Revista Politécnica ISSN 1900-2351 (Impreso), ISSN 2256-5353 (En línea), Año 16, Número 32, páginas 68-76, Julio-Diciembre 2020

DOI: https://doi.org/10.33571/rpolitec.v16n32a6

MODULO IP BASADO EN FPGA PARA LA DECODIFICACIÓN DE ENCODERS DE CUADRATURA

Andrés-David Suárez-Gómez [1], Wilson-Javier Pérez-Holguín [2]

[1] Magister en Ingeniería con énfasis en Ingeniería Electrónica, Joven Investigador Colciencias, andresdavid.suarez@uptc.edu.co
[2] Doctor en Ingeniería énfasis en Ingeniería Electrónica, Docente de Planta, wilson.perez@uptc.edu.co
[1,2] Grupo de Investigación en Robótica y Automatización Industrial GIRA, Universidad Pedagógica y Tecnológica de Colombia

RESUMEN

Los encoders de cuadratura son los sensores más ampliamente utilizados para medir la posición, velocidad y aceleración de motores. Para su correcto funcionamiento se requiere la implementación de circuitos de decodificación basados en el conteo de pulsos y la determinación de la dirección de rotación, que consideren el uso eficiente de recursos y la eliminación de errores. Dichos circuitos se implementan usualmente en microcontroladores; sin embargo, las FPGAs tienen características que las hacen ideales para este tipo de aplicaciones. En este trabajo se presenta un módulo IP para el acondicionamiento de cuatro encoders de cuadratura, que genera valores de conteo para cada encoder en una base de tiempo que puede ser determinada interna o externamente y elimina los errores que se presentan en los cambios de dirección de rotación. Se presentan los resultados de simulación de la descripción de hardware y su implementación en un robot móvil basado en FPGA.

Palabras clave: Encoder de cuadratura; IP personalizada; FPGA; robótica.

Recibido: 4 de Junio de 2020. Aceptado: 14 de Diciembre de 2020
Received: June 4, 2020. Accepted: December 14, 2020

FPGA-BASED CUSTOM IP FOR QUADRATURE ENCODERS DECODING

ABSTRACT

Quadrature encoders are the most widely used sensors to measure the position, speed and acceleration of motors. For its proper functioning, it is required the implementation of decoding circuits based on the pulse count and the determination of the direction of rotation of the motor shaft, considering the efficient use of resources and the elimination of errors. Such circuits are usually implemented in microcontrollers-based or microprocessors-based platforms. However, FPGAs have characteristics that make them ideal for this type of application. In this work, a custom IP for the conditioning of four quadrature encoders is presented, which generates count values for each encoder on a time base that can be determined internally or externally and eliminate the errors that occur in the changes of the direction of rotation. The simulation results of the hardware description and its implementation in an FPGA-based mobile robot are presented.

Keywords: *Quadrature encoder; custom IP; FPGA; robotics.*

Cómo citar este artículo: A. Suárez, W. Pérez. "Modulo IP basado en FPGA para la decodificación de encoders de cuadratura", Revista Politécnica, vol.16, no.32 pp.68-76, 2020. DOI:10.33571/rpolitec.v16n32a6

1. INTRODUCCIÓN

El desempeño de un sistema de control depende en gran medida de los sensores utilizados para medir las variables del sistema que se desea controlar. Las características más importantes a tener cuenta son la resolución, precisión y exactitud [1]. En aplicaciones de control donde se usan motores como actuadores, los encoders de cuadratura son ampliamente utilizados para realizar la medición de la posición de su eje [2],[3]. Con base en esta información se pueden obtener datos como la posición, velocidad y aceleración del sistema que son fundamentales para para el desarrollo de sistemas de control [1],[2]. Estos aspectos son importantes en campos como la robótica, en donde esta información es necesaria para determinar la posición de un robot móvil o la posición de las articulaciones de un manipulador, entre otras [4],[5].

Los encoders de cuadratura permiten medir la posición y dirección de rotación a partir de la información de dos canales –comúnmente llamados Canal A y Canal B–. El Canal A se encuentra desfasado 90° con respecto al Canal B (ver Figura 1). Para determinar la dirección de rotación se monitorea la transición de un canal con respecto al otro canal [1],[2],[7]. Esta información debe ser decodificada en un circuito de acondicionamiento normalmente llamado decodificador de cuadratura.

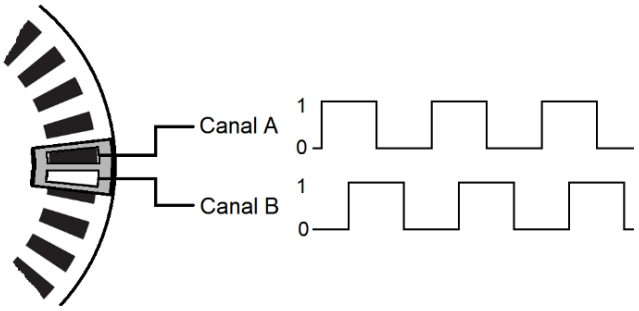

Figura 1. Señales de los canales A y B de un encoder de cuadratura.

El circuito decodificador de cuadratura se basa en un contador de alta velocidad y alta resolución que incrementa o disminuye su valor de acuerdo con la diferencia de fase entre los canales A y B, la cual depende de la dirección de rotación del eje del motor [7]. El circuito convencional usado para determinar la dirección de movimiento es un flip-flop tipo D. Este circuito presenta un error cuando ocurre un cambio en la dirección de rotación ya que no logra realizar la detección en el momento exacto del cambio. Lo mismo ocurre para una gran variedad de circuitos convencionales usados para la decodificación de las señales de este tipo de sensores [2].

Normalmente, el acondicionamiento de las señales de los encoders de cuadratura se hace por medio de microcontroladores o microprocesadores. No obstante, en los últimos años se ha visto un incremento en el uso de las FPGAs, gracias a que permiten integrar sistemas de hardware complejos y sus interfaces en un solo dispositivo [5].

Las FPGAs se usan en aplicaciones como procesamiento de imagen o audio, modulación o demodulación digital, procesamiento de datos, automatización e interfaces periféricas, entre otras. Las ventajas de este tipo de tecnología incluyen un corto tiempo de desarrollo, tamaño compacto, bajo precio, bajo consumo de energía, baja latencia, reconfigurabilidad, flexibilidad, escalabilidad y operación en tiempo real, entre otras [1],[2],[8].

A partir de estas características, es posible diseñar circuitos para aplicaciones de alto rendimiento, con bajos tiempos de procesamiento, operación paralela y aceleración en hardware [5]. Adicionalmente, en aplicaciones como el control de motores, es posible tener múltiples circuitos decodificadores y controladores en un solo chip (FPGA), sin comprometer el rendimiento del sistema [1],[7],[8].

El diseño de módulos de hardware para la decodificación de encoders de cuadratura basado esquemas de codiseño hardware/software, flexibiliza el desarrollo del sistema y mejora el poder de procesamiento, al descargar al procesador principal de tareas rutinarias y sencillas como las requeridas en este caso.

A pesar de las ventajas de las FPGAs para la implementación de circuitos de decodificación de encoders de cuadratura, se encuentran pocos trabajos reportados en la literatura centrados en su diseño y aplicación. En

[1] y [7] se presenta la implementación de un decodificador de cuadratura basado en máquinas de estado. En [1], se realiza una interfaz entre el decodificador y un procesador *soft-core* por medio de una UART. En [4]–[6] y [9], los autores implementan un algoritmo para la medición de velocidad y aceleración de un motor a través de encoders. En [2], Gorbounov realiza la implementación de un circuito decodificador de encoders de cuadratura que elimina los errores presentes en los cambios de dirección de rotación en el eje del motor. En [8], Aydogmus y Boztas presentan el diseño e implementación del control de velocidad de un motor en donde se incluye un bloque para el acondicionamiento de señales de encoders de cuadratura que solo usa los flancos ascendentes del canal A. En [3] se realiza la implementación en Nios II de un filtro previamente calculado en MATLAB para el control de velocidad de un motor usando encoders de baja resolución. La mayoría de estos trabajos usan los flancos ascendentes y descendentes de los canales A y B para multiplicar por cuatro la frecuencia de la señal empleada para el cálculo de la posición, con lo que se mejora la resolución del encoder.

La mayoría de trabajos en los que se describe la implementación de circuitos de acondicionamiento basados en FPGA no tienen en cuenta los errores que se presentan en los cambios de dirección de rotación del motor [2]. En dichos trabajos, el conteo del número de pulsos recibidos por un canal (A o B) es realizado a partir de la diferencia entre el valor actual y el valor anterior de conteo, dentro de una ventana de tiempo predeterminada. Los contadores tienen un numero de bits relativamente grande (16 o 32 bits por enconder), lo cual implica un considerable uso de recursos para aplicaciones en las que se requieran varios encoders.

En el presente trabajo se describe el diseño de un módulo IP (*Intellectual Property*) para el acondicionamiento de cuatro encoders de cuadratura. La salida corresponde a un bus paralelo que reporta el cambio de posición de los cuatro encoders en un tiempo determinado. Adicionalmente, el módulo corrige los errores que se presentan en los cambios de dirección de rotación del motor e incluye la posibilidad de fijar el tiempo de conteo internamente en la IP, o externamente desde otro módulo de hardware o procesador conectado al módulo diseñado.

2. MATERIALES Y METODO

La implementación en hardware del módulo IP propuesto se divide en tres fases: i) desarrollo de un circuito de adquisición de datos, ii) desarrollo del decodificador de cuadratura, y iii) integración de las dos fases anteriores, considerando su posterior comunicación con otros bloques de hardware o un procesador en el sistema. A continuación, se explica detalladamente el proceso de desarrollo de cada una de estas fases.

2.1. Circuito de adquisición de datos

El desarrollo del decodificador de cuadratura se basa en el circuito presentado en [2] que garantiza la eliminación de errores en los cambios de dirección de rotación. En la Figura 2 se presenta el diagrama del circuito implementado que realiza la operación de adquisición de datos del encoder tomando como entrada las señales de los canales A y B.

La primera etapa de este circuito corresponde a un bloque denominado MULTIPLICADOR DE FRECUENCIA Y DETECTOR DE ERROR (ver Figura 2a). En este bloque, los Flip-Flops D permiten generar una señal (sincrónica) que se encuentra ligeramente retrasada con respecto a la entrada. Este retraso es visto como una diferencia por las compuertas XOR (G1 y G2) conectadas a las salidas de los Flip-Flops D, lo que genera un pulso en cada uno de los flancos ascendente y descendente de las señales de los canales A y B, y produce las señales AP y BP, respectivamente. Estos pulsos son sumados en la compuerta OR (G6) para obtener la señal PULSO_X4, cuya frecuencia es cuatro veces la de los canales de entrada (ver Figura 3). Por otra parte, se usa un Flip-Flop JK para detectar la dirección de rotación. Esta dirección se determina teniendo en cuenta el estado de las señales de los canales A y B, así como el desfase de 90° entre éstas. Como salida se obtiene la señal DIR_ROTACION_CE, la cual contiene errores que serán corregidos en una etapa posterior.

La siguiente etapa corresponde al bloque de RESET DE DIRECCIÓN (ver Figura 2b), que consiste en la implementación de un *shift-register* por cada canal, cuyo reloj es la frecuencia cuadruplicada del bloque anterior. Cuando se produce un cambio de dirección de rotación se generan dos pulsos consecutivos en uno de los canales. Esta situación es detectada a través de las compuertas AND (G5 y G6). La salida de este bloque, denominada RESET_DIRECCIÓN, cambia exactamente en el momento en que se hace un cambio de dirección de rotación del motor [2].

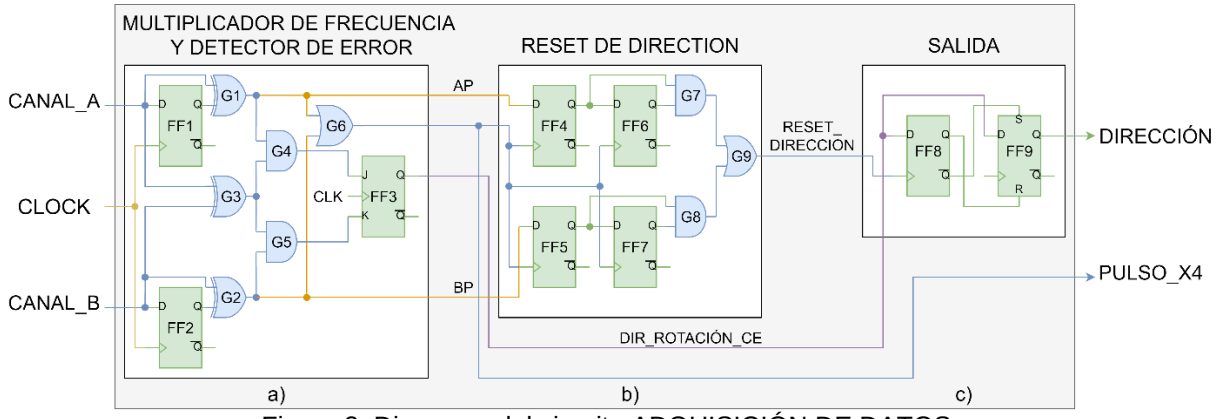

Figura 2. Diagrama del circuito ADQUISICIÓN DE DATOS.

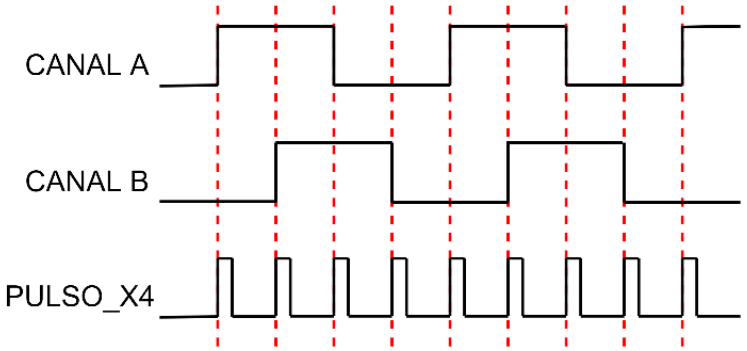

Figura 3. Multiplicación de frecuencia de los canales de un encoder de cuadratura.

La última etapa, denominada circuito de SALIDA, genera las señales: DIRECCIÓN que indica la dirección rotación de motor libre de errores en los cambios de dirección de rotación (0 para sentido horario y 1 para sentido antihorario) y la señal multiplicada en frecuencia PULSO_X4. Esto lo hace por medio de dos Flip-Flop D (FF8 y FF9). El primero trabaja de forma sincrónica tomando como señal de reloj la señal RESET_DIRECCIÓN, mientras que el segundo Flip-Flop trabaja de manera asíncrona con las señales Q y Q' del FF8 generando un 0 (Reset) o 1 (Set) en el momento exacto de cambio de dirección.

Cabe mencionar que en los circuitos convencionales de detección de dirección de rotación con encoders de cuadratura se presenta un error debido al retardo entre el momento de cambio de dirección y el flanco ascendente de los canales A y B [2]. Durante este intervalo de tiempo, se produce un conteo de pulsos en la dirección equivocada por parte del decodificador (ver Figura 4). La detección del momento exacto del cambio de dirección elimina este error y mejora el desempeño de los sistemas de control que gobiernan el motor.

2.2. Decodificador de cuadratura

El esquema general del DECODIFICADOR DE CUADRATURA (ver Figura 5) está conformado por tres bloques: el circuito de adquisición de datos (explicado anteriormente), un circuito anti-rebote y un contador ascendente/descendente. El circuito anti-rebote funciona como un filtro que elimina las oscilaciones que se presentan en los flancos ascendente y descendente de los canales A y B. El número de ciclos de reloj para la operación de anti-rebote puede ser elegido por el usuario. Sin embargo, se debe considerar el efecto que esto tiene sobre el retardo que se genera entre las señales de los canales A y B y las señales A_F y B_F (salidas del circuito de anti-rebote).

El CONTADOR cuenta a una frecuencia determinada por la señal PULSO_X4, en sentido ascendente/descendente determinado por la señal DIRECCIÓN. Adicionalmente, cuenta con las señales de entrada ENABLE y RESET para habilitar o resetear asíncronamente el conteo. La señal de salida denominada CONTEO, contiene la información de la posición del eje del motor codificada por el encoder de cuadratura, a partir de la cual se puede obtener el desplazamiento lineal o angular, la velocidad y la aceleración respectivas.

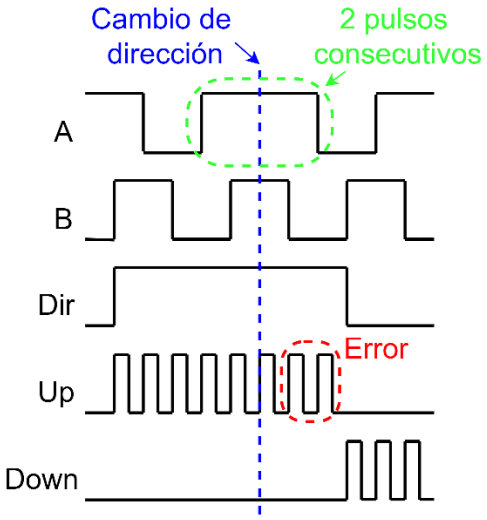

Figura 4. Error en el cambio de dirección de rotación en circuitos convencionales de detección de dirección de rotación con encoders de cuadratura.

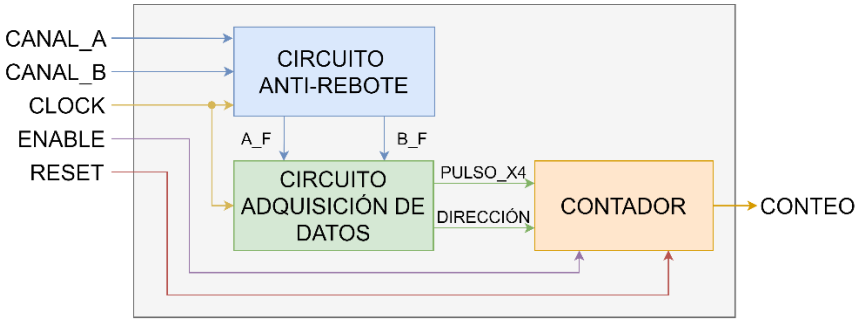

Figura 5. Diagrama del circuito DECODIFICADOR DE CUADRATURA.

2.3. Integración de la IP

Durante la etapa de integración se deben considerar los parámetros que garanticen la adecuada conexión de cada uno de los circuitos que conforman el módulo IP diseñado y la conexión de éste con otros bloques de hardware o con un procesador en el sistema (ver Figura 6). Para este caso, dado que la verificación del funcionamiento de la IP se hace sobre una FPGA de Altera, se deben tener en cuenta las características del bus Avalon y del procesador Nios II. Sin embargo, gracias a la descripción genérica realizada el módulo IP desarrollado puede ser verificado sin cambios sobre FPGAs de otros fabricantes.

El módulo IP integra cuatro decodificadores de cuadratura como los descritos anteriormente, por lo que puede ser empleado, por ejemplo, para determinar la posición y dirección de rotación de hasta cuatro motores empleados para la locomoción de un robot móvil. El usuario puede seleccionar el número (N) de bits para cada contador en el módulo. Las salidas de los contadores se asignan a un bus (READDATA) de 4*N bits de ancho, que contiene la información de los cuatro encoders.

Se debe tener en cuenta que la elección del número N de bits a ser empleado en los contadores puede afectar la velocidad de comunicación con el bloque receptor de los datos. Para el caso del procesador Nios II, si el valor 4*N es mayor que 32, se requiere realizar más de una operación de lectura del bus READDATA. En este diseño, los contadores realizan su operación durante un periodo de tiempo determinado por el usuario, al final del cual se resetean. Esto permite que el número N de bits a emplear sea reducido. A partir del periodo y el valor de conteo se puede determinar fácilmente la velocidad y aceleración correspondientes.

En módulo IP diseñado dispone de dos modos de operación para definir el periodo de conteo. En el primer modo (RESET ON TM), el usuario puede definir el periodo de conteo (en milisegundos) durante la instanciación de la IP a través de la interfaz en Qsys. En este modo se crea una señal de interrupción (INT_SENDER) que informa a los demás bloques de hardware que se cumplió el periodo establecido y se actualizó el valor escrito en el bus READDATA. En el segundo modo de operación (RESET ON READ), un bloque externo o

un procesador, puede establecer el periodo de conteo por medio de la señal READ_N, que le indica al módulo IP que se está realizando un proceso de lectura del bus READDATA. Una vez finalizada la lectura, se resetean los contadores y se inicia un nuevo proceso de conteo esperando por otra operación de lectura.

La implementación de los modos de operación requiere los circuitos TM y FSM (ver Figura 6). TM genera un pulso cuando se cumple el tiempo establecido en el modo RESET ON TM, mientras que el FSM toma la señal READ_N y la ajusta para que dure exactamente un ciclo de reloj cuando opera en el modo RESET ON READ. El circuito FORMATO DE DATOS toma las señales generadas por los bloques TM y FSM y, de acuerdo con el modo de operación, actualiza el valor del bus READDATA teniendo en cuenta la señal PERIODO_TM o la señal sREADN (dependiendo del modo de operación), luego de lo cual resetea los contadores con la señal RESET_CONTADOR.

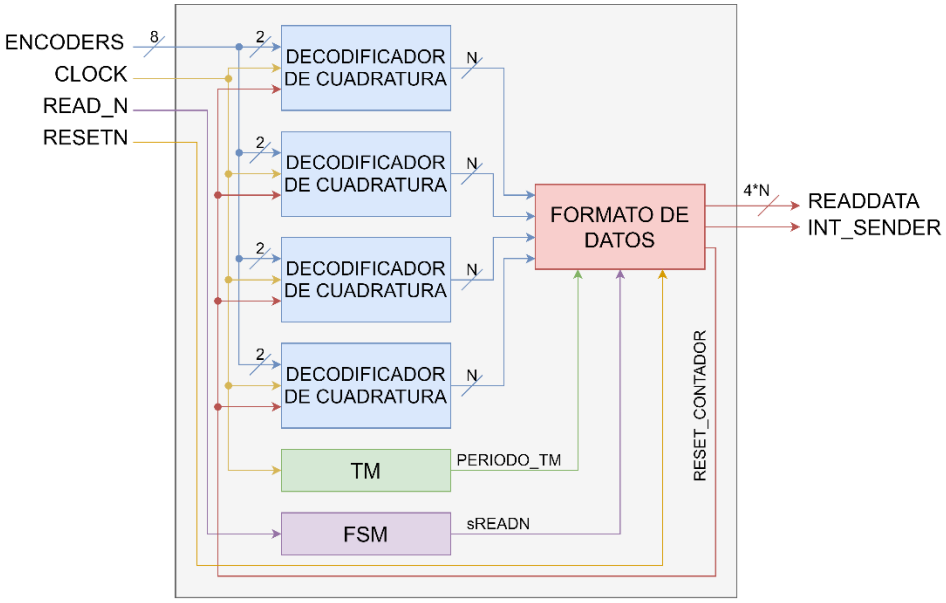

Figura 6. Diagrama de la IP para la decodificación de encoders de cuadratura.

3. RESULTADOS

La implementación del módulo IP desarrollado fue realizada en el software Quartus II v13.1 usando la herramienta Qsys. En la Figura 7 se presenta la interfaz de usuario para la inclusión de este módulo en un sistema determinado. La interfaz permite establecer la frecuencia de la señal de reloj, el periodo de la señal generada por el circuito TM, el modo de operación, el número N de bits de cada contador y el número de ciclos de reloj para la operación de anti-rebote.

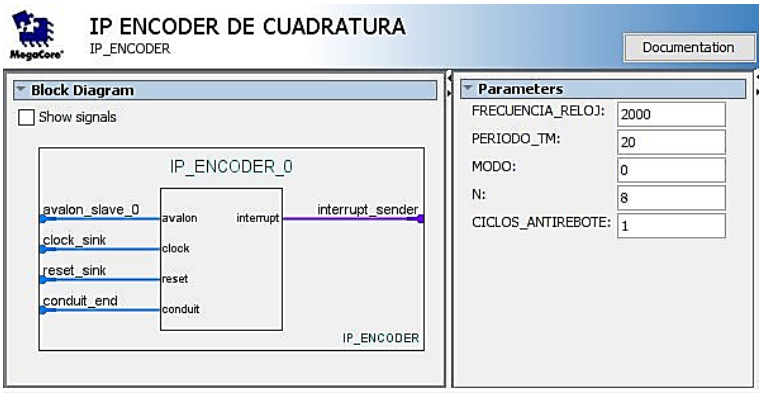

Figura 7. Interfaz de la IP en Qsys de Quartus II v13.1.

Cuando el parámetro MODO es 0, el modo de operación seleccionado es RESET ON TM. En este caso, el valor del bus READDATA se carga desde la salida del circuito TM, se resetean los contadores y se genera la señal INT_SENDER, que les indica a los demás circuitos que existe un nuevo dato en el bus. Cuando MODO

es 1, el modo de operación es RESET ON READ. En este modo, el bloque externo lee el bus READDATA cada vez que se cumple el periodo determinado, se genera un pulso en la señal READ_N y se resetean los contadores.

Los resultados de simulación para el módulo IP desarrollado se presentan en la Figura 8a para el modo de operación RESET ON TM y en la Figura 8b para el modo de operación RESET ON READ. Con el fin de facilitar la lectura de los resultados, el bus READDATA se divide en cuatro grupos que representan cada uno de los contadores.

En la Figura 8 se observa que en el modo RESET ON TM, la señal READ_N no tiene efecto sobre el bus READDATA y la señal INT_SENDER informa a los demás bloques que hay un nuevo dato disponible en el bus cada vez que se cumple el periodo de conteo. Por otro lado, en el modo RESET ON READ, cuando el módulo IP detecta que la señal READ_N pasa a estado lógico bajo se inicia una operación de lectura en el bus READDATA. Al final de esta operación, se mantiene el dato en el bus y se hace la decodificación esperando por una nueva operación de lectura. En este modo de operación, la señal INT_SENDER permanece siempre en estado lógico bajo, por lo que no se originan solicitudes de interrupción para los bloques externos o procesador conectados a la IP. Los resultados de simulación corroboran el correcto funcionamiento del módulo IP desarrollado.

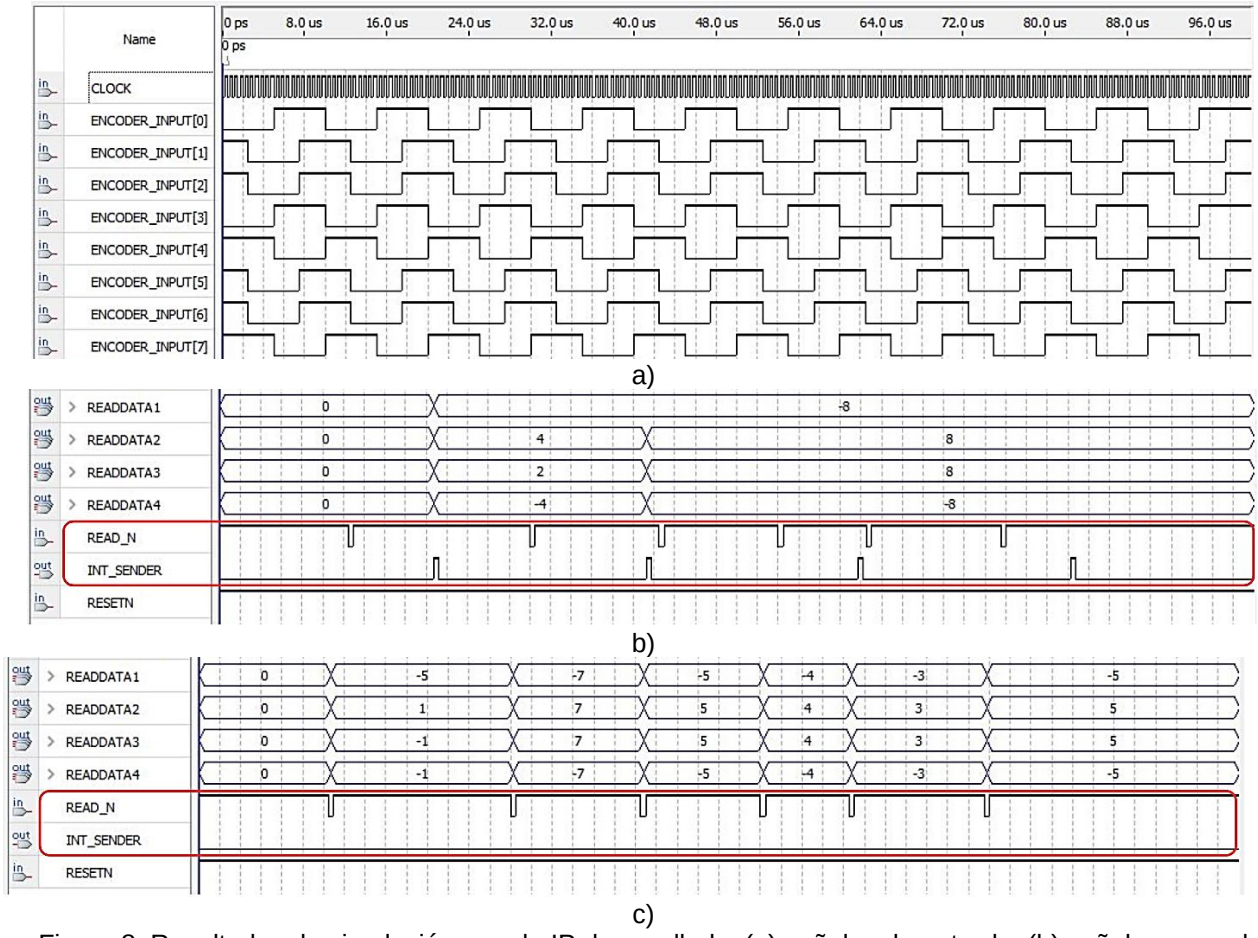

Figura 8. Resultados de simulación para la IP desarrollada, (a) señales de entrada, (b) señales para el modo RESET ON TM y (b) señales para el modo RESET ON READ.

Finalmente, el módulo IP diseñado es puesto a prueba en una aplicación de robótica móvil (ver Figura 9). Para esto se empleó una plataforma Rover 5 que dispone de dos encoders de cuadratura, una tarjeta XBee para comunicación inalámbrica con un PC que actúa como estación de monitoreo y es controlada mediante una tarjeta de desarrollo DE0-Nano de Altera®. La prueba consiste en tomar el valor de los contadores del módulo IP desarrollado y compararlos con la implementación en software del mismo conteo basado en inte-

rrupciones generadas por las señales provenientes de los encoders. El módulo IP es conectado a un procesador Nios II sintetizado en la tarjeta DE0-Nano, a través del cual se establece el periodo de conteo y el modo de operación (RESET ON READ). La información registrada se envía al PC por medio de la tarjeta XBee.

4. DISCUSIÓN

El módulo IP desarrollado presenta varias ventajas y novedades con respecto a otros trabajos presentados en la literatura, tales como: i) aborda los errores presentes en los cambios de la dirección de rotación, ii) se puede especificar el número de ciclos de reloj para la implementación del circuito de anti-rebote, iii) ofrece la posibilidad de definir el número N de bits para cada contador, el cual es significativamente menor al descrito en otros casos debido a que el conteo se resetea con base en un periodo de conteo que iv) puede ser definido por el usuario internamente (dentro de la IP) o externamente (por medio de otro módulo de hardware o un procesador).

La multiplicación por 4 de la frecuencia de los canales A y B permite mejorar la resolución del encoder. Como se observa en la Figura 9, mientras que la implementación en software con interrupciones produce un valor de conteo entre 15 y 16, el valor de conteo en la IP desarrollada se encuentra entre 62 y 64 para un periodo de conteo de 100 ms. Esto implica que el valor porcentual del error pasa de 6.25% (1/16) a 1.5625% (1/64).

Figura 9. Valor de conteo software vs. módulo IP.

Adicionalmente, se mejora el desempeño del sistema al disminuir por cuatro de la generación de solicitudes de interrupción en el modo RESET ON TM, o la total eliminación de estas en el modo RESET ON READ. Mientras que con la IP diseñada se presenta una interrupción cada vez que se cumple el periodo de conteo, en el caso de la implementación en software se presenta una interrupción en cada flanco ascendente de un canal (A o B) para cada encoder de cuadratura. Por ejemplo, en la implementación realizada para el Rover 5, en un periodo de conteo de 100 ms se tienen 16 interrupciones por cada encoder, para un total de 64 interrupciones, mientras que con la IP desarrollada se tiene una sola interrupción en el mismo periodo.

La descripción de hardware en VHDL para el módulo IP presentado se encuentra disponible en el siguiente enlace de GitHub: https://bit.ly/3mldKDy

5. CONCLUSIONES

En este trabajo se presenta el desarrollo de un módulo IP basado en FPGA para la decodificación de señales de encoders de cuadratura. Este módulo IP tiene en cuenta aspectos tales como: el aprovechamiento de las características de las FPGAs, el ahorro de recursos representado en el empleo de un menor número de bits N para la implementación de los contadores de cada encoder, la posibilidad de seleccionar del tiempo de conteo por parte del usuario (interna o externamente), la mejora en la resolución del encoder, la eliminación de errores en la decodificación durante los cambios de dirección de rotación y la flexibilidad de la interfaz del módulo IP a través de la cual se pueden especificar parámetros de operación tales como el tipo de operación (RESET ON TM o RESET ON READ), la frecuencia de reloj, el número de ciclos para hacer la operación de

anti-rebote y el número de bits a usar por cada. Todas estas características permiten que el módulo IP desarrollado pueda ser usado en una gran variedad de aplicaciones de manera sencilla. El correcto funcionamiento del módulo IP fue corroborado mediante simulaciones y por medio de su implementación en un robot móvil basado en FPGA controlado a través de un procesador Nios II.

6. AGRADECIMIENTOS

Los autores expresan su agradecimiento a MinCiencias y a la Universidad Pedagógica y Tecnológica de Colombia por la beca-pasantía de Joven Investigador, articulada con el proyecto con código interno SGI-2623.

7. REFERENCIAS BIBLIOGRÁFICAS

[1] S. Sarkar and K. Sivayazi, “A New Decoding Logic for Quadrature Encoder Interfacing for Software Implementation on FPGA,” in 2018 3rd International Conference for Convergence in Technology (I2CT), 2018, pp. 1–4.
[2] Y. Gorbounov, “Accurate Quadrature Encoder Decoding Using Programmable Logic,” J. Eng. Res. Technol., vol. 2, no. 4, 2016.
[3] H. I. V. Jadan, C. R. M. Vera, R. P. Intriago, and V. S. Padilla, “Implementing a Kalman Filter on FPGA Embedded Processor for Speed Control of a DC Motor Using Low Resolution Incremental Encoders,” in Proceedings of the World Congress on Engineering and Computer Science, 2016, vol. 1, pp. 367-371.
[4] K. Banerjee, B. Dam, and K. Majumdar, “An FPGA-based integrated signal conditioner for measurement of position, velocity and acceleration of a rotating shaft using an incremental encoder,” in 2016 IEEE First International Conference on Control, Measurement and Instrumentation (CMI), 2016, pp. 440–444.
[5] A. Hace and M. Čurkovič, “Accurate FPGA-based velocity measurement with an incremental encoder by a fast generalized divisionless MT-type algorithm,” Sensors, vol. 18, no. 10, p. 3250, 2018.
[6] J. Y. Wu, Z. Chen, A. Deguet, and P. Kazanzides, “FPGA-Based Velocity Estimation for Control of Robots with Low-Resolution Encoders,” in 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2018, pp. 6384–6389.
[7] C. Quintáns, J. Fariña, and J. Marcos-Acevedo, “Improving the performance of incremental encoders with conditioning circuits based on FPGA,” Measurement, vol. 90, pp. 1–3, 2016.
[8] O. Aydogmus and G. Boztas, “Implementation of an FPGA-Based Motor Control with Low Resource Utilization,” in 2019 4th International Conference on Power Electronics and their Applications (ICPEA), 2019, pp. 1–5.
[9] D. S. Debnath, M. Pal, K. Banerjee, B. Dam, and K. Majumdar, “An FPGA-based incremental encoder signal conditioner with reduced error in rotational rate estimation over a wide range of rotational speeds,” in 2016 2nd International Conference on Control, Instrumentation, Energy & Communication (CIEC), 2016, pp. 120–124.