Rev. Cienc. Tecnol.
Año 13 / Nº 15 / 2011 / 43–52

Procesador digital sincrónico en tiempo real soportado sobre un circuito fpga spartan3E.

Synchronous digital processor on real time supported in a fpga circuit spartan3E.

Juan Raúl Rodríguez Suárez, Ismel Domínguez Rodríguez

Resumen

En este trabajo se propone el diseño, implementación y comprobación de un procesador discreto sincrónico en tiempo real soportado sobre un circuito reconfigurable FPGA del tipo Xilinx Spartan-3E Starter Kit. Se realiza el diseño de un procesador digital sincrónico en tiempo real y se implementa sobre un circuito FPGA empleando la herramienta XSG que se instala en el Simulink de MATLAB. Se construyó además una tarjeta auxiliar DSP_Gidam01 que junto con la tarjeta Xilinx Spartan-3E contiene los componentes del sistema. Sin embargo los circuitos de control y de procesamiento digital se construyen sobre el circuito reconfigurable FPGA de la tarjeta. Se introduce brevemente la utilización del PROTEUS como herramienta para construir el PCB de la tarjeta auxiliar. Se evalúa la síntesis y funcionamiento del procesador con la implementación de determinados filtros digitales y la comparación entre las respuestas simulada y real. También se presenta una aplicación para procesamiento de audio con un banco de filtros.

Palabras clave: Circuito Reconfigurable, Procesador, Filtros digitales, Digitalización de audio, FPGA.

Abstract

This work intends the design, implementation and testing of a synchronous real time discrete processor supported on a reconfiguration circuit FPGA of the Xilinx Spartan-3E Starter Kit type. The design of a synchronous digital processor in real time is carried out and it is implemented on a circuit FPGA using the XSG tool that settles in the Simulink of MATLAB. An auxiliary card DSP_Gidam01 was also built that together with the card Xilinx Spartan-3E contains the components of the system. However, the control circuits and the digital processors are built on the FPGA reconfigurable circuit of the card. The use of the PROTEUS as tool is introduced for a short period, to build the PCB of the auxiliary card. The synthesis and operation of the processor is evaluated with the implementation of certain digital filters and the comparison among the simulated and real answers. An application is also presented for audio prosecution with a bank of filters.

Key words: Processor, Reconfiguration circuit, Digital filters, Audio digitalization, FPGA.

Introducción

El desarrollo de aplicaciones de procesamiento digital de señales soportadas sobre circuitos FPGA [1] se incrementa constantemente año tras año y va desplazando a los circuitos integrados estándar DSP. Estos procesadores tienen una arquitectura del tipo Harvard [2], caracterizada por emplear memorias separadas para datos y programas que se conectan a la unidad central de procesamiento por los buses respectivos, lo que permite que puedan ser accedidas simultáneamente y así realizar rápidamente las operaciones de sumas de productos que conforman el prototipo de las ecuaciones en diferencias que definen un sistema digital. Para realizar el procesamiento en tiempo real de un sistema, la salida o respuesta del mismo tiene que ser calculada mediante su ecuación en diferencias en un tiempo menor que la frecuencia de muestreo del sistema, antes de que la siguiente muestra de entrada arribe al procesador. Además los procesadores DSP operan a frecuencias de reloj de centenares de MHz a fin de ejecutar las operaciones matemáticas necesarias rápidamente y emplean también otras técnicas como el procesamiento paralelo (pipeline) que segmenta una instrucción en partes lo que permite que puedan ejecutarse simultáneamente varias instrucciones sin necesidad de esperar a que termine una para empezar la siguiente.

Por otra parte en un circuito FPGA se pueden implementar directamente las operaciones de procesamiento digital relativas a la ecuación en diferencia que definen dicho sistema, distribuyendo en el circuito los bloques que implementan dichas operaciones – sumadores, multiplicadores entre otros- y realizando las interconexiones necesarias.

Pueden implementarse fácilmente otras operaciones como retardos en tiempo, cambios de la frecuencia de muestreo, así como emplear bloques de memoria distribuida o segregada que facilitan el procesamiento digital de la señal que se desee. Este tipo de procesamiento único en su tipo es especialmente útil en aplicaciones donde se necesite altas velocidades de procesamiento, ya que las operaciones no se realizan en forma secuencial como en un procesador digital que posee una determinada unidad aritmética, sino en un procesamiento en paralelo o concurrente, ya que pueden sintetizarse los bloques aritméticos que se necesiten (sumadores, multiplicadores, registros) que están disponibles en un determinado circuito FPGA.

Los circuitos FPGA incrementan constantemente sus capacidades, al aumentar el nivel de integración de los mismos, incrementando la cantidad de sus recursos lógicos y de memoria. Para aplicaciones de procesamiento digital se incorporan bloques especiales como multiplicadores dedicados e incluso microprocesadores dedicados de altas prestaciones de 32 bits y módulos dedicados empotrados de comunicaciones digitales. Están disponibles además un considerable número de aplicaciones bien establecidas que pueden ser usadas y que se distribuyen libremente o son parte del sistema ofertado por el fabricante denominados módulos de propiedad intelectual IP [3], donde se destacan operaciones típicas del procesamiento digital como los filtros FIR y la transformada FFT entre otras. También se dispone de módulos IP de potentes microprocesadores de 32 bits como el MicroBlaze [4] para dispositivos de Xilinx, con unidad aritmética en punto flotante que pueden realizar aplicaciones de procesamiento digital de señales [5]. Un sistema de procesamiento digital puede así puede ser implementado mediante una configuración del tipo SoC, en el cual en un circuito FPGA se implementan diversos módulos IP, uno de ellos constituido por un microprocesador configurado por software que controla todo el sistema y se conecta con otros módulos IP o incluso módulos concebidos por el usuario que se configuran en el circuito y se conectan como periféricos a los buses del procesador.

Aunque la tarjeta Spartan3E Starter Kit de Xilinx [6] incluye un circuito FPGA xc3s500e, una memoria DDR externa, un convertidor ADC del tipo LTC-1407A y un convertidor DAC del tipo LTC2624 [7] entre otros, no es adecuada para una implementación de procesamiento digital de señales debido a que los convertidores comparten líneas comunes en sus terminales de control lo que imposibilita lograr la operación simultánea de los mismos y así producir el flujo de datos característico de un procesador digital en tiempo real. Las aplicaciones estándar reportadas para esta tarjeta comprenden así la operación de sólo uno de estos convertidores. Por otra parte esta tarjeta no posee los circuitos de filtros necesarios para instrumentar los filtros anti-réplicas e interpoladores que son imprescindibles para el tratamiento correcto de la señal a digitalizar [8].

El objetivo de este trabajo es presentar la implementación de un procesador digital sincrónico en tiempo real sobre la base de la tarjeta Spartan3E Starter Kit y evaluar el comportamiento del mismo. La solución propuesta incluye la construcción de una tarjeta auxiliar con un nuevo convertidor ADC y los circuitos de filtrado anti-réplica y de filtrado de recuperación .Empleando además el convertidor original LTC2624 disponible en la tarjeta Spartan 3E se puede entonces implementar un procesador digital sincrónico en tiempo real.

Materiales y métodos

En esta sección se presenta el diseño de la tarjeta auxiliar para implementar un convertidor ADC adicional y los filtros anti réplicas y recuperadores así como los procedimientos y características de los circuitos sintetizados en la FPGA para implementar el procesador digital de señales.

Diseño de la tarjeta auxiliar

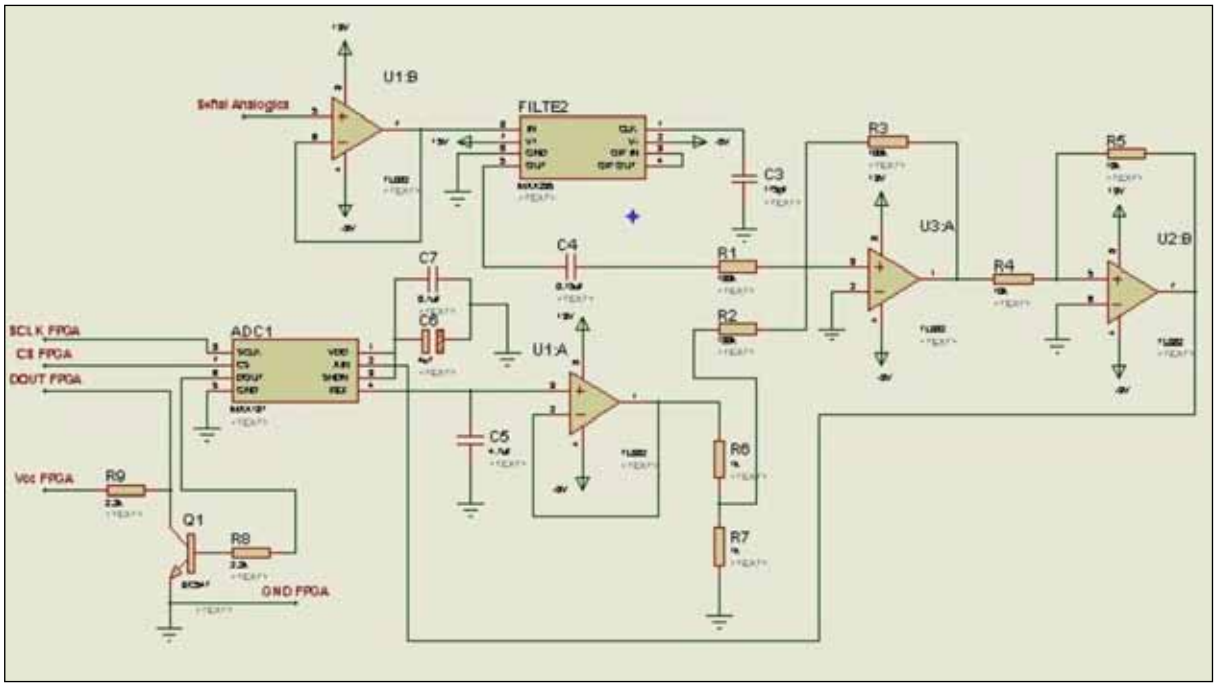
La tarjeta auxiliar diseñada contiene un convertidor ADC con sus líneas de control independientes al convertidor DAC LTC2624 disponible en la tarjeta inicial lo que permite entonces lograr la operación simultánea de los mismos a fin de lograr el flujo de muestras del procesamiento, y los filtros analógicos para las operaciones anti-réplicas y de interpolación. Se seleccionó el convertidor ADC MAX187 [9], que es también un convertidor de 12 bits, unipolar en formato serie.

Para implementar los filtros analógicos anti-réplicas e interpolador se seleccionó el circuito MAX295 [10], que es un filtro paso bajo del tipo Butterworth de orden 8 con una frecuencia de corte que puede ser programada mediante el control de una señal de reloj que posea una frecuencia de 50 veces la frecuencia de corte necesaria. Así ambos filtros MAX295 se programan con una frecuencia de corte de 4 kHz si el sistema va a operar a una frecuencia de muestreo de 8 kHz, por lo que el circuito FPGA debe poseer un bloque de generaración de una señal de 200 kHz que debe ser aplicada a la entrada de reloj de cada uno de los filtros empleados.

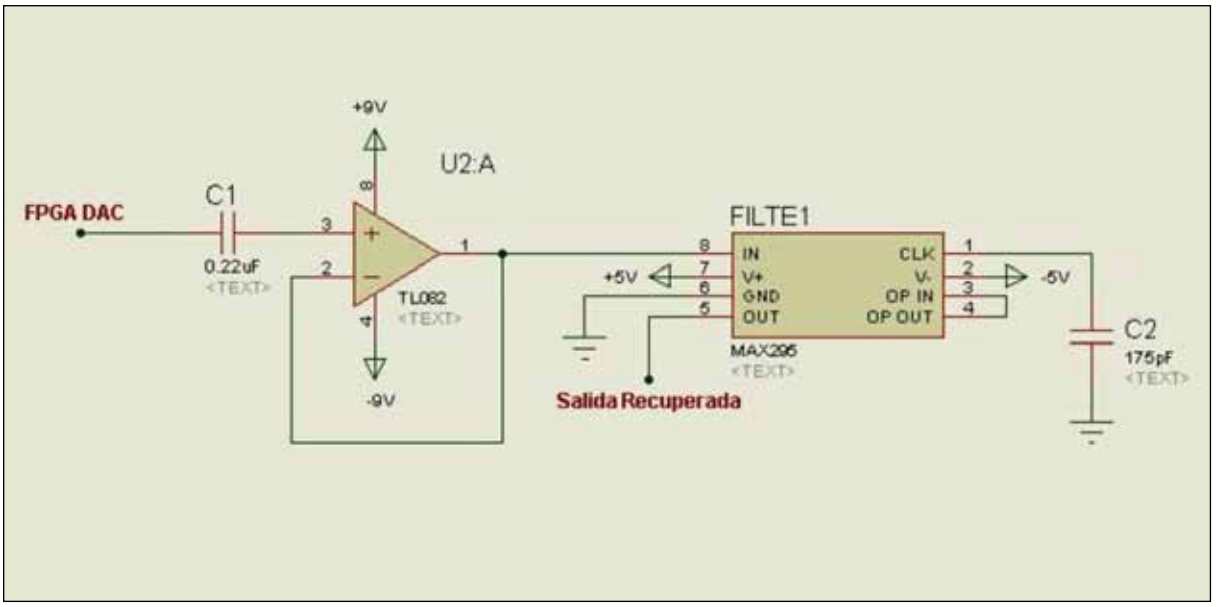
Por cuanto los convertidores empleados tienen el formato unipolar, para procesar señales bipolares se hace necesario instrumentar circuitos de acondicionamiento de la señal. Así para manejar una señal de entrada bipolar se implementa un circuito para desplazar el nivel a la mitad del nivel de referencia del convertidor MAX187 (4.096 V), por lo que los valores positivos de la entrada se detectan como un voltaje mayor que 2.048 V y viceversa.

En el caso de las muestras a la salida se suprime la componente CD con un condensador de bloqueo y se aplica finalmente el filtro paso bajo recuperador. Esto permite medir señales bipolares de hasta 1.25 V de amplitud en el sistema.

En la figura 1 se muestra el diagrama del circuito de la



a)



b)

Figura 1. Bloques de la tarjeta auxiliar para acondicionar la señal a) del convertidor ADC y b) del convertidor DAC.

tarjeta auxiliar diseñada. En resumen, la señal bipolar de entrada pasa por el filtro anti-réplicas a fin de atenuar las componentes con frecuencias mayores a 4 kHz, se desplaza de nivel y se digitaliza por el convertidor ADC. Las señales de control del convertidor ADC y el reloj del filtro anti-réplicas se reciben del circuito FPGA y se envía hacia éste la secuencia de datos serie del terminal de salida DOUT del convertidor ADC. Después de procesada la señal en la FPGA, el flujo de datos de salida se envía al convertidor DAC a fin de obtener una secuencia de voltajes que son enviados hacia el filtro interpolador a fin de reconstruir la señal análogica a la salida a partir de dichas muestras.

Diseño y síntesis del procesador digital

La programación de un circuito FPGA tiene el objetivo de implementar un circuito lógico determinado que se necesite a partir de los bloques lógicos disponibles en dicho circuito FPGA. Este circuito deseado se describe mediante un programa confeccionado con un determinado lenguaje de descripción de hardware y se emplea además un entorno integrado de desarrollo ofertado por el fabricante del circuito FPGA que permite realizar una serie de operaciones que comprenden de forma general la síntesis del circuito que define los bloques lógicos elementales que lo formarán, la selección de los bloques lógicos que se usarán y sus interconexiones , la simulación funcional del circuito a fin de determinar si cumple con las restricciones establecidas y finalmente la implementación del circuito que se hace

mediante la confección del fichero de datos que permite la programación del hardware. Para los circuitos Xilinx se emplea el entorno de desarrollo ISE. Sin embargo el uso de los lenguajes de descripción de hardware es precisamente una de las limitantes que restringe en cierta manera el empleo de los circuitos FPGA sobre todo debido a que se requiere de cierto entrenamiento y filosofía de trabajo que difiere de la programación estándar de los microprocesadores o microcontroladores. Ello es debido sobre todo al hecho de que un circuito FPGA el procesamiento es en paralelo o concurrente y no secuencialmente como es en los microcontroladores .

Sin embargo en este trabajo se ha empleado la herramienta XSG [11] que ha sido desarrollada para implementar aplicaciones de procesamiento digital de señales en circuitos FPGA de Xilinx y que permite instrumentar dichas aplicaciones directamente dentro del ambiente del programa Matlab y su aplicación Simulink [12]. El uso de esta herramienta facilita que diseñadores de aplicaciones de procesamiento digital de señales que tradicionalmente tienen experiencia en simulación con Matlab puedan aplicar casi directamente circuitos FPGA sin un conocimiento profundo de la programación en lenguaje de descripción de hardware pues XSG genera automáticamente un fichero en este lenguaje a partir de los bloques de la biblioteca disponibles en Simulink y realiza todo el proceso de síntesis e implementación llamando al entorno de desarrollo ISE de Xilinx dentro del propio ambiente de trabajo de Matlab obteniéndose finalmente el fichero de configuración para implementar físicamente el circuito deseado en la FPGA. Esta herramienta posee la ventaja que permite primeramente obtener el circuito deseado a partir de la simulación en Matlab-Simulink y entonces como un paso final la síntesis y generación del mismo. En el flujo de diseño original a partir de un lenguaje de descripción de hardware con entorno de desarrollo, primero es necesario concebir una versión inicial del circuito, sintetizarlo y entonces comprobar si cumple la función deseada.

Primeramente se desarrolló una aplicación básica conformada por tres bloques: la generación del reloj para los filtros MAX295, el bloque de interfaz ADC y el bloque de interfaz DAC. A fin de poder comprobar la funcionalidad de estos bloques cuando se aplique una señal bipolar de entrada al sistema debe resultar en una señal analógica de salida completamente idéntica a la entrada. Posteriormente se introduce un cuarto bloque para generar el filtro digital que se desee aplicar a fin de modificar las características de la señal de entrada.

Bloque de reloj para filtros

El bloque de generación del reloj para los filtros MAX295 se muestra en la figura 2, es un bloque compuesto por un contador ascendente, sin signo, que divide la señal de reloj de 50 MHz disponible en la Spartan3E y un bloque que compara la salida del contador con una constante que define el ciclo útil del reloj a 50%. Para una frecuencia de corte de 4 kHz, se debe de generar una señal de 200 kHz, lo que se logra imponiendo una constante de 250 para el contador y 125 para la comparación.

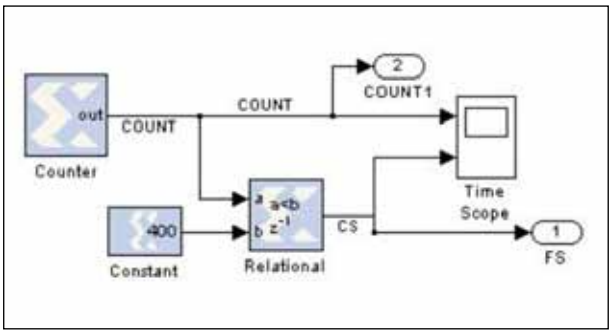


Figura. 2: Diseño del reloj para programar el filtro MAX295 con una frecuencia de corte 4 kHz.

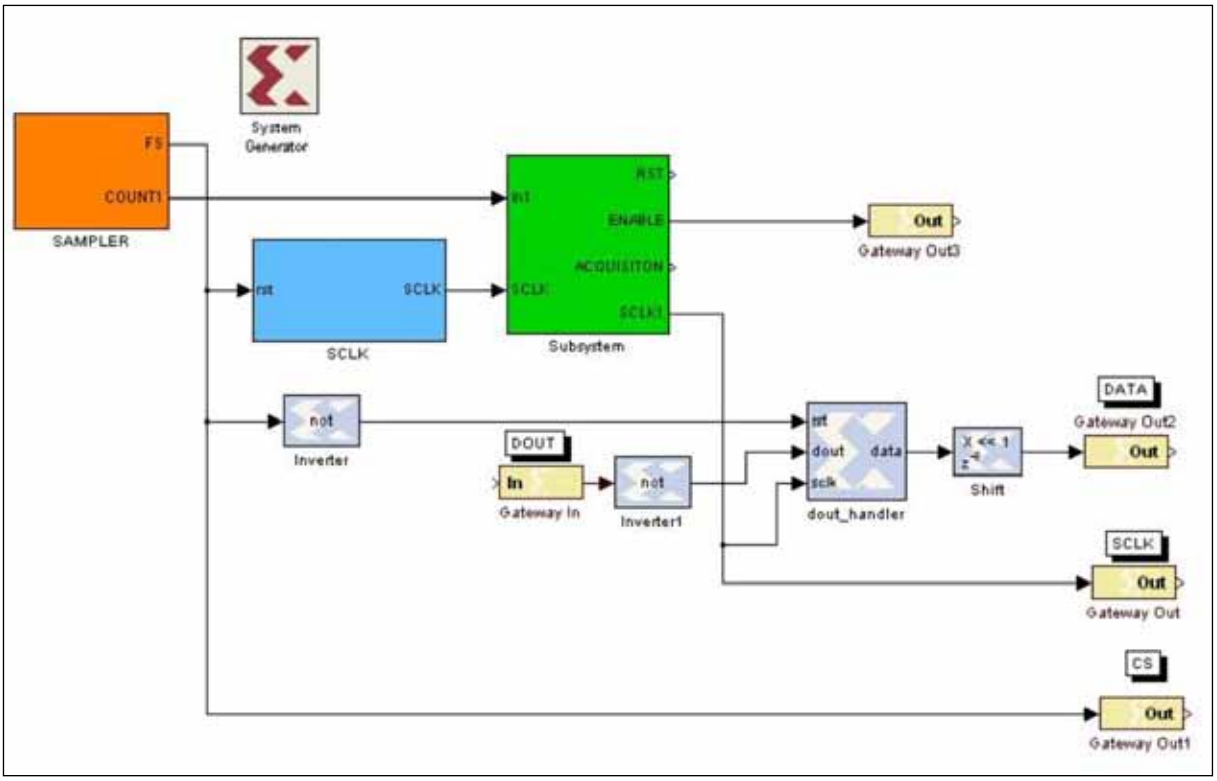
Bloque de interfaz convertidor ADC

El convertidor ADC MAX187 tiene un formato serie paralelo SPI con tres terminales de control -SCLK, CS y DOUT- por lo que en circuito FPGA tiene que implementarse una interfaz para generar y procesar dichas señales, incluyendo la conversión de 12 bits serie a paralelo.

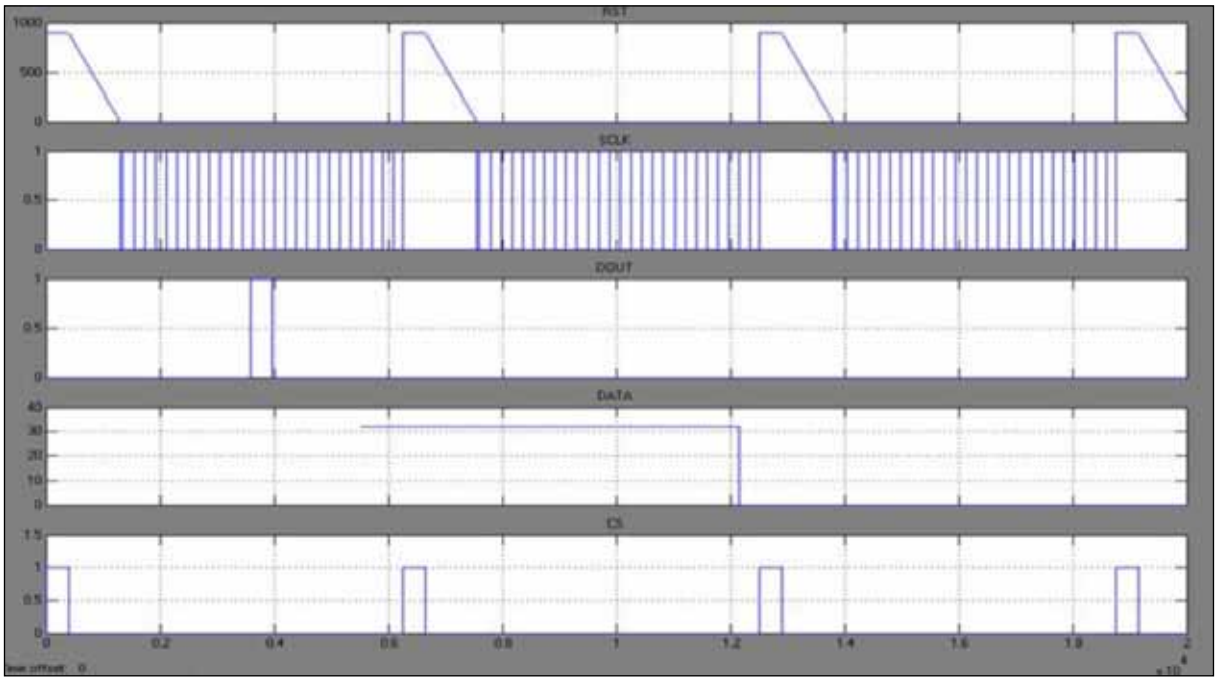
El bloque de interfaz para el convertidor MAX187 debe generar las señales SCLK y CS, así como procesar la respuesta serie DOUT del mismo, cumpliendo los requerimientos de la carta de tiempo del convertidor. El inicio de la conversión se determina por el flanco de caída de CS y la comunicación perdura siempre que se mantenga en un nivel bajo. Por lo tanto esta señal se establece con un procedimiento similar al descrito en el bloque anterior, pero con un ciclo útil del 5% y un período total de 125 µs que define el muestreo de 4 kHz.

Cuando se inicia la conversión, el terminal DOUT responde cambiando a un nivel bajo y pasa a nivel alto después de transcurrido el tiempo de conversión de 8.5 µs y el dato se almacena en un registro interno del convertidor. Para obtener el código almacenado en dicho registro es necesario generar entonces la señal de reloj SCLK de forma tal que con cada flanco de caída de la misma se puede leer, uno a uno, los bits del código del convertidor en DOUT comenzando por el bit más significativo.

La generación de la señal SCLK se ha implementado también con el esquema de la figura 2, pero con el contador habilitado con la señal CS en nivel bajo y seleccionando la constante del contador como 380, resultando en una señal de 7.6 µs con ciclo útil del 50%. Se ha adicionado una lógica a fin de que este reloj se active después del fin de conversión según otro contador que introduce una demora durante 10 µs. Para procesar la señal serie DOUT se realizó un programa VHDL que en esencia realiza la conversión serie a paralelo. Se empleó un bloque del tipo "black box" de la biblioteca XSG que permite incluir este



a)



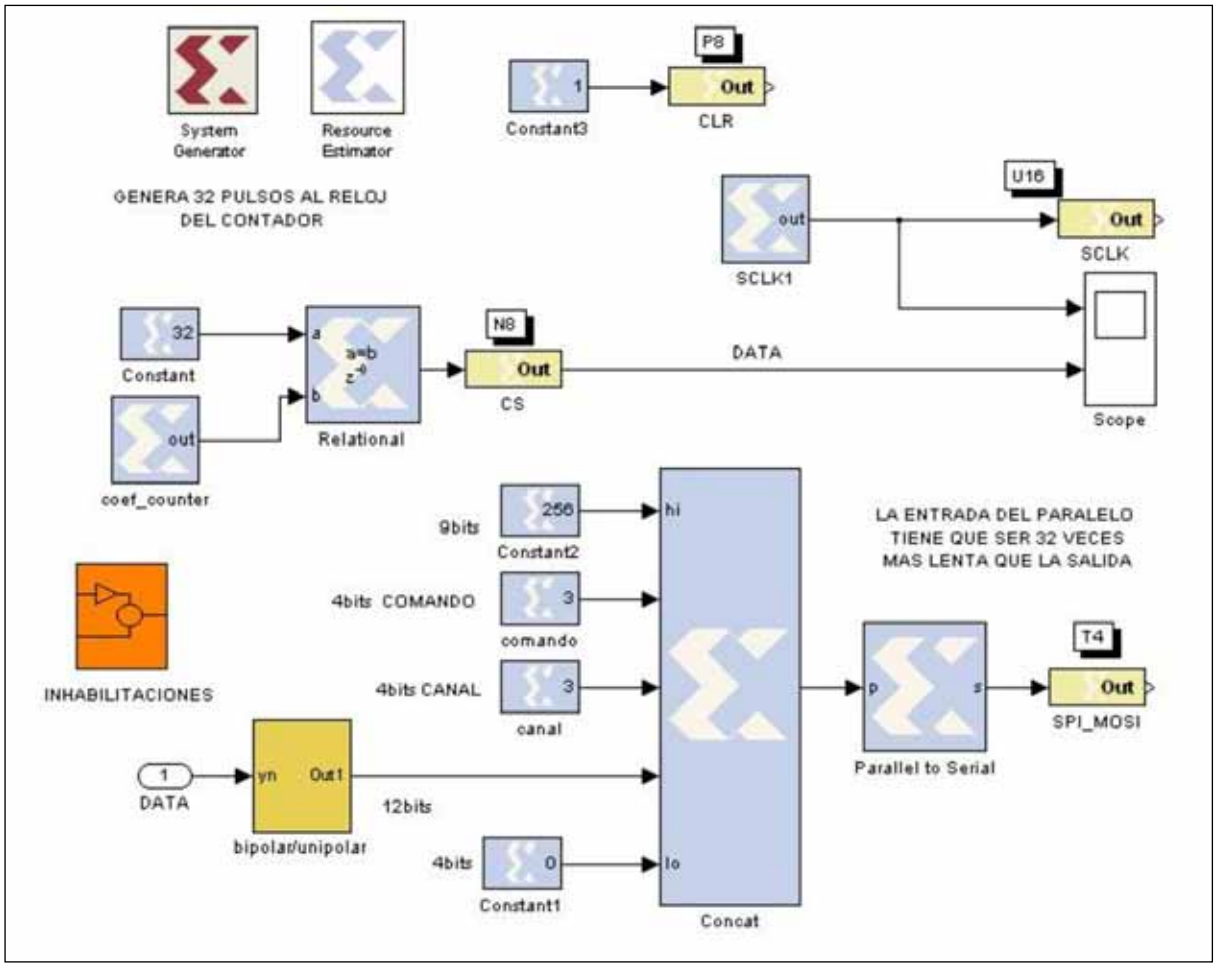
b)

Figura 3: Detalles de la implementación del a) Modelo en Simulink y b) la simulación de la interfaz para manejar el convertidor MAX187.

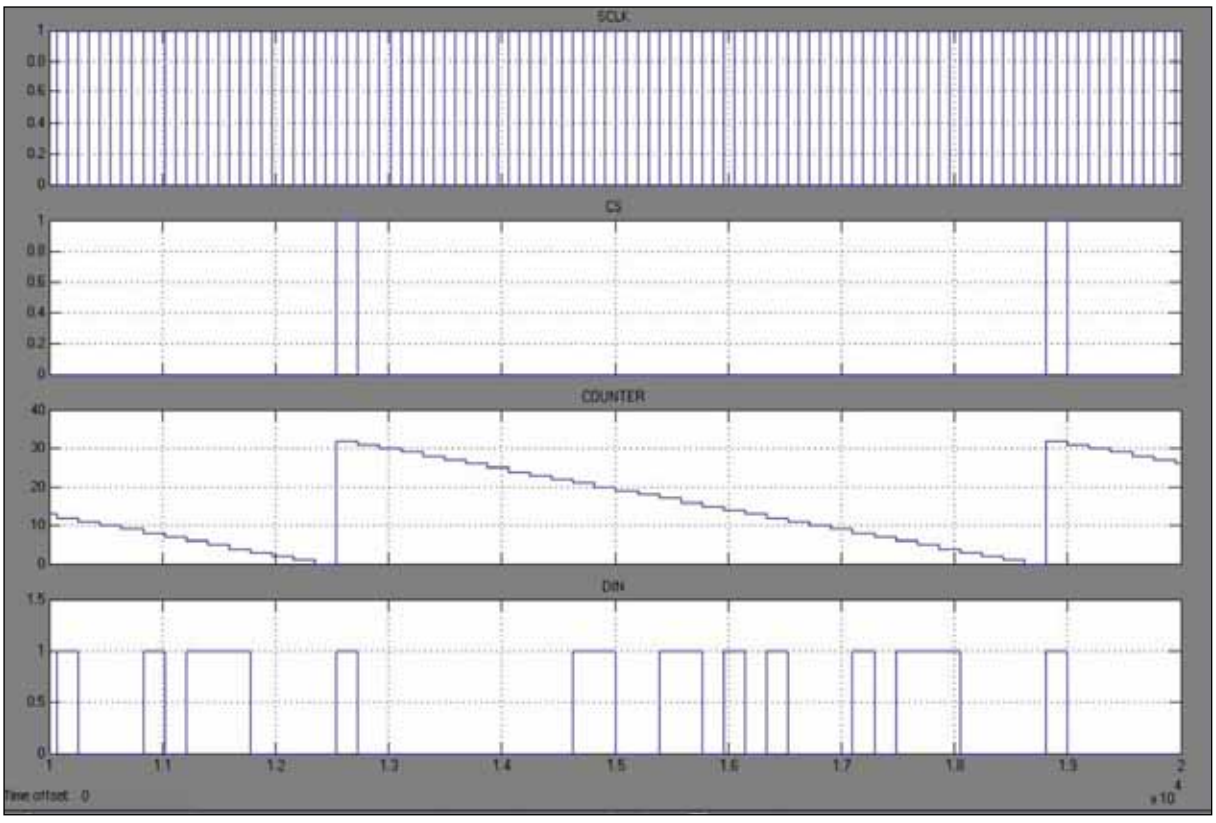
código dentro del ambiente de Simulink, Los resultados de la simulación y el modelo en Simulink de la interfaz analizada se presentan en la figura 3. En la simulación se ha incluido una señal generada en Simulink que muestra la respuesta DOUT del convertidor, la conversión de serie a paralelo se muestra en la señal DATA que es activa en el próximo ciclo de CS. El modelo presenta los subsistemas para generar cada una de las señales necesarias y las conexiones asignadas con los terminales de entrada y salida del FPGA.

Bloque de interfaz para convertidor DAC

Para manejar el convertidor DAC LTC2624 se requiere un procedimiento análogo al descrito en el convertidor ADC pues también posee un formato serie paralelo, es preciso generar las señales de control del mismo mediante la implementación de su carta de tiempo. Para generar un voltaje determinado es necesario enviar el código en 12 bits del mismo, mediante una palabra de control de 32 bits que incluye además del código a los bits adicionales para



a)



b)

Figura 4: Modelo para a) la interfaz del convertidor DAC y b) resultados de la simulación del mismo.

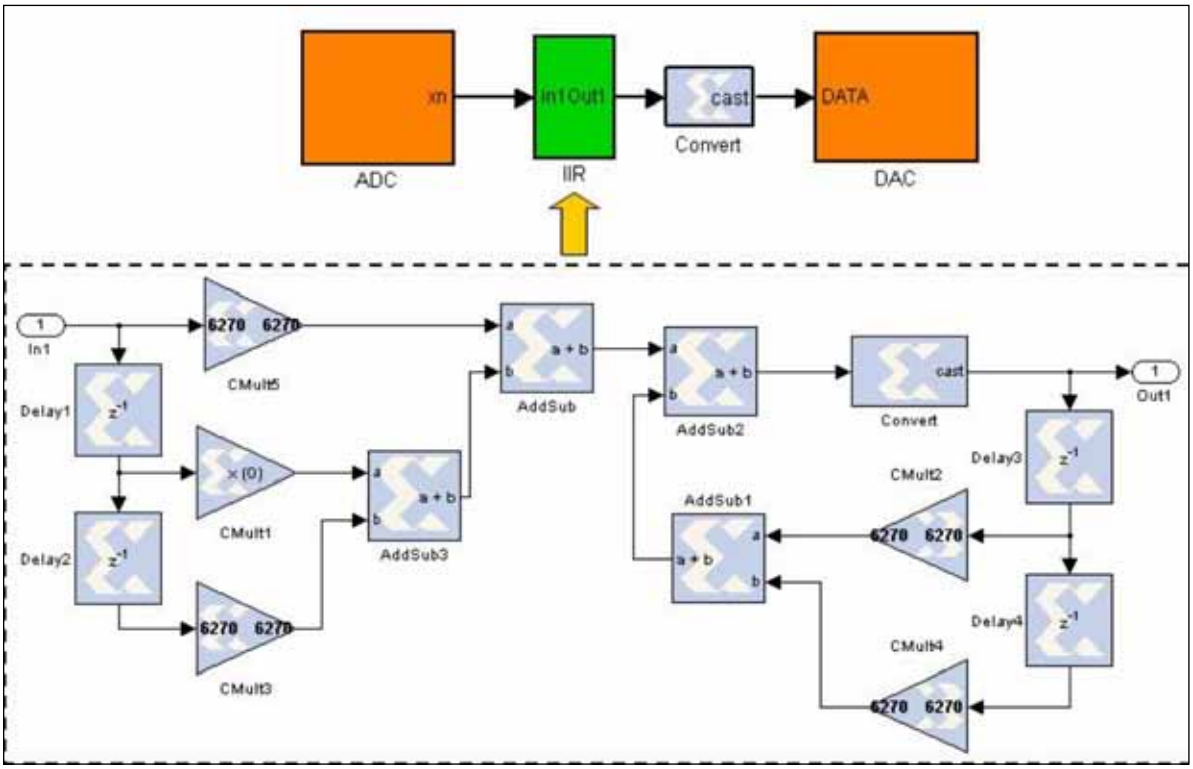


Figura 5. Implementación de un filtro IIR resonador de orden 2

la selección de uno de los cuatro canales disponibles y un comando de acción entre otros. Primeramente se concatenan todos los bits que conforman los 32 bits de la palabra de control y se convierten a formato serie. Para activar el funcionamiento del convertidor se baja CS y se introducen cada uno de los 32 bits de la palabra de control mediante un convertidor de paralelo a serie que la conecta con el terminal SPI_MOSI de dicho convertidor. La generación de CS se hace mediante un contador que cuenta 33 pulsos de un reloj de periodo 3.8 µs que resulta en un periodo de muestreo total de 125 µs. El sistema incluye un bloque para convertir el dato del código de voltaje de formato bipolar a unipolar. En la figura 4 se muestra el modelo del sistema descrito y los resultados de la simulación para generar las señales correspondientes.

Bloques para aplicaciones de procesamiento digital de señales

Después de desarrollada la aplicación básica, se pueden generar otros tipos de aplicaciones realizando el bloque correspondiente de procesamiento y conectando la entrada del bloque en cuestión con la salida de la interfaz del ADC y la salida del bloque hacia la entrada de la interfaz DAC. La estrategia para desarrollar cualquier aplicación digital se basa en implementar la ecuación en diferencias correspondiente a la misma, que no es más que una combinación de sumadores, multiplicadores y registros para almacenar los diversos retardos que se requieran.

Por ejemplo para implementar un filtro pico, resonador de orden 2, se implementa su ecuación en diferencias con los coeficientes correspondientes calculados de acuerdo a las características del filtro. Podemos realizar el diseño con la herramienta de diseño de filtros digitales fdatool [12] de Matlab, para una frecuencia central de 770 Hz con un ancho de banda de 10 Hz y una frecuencia de muestreo se obtiene la ecuación en diferencias siguiente:

$$y_n + 1.424\, y_{n-1} + 0.6682\, y_{n-2} = x_n - 0.1659\, x_{n-2}$$

El modelo de dicha ecuación en forma directa 1 se expone en la figura 5. Para su implementación se necesita efectuar la suma de 4 productos. Se han incorporado 4 bloques para implementar los retardos de las entradas y salidas, así como bloques de multiplicación por una constante para implementar las multiplicaciones por los coeficientes respectivos que emplea aritmética distribuida y evita el uso de multiplicadores dedicados así como los bloques para sumar los factores de los productos. Por tratarse de un filtro IIR, hay que tener en cuenta que la salidas retardadas se multiplican constantemente por coeficientes lo que resulta en el incremento constante del número de bits de las mismas por ello en cada iteración se implementa un bloque convertidor de formato que vuelve a reducir el número de bits antes de comenzar la siguiente iteración.

Discusión de los resultados

Después de sintetizados los diseños y comprobado su funcionamiento por simulación se procedió programar el circuito FPGA y realizar las pruebas experimentales correspondientes. Primeramente se realizó la verificación de la aplicación básica descrita en la sección anterior formada por los tres bloques de manejo del convertidor ADC, del convertidor DAC y del filtro programable. Para asegurar la identidad entre las señales de entrada y salidas se introdujo una constante de corrección que fue ajustada experimentalmente teniendo en cuenta las diferencias en las ecuaciones de los convertidores MAX187 y LTC2624 debido a que tienen voltajes de referencia distintos de 4.096 V y 2.5 V respectivamente. Se realizó una calibración estática primeramente – ver figura 6- imponiendo un voltaje de entrada en el convertidor ADC y midiendo el voltaje de salida en el convertidor DAC, teniendo en cuenta que para voltajes de entrada mayores que la mitad de la referencia del ADC (2.048 V) se corresponde con un voltaje positivo y menores con un voltaje negativo. Por lo tanto un voltaje de entrada de 2.548 V equivale a un voltaje de +0.5 V bipolar lo cual resulta en un voltaje de 1.75 V a la salida del convertidor DAC, que equivale a +0.5 V respecto a su referencia de 1.25 V. Empleando la herramienta polytool de Matlab se obtuvo una línea de regresión con una pendiente de 1.0099 e intercepto de -0.0036 con un coeficiente de regresión mejor de 0.98. Estos valores demuestran una excelente concordancia en la linealidad del sistema.

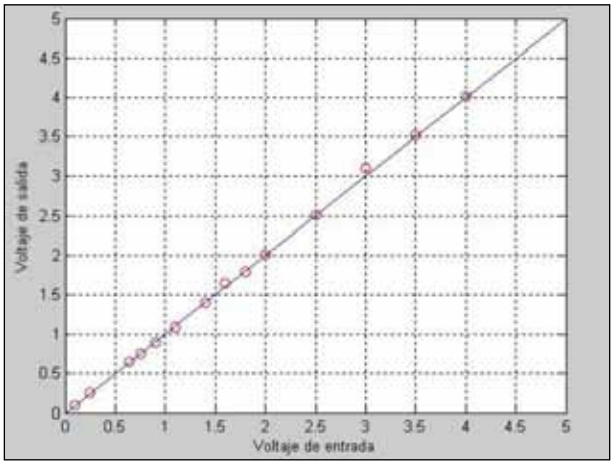


Figura. 6: Calibración estática de entrada y salida

Otro aspecto es la evaluación del trabajo del filtro interpolador de la tarjeta auxiliar que se programa como filtro paso bajo con una frecuencia de corte de 4 kHz y de octavo orden. En la figura 7 se muestra los resultados de las mediciones en un osciloscopio de la aplicación básica con ganancia unitaria. Se aplica a la entrada del procesador una señal compuesta por dos tonos con frecuencias de 697 y 770 Hz, la señal se digitaliza y se pasa directamente al convertidor DAC que se conecta a la entrada del filtro interpolador . En el canal izquierdo del osciloscopio se muestra la señal a la entrada del filtro y en el derecho se muestra la señal a su salida . Se observa como la señal de entrada se tienen los valores discretos de las diferentes muestras de voltaje generadas por el convertidor DAC mientras que a la salida se ha conformado una señal analógica continua resultado de la acción eficaz del filtro interpolador.

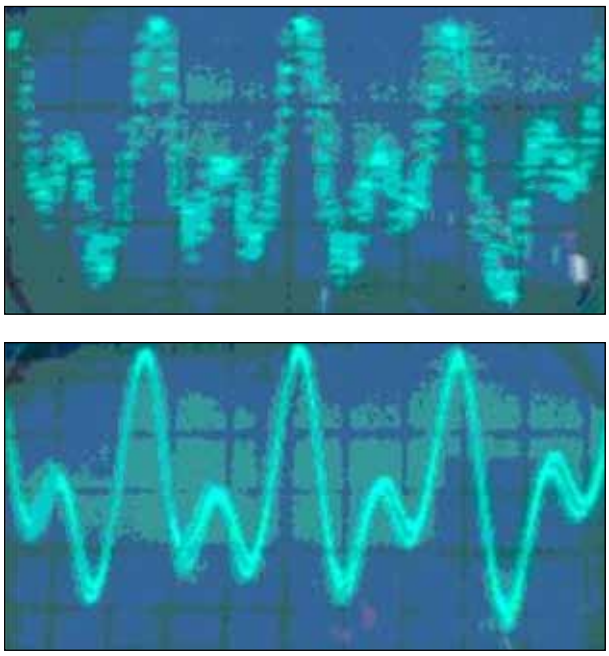


Figura. 7. Mediciones en un osciloscopio a la salida del convertidor DAC y a la salida del filtro interpolador correspondiente.

Finalmente se aprecia en la figura 8, el funcionamiento completo de la aplicación básica donde se muestra como la señal a digitalizar (canal A) y la salida generada por filtro interpolador(canal B) se corresponden adecuadamente lo que demuestra el funcionamiento dinámico de la ganancia unidad con la que se ha implementado el sistema. Las escalas de los voltajes en cada canal son por supuesto iguales.

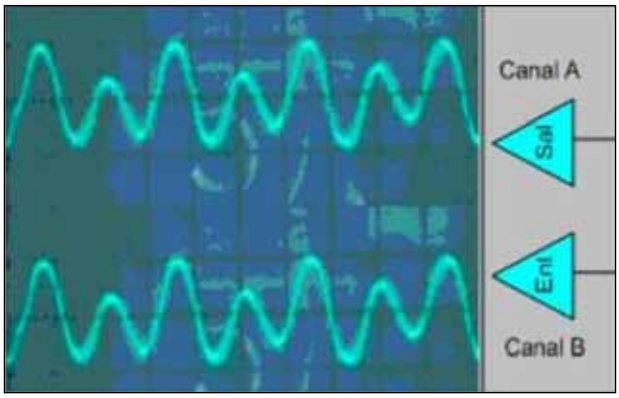


Figura 8. Medición en el osciloscopio de la aplicación básica con ganancia unitaria.

Para verificar una aplicación de procesamiento digital de señales tal y como la descrita en el filtro pico de la sección anterior, se muestra en la figura 9, el módulo de la respuesta de frecuencia medida de dicho filtro en

comparación con la respuesta frecuencia calculada. La respuesta de frecuencia medida se realizó midiendo la razón entre el voltaje de entrada y salida efectivos en el filtro, empleando un voltímetro de CA digital de verdadero valor con error menor del 0.1% La respuesta de frecuencia calculada se obtuvo mediante la herramienta de diseño de filtros digitales de Matlab fdatool con las características ya antes mencionadas. La correspondencia entre ambas respuestas la calculada y la medida es mejor que el 98%.

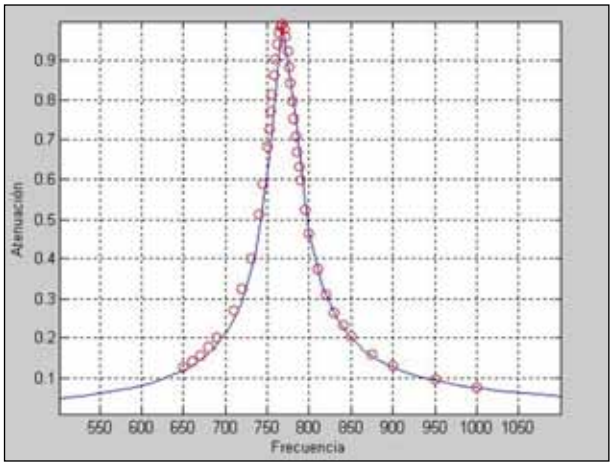


Figura 9. Respuesta de frecuencia calculada y teórica del filtro resonador.

En la figura 10 se aprecia un detalle del experimento anterior pero ahora comparando señales sinusoidales a la entrada y a la salida del filtro resonador medidas en un osciloscopio. Como el filtro está sintonizado a la frecuencia de 770 Hz, con una frecuencia de la señal de entrada de 850 Hz (canal A) se obtiene a la salida del filtro (canal B) una señal de la misma frecuencia pero atenuada a 0.2 que se corresponde con la característica de la figura 9.

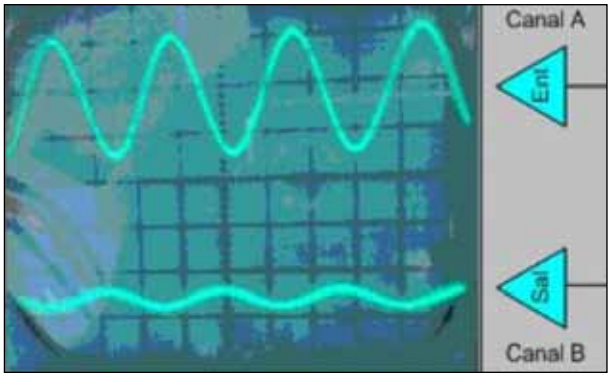


Figura 10. Medición en un osciloscopio de la respuesta del filtro pico con una entrada fuera de la frecuencia de sintonía.

Finalmente se diseñó una aplicación que permite procesar una señal de audio en tiempo real. Para ello se alimentó la entrada del sistema con la salida de la tarjeta de audio de una PC en la cual se ejecuta un fichero de audio. En el procesador digital se instrumentaron varios filtros paso bajo y paso alto con distintas frecuencias de corte que recibían la señal de audio digitalizada por el sistema. Los filtros fueron diseñados con el bloque para filtros FIR del tipo MAC – multiplicador y acumulador- el cual permite implementar filtros de gran orden con un sólo multiplicador. En nuestro caso empleamos filtros de orden 66, lo cual implica que el filtro realmente opera a una frecuencia de muestreo 66 veces superior a la frecuencia establecida. Las salidas de los distintos filtros alimentan un multiplexor de 4 canales que se controla con dos interruptores disponibles en la tarjeta Spartan3E. La salida del multiplexor alimenta el bloque interfaz del convertidor DAC pasa a la tarjeta auxiliar donde se genera la señal de salida analógica recuperada y se alimenta en un amplificador de audio. Conmutando convenientemente los distintos filtros con los interruptores se puede apreciar el efecto de los mismos al atenuar o enfatizar distintas bandas del espectro de audio. El modelo del sistema completo se muestra en la figura 11. Se omiten los detalles del diseño de los filtros, pero se ha seguido un procedimiento similar al filtro resonador anteriormente explicado, es decir se ha implementado su ecuación en diferencias.

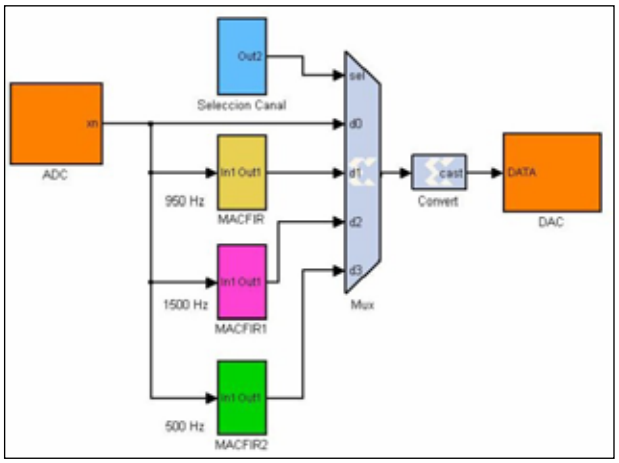


Figura 11. Modelo de la aplicación final.

Conclusiones

Se ha sintetizado una aplicación genérica de un procesador digital en tiempo real para señales de audio a partir de la herramienta XSG instalada en el programa Matlab-Simulink para un circuito FPGA Spartan3E. El circuito inicialmente ha sido concebido para una frecuencia de muestreo de 8 kHz pero puede modificarse fácilmente hasta frecuencias superiores de 40 Hz puesto que los convertidores ADC y DAC lo permiten y obtener así una calidad de audio del tipo disco compacto. El uso de filtros programables MAX295 permite que los cambios en la frecuencia de muestreo que se hagan puedan extenderse a los filtros anti-réplicas e interpolador con sólo reajustar la generación de la señal de reloj de control para los mismos

En la aplicación desarrollada se han implementado y caracterizado diversos filtros digitales como un filtro IIR del tipo resonador y filtros FIR paso bajo y paso alto, pero como los filtros se generan a partir de su ecuación en diferencias mediante bloques en Simulink se puede cambiar las

características de los mismos según se desee con relativa facilidad, obteniendo el diseño del filtro deseado en el propio ambiente de Matlab con la herramienta de diseño de filtros fdatool. Se puede por lo tanto emplear la aplicación propuesta como una plantilla genérica formada con los bloques de control de los convertidores ADC y DAC y de los filtros interpolador y anti-réplica, cambiando solamente por lo tanto el bloque de filtros digitales según se necesite

Se ha obtenido por otra parte una excelente concordancia entre lo simulado en Matlab y en la práctica. Esta herramienta puede ser usada tanto para fines docentes en enseñanza de pre o postgrado así como en aplicaciones de investigación que lo requieran. La biblioteca XSG dispone de otros muchos recursos disponibles que no han sido comentados en este trabajo que pueden ser también instrumentados. Es realmente notable el ahorro de tiempo para el desarrollo de una aplicación que se ha obtenido con la herramienta XSG, respecto al flujo de diseño tradicional de circuitos FPGA basado en el uso directo de lenguaje de descripción de hardware.

Aunque se ha empleado una tarjeta Spartan3E Starter Kit con una tarjeta auxiliar para lograr el procesamiento en tiempo real y el uso de filtros interpoladores y anti-réplicas, el sistema puede extenderse a otro tipo de tarjetas o circuitos FPGA que se dispongan haciendo las adaptaciones necesarias.

Referencias

1. **Bamdad A. y Amit K. DSP or FPGA.** *How to choose the right device*. Disponible en http://www.dspdesignline.com (verificado en Marzo, 2008)
2. **Smith W. S.** *The Scientist and Engineer's Guide to Digital Signal Processing*. p. 503. Disponible en http://www.dspguide.com (verificado en Abril, 2008)
3. **Cabrera A.** *Sistemas empotrados y módulos IP*. Curso de Sistemas Digitales Empotrados. Dpto. de Automática y Computación. ISPJAE, Habana. 2008.
4. **Cabrera A.** *Sistema de procesamiento MicroBlaze*. Curso de Sistemas Digitales Empotrados. Dpto. de Automática y computación. ISPJAE, Habana. 2008.
5. **Herrera R.** *Síntesis y evaluación de un procesador DSP empotrado en una FPGA*. Trabajo de Diploma. Universidad de Pinar del Río. Junio 2008.
6. **Xilinx Inc.** *Spartan 3E User Guide*. Disponible en http:// www.xilinx.com (verificado en Marzo, 2006)
7. **Linear Technology.** *Quad 12 bits rail to rail DAC.* Disponible en http://www.linear.com (verificado en Mayo, 2008)
8. **Oppenheim A.V., Schafter R. W., Buck J.R.,** *"Discrete time signal processing"*, Chapter 4, Prentice Hall 2th Edition.1999.
9. **Maxim.** *MAX187 12 bit serial ADC*. Disponible en http:// www.maxim-ic.com (verificado en Abril, 2008)
10. **Maxim.** *Max295 8th order Switched Capacitor Filter.* Disponible en http:// www.maxim-ic.com (verificado en Febrero, 2008)
11. **Xilinx Inc.** *System Generator for DSP*. User Guide Version 8.1 Disponible en http://www.xilinx.com (verificado en Mayo, 2008)
12. **Matworks.** *Signal Processing Toolbox.* FDATOOL: Filter Design & Analysis Tool. Disponible en http://www. Mathworks.com (verificado en Mayo 2008)

Recibido: 02/02/09
Aprobado: 24/10/11

- C. Juan Raúl Rodríguez Suárez[1]
  Profesor Auxiliar del Departamento de Telecomunicaciones y Electrónica de la Universidad de Pinar del Rio. Cuba. Miembro del Grupo de Investigación para el Diagnóstico Avanzado de Maquinarias GIDAM. Trabaja en las temáticas de procesamiento digital de señales, sensores e instrumentación.

- Ismel Domínguez Rodríguez[1]
  Ingeniero en Telecomunicaciones y Electrónica. Miembro del Grupo de Investigación para el Diagnóstico Avanzado de Maquinarias GIDAM.

1. Universidad de Pinar del Río Hermanos Saíz Montes de Oca, Martí 270, P. del Río, Cuba.
(jotar@tele.upr.edu.cu, ismel_t04@telemail.upr.edu.cu.)