Universidad Autónoma de Yucatán en el departamento de Mecatrónica. Sus áreas de interés son: aplicaciones para el procesamiento inteligente de imágenes/señales, sistemas en tiempo real, sistemas embebidos, sistemas multiprocesador en un chip (MPSoC), co-diseño HW/SW y cómputo paralelo con GPUs, entre otras.

María Isabel Rocha Gaso recibió su título de Ingeniera en Computación de la Universidad Nacional Autónoma de México en el año 2006. En 2008, recibió un diploma de especialización en Ingeniería Bioelectrónica de la Universidad Politécnica de Valencia, España. En 2013, esta misma universidad, junto con la Universidad Católica de Louvain, Bélgica, le otorgó el título de Doctora en Ciencias de la Ingeniería con especialidad en electrónica. En 2014, realizó un postdoctorado en la Universidad de Lorraine, Francia, como investigadora del CNRS. Actualmente es Profesora Investigadora en la Universidad de Quintana Roo, Cancún, México. Dentro de sus intereses en investigación se incluyen: sensores, FPGAs, bioelectrónica, biosensores, microfabricación, sensores acústicos, MEMs, transductores piezoeléctricos y aplicaciones, adquisición, tratamiento y procesamiento de datos.



Vladimir Veniamin Cabañas Victoria es Maestro en Tecnologías de Información por el Instituto Tecnológico y de Estudios Superiores de Monterrey. (2005) con la especialidad en seguridad informática. Ingeniero en Sistemas Computacionales por el Instituto Tecnológico de Acapulco. (2001), con especialidad en redes de computadoras. Actualmente es profesor en la Universidad de Quintana Roo. Sus áreas de interés son telecomunicaciones y seguridad informática.