DISEÑO DE UN ALGORITMO EN HARDWARE PARA LA SINCRONIZACIÓN DE PORTADORA

Ferney Amaya-Fernández, Jaime Velasco-Medina*

Grupo de Bionanoelectrónica, Escuela EIEE, Universidad del Valle, Cali, Colombia
*Grupo de Automática y Robótica, GAR, Universidad Javeriana, Cali, Colombia
E-mail: foamaya@puj.edu.co, jvelasco@univalle.edu.co

ABSTRACT

Este artículo presenta el diseño de un nuevo algoritmo implementado en hardware para la sincronización de portadora en Radio Software (RS). La sincronización de portadora y de bits es uno de los mayores desafíos en un sistema de comunicación. Con el propósito de verificar el funcionamiento del algoritmo, se realizaron simulaciones en Matlab, las cuales se compararon con los resultados de simulación de un DPLL (Digital Phase-Looked Loop). En este caso, el algoritmo FACS (*Fast Adquisition Carrier Synchronization*) presenta a nivel de simulación un tiempo de adquisición de portadora menor y permite sincronizar la frecuencia de portadora en un rango de frecuencias mayor que el DPLL. El diseño en hardware del algoritmo FACS en un FPGA se realizó usando Xilinx System Generator y usa un total de 13 bloques lógicos configurables CLBs. Con el propósito de verificar el funcionamiento del algoritmo en hardware, se diseñó un sincronizador de portadora para BPSK (Binary Phase Shift Keying). Entonces teniendo en cuenta los resultados obtenidos, este nuevo algoritmo implementado en hardware es adecuado para sincronizar la portadora.

1. INTRODUCCIÒN

La técnica Radio Software (RS) propone el diseño de sistemas inalámbricos configurados y controlados desde software sin cambiar el hardware [1]. El objetivo principal de RS es desarrollar radios completamente digitales, capaces de operar en un rango de frecuencias amplio (de 2MHz a 2.5GHz), con posibilidad de configurar, controlar y actualizar desde software las bandas de radio frecuencia y los modos de acceso al canal [1]. Para llevar a cabo lo anterior considerando la técnica RS, se emplean sofisticadas tareas de procesamiento de señales tales como: modulación, sincronización de portadora, desplazamiento en frecuencia (Digital Down Converter, DDC y Digital Up Converter, DUC) y detección de bits.

Teniendo en cuenta que los requerimientos de las aplicaciones actuales de procesamiento digital de señales en RS son tan exigentes en desempeño, la tecnología de FPGAs se ha convertido en una alternativa de implementación debido a su flexibilidad y al alto nivel de paralelismo que puede alcanzarse.

Una de las tareas más exigentes en un sistema de comunicación, la cual requiere de muchos recursos de hardware es la sincronización de portadora y de bits [2].

Teniendo en cuenta las consideraciones anteriores, este artículo presenta el diseño de un nuevo algoritmo en hardware para la sincronización de portadora, el cual tiene un bajo tiempo de adquisición de portadora y permite sincronizar la frecuencia de portadora en un rango de frecuencias mayor con respecto a un DPLL. Con el propósito de verificar el diseño funcional del algoritmo, se realizaron simulaciones en Matlab, las cuales se compararon con los resultados de simulación de un DPLL. En este caso, en las simulaciónes se emplea como entrada una señal BPSK, la cual es diferente en fase y frecuencia con respecto a la señal del oscilador local del receptor.

El diseño en hardware se realizó usando la herramienta Xilinx System Generator y fue sintetizado en el FPGA Spartan-3 XC3S200 usando 13 bloques lógicos configurables CLBs. Con el propósito de verificar el correcto funcionamiento del algoritmo implementado en hardware, se diseñó un sincronizador de portadora para BPSK.

Este trabajo está organizado de la siguiente forma. Inicialmente la sección 2 describe los trabajos previos, la sección 3 presenta algunos aspectos básicos sobre la sincronización de portadora empleando el criterio de máxima probabilidad. La sección 4 presenta dos técnicas para diseñar el sincronizador de portadora, la primera basada en un DPLL y la segunda basada en el algoritmo propuesto, también esta sección presenta las simulaciones en Matlab para los diseños. La sección 5 presenta el diseño del algoritmo FACS en un FPGA usando Xilinx

System Generador. Finalmente, la sección 6 presenta las conclusiones y el trabajo futuro.

2. TRABAJOS PREVIOS

Desde la aparición de Radio Software se han propuesto varias alternativas para la implementación de radios digitales empleando circuitos programables como Microprocesadores de Propósito General (General Purpose Processors, GPPs), Procesadores Digitales de Señales (Digital Signal Processors, DSPs), FPGAs (Field Programmable Gate Arrays) y ASICs (Application Specific Integrated Circuits).

Los fabricantes de FPGAs están orientando esfuerzos hacia la implementación de RS en FPGAs. Este es el caso de Altera y Xilinx, los dos más grandes vendedores de FPGAs en el mundo, están proponiendo soluciones para RS. En la Figura 1 se muestra el diagrama de una arquitectura para RS propuesta por Altera [3]; en la cual se implementan el módulo de procesamiento en banda base y el módulo de procesamiento en frecuencia intermedia. En esta arquitectura se presentan los bloques funcionales para el procesamiento en frecuencia intermedia: DUP (Digital Up-Converter), CFR (Crest Factor Reduction), DPD (Digital PreDistortion) y DDC (Digital Down-Converter); y los bloques para el procesamiento en radio frecuencia: PA (Power Amplifier), VCO (Voltaje Controller Oscilator) y LNA (Low Noise Amplifier).

En la Figura 2 Xilinx presenta una arquitectura para el procesamiento en banda base, la cual puede ser implementada totalmente en un FPGA [4].

Adicionalmente, Altera y Xilinx han desarrollado módulos IPs para el procesamiento de señales en comunicaciones. Entre los módulos de propiedad intelectual de Altera y Xilinx se encuentran: transformada rápida de Fourier (FFT: Fast Fourier Transform), codificadores para la corrección de errores (Reed Solomon, Viterbi, Turbo Códigos), convertidores de frecuencia digitales (up/down converter) y moduladores digitales.

Chris Dick, Fred Harris y Michael Rice en [2], [5], [6] presentan el diseño del algoritmo CORDIC (COrdinate Rotation DIgital Computer), el cual es empleado en la sincronización de portadora para QAM (Quadrature Amplitude Modulation) y PSK (Phase Shift Keying). En [5] se presentan dos diseños para sincronizar la portadora en SR para QAM usando un FPGA de Xilinx. Un diseño usa LUTs y el otro el algoritmo CORDIC. En este caso, el algoritmo CORDIC presenta mejor desempeño, obteniéndose un menor tiempo de adquisición de la portadora.

En [2] se emplea el criterio de máxima probabilidad (maximum-likelihood, ML) para sincronizar la portadora para QPSK (Qadrature Phase Shift Keying) y QAM. En este caso, se usa un PLL (Phase-Looked Loop) para realizar la sincronización, el cual emplea la tangente inversa. El modelo fue verificado empleando Simulink, VHDL y Xilinx Core Generator. En este artículo se presenta el área ocupada, mas sin embargo, no presenta resultados sobre la velocidad. El PLL QPSK usa aproximadamente 1000 elementos lógicos de una FPGA Virtex de Xilinx

3. SINCRONIZACIÓN DE PORTADORA

En un sistema de comunicación, sin la adecuada sincronización entre la portadora y la señal generada por el oscilador local, no es posible reconocer la información enviada por el transmisor. En la demodulación coherente, el receptor debe sincronizar su oscilador local en frecuencia y fase con la portadora. Para sincronizar la

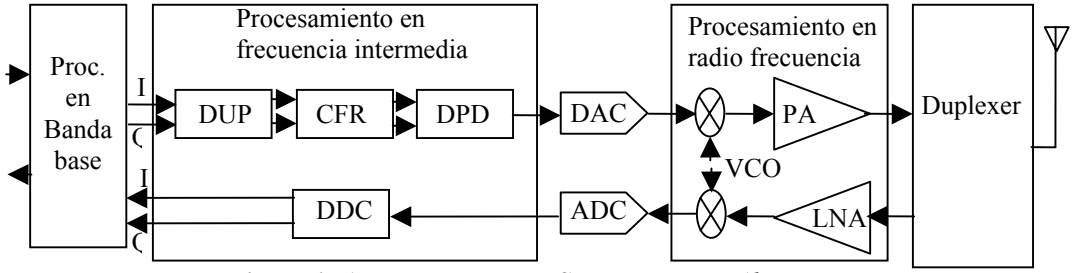

Figura 1: *Arquitectura para RS propuesta por Altera.*

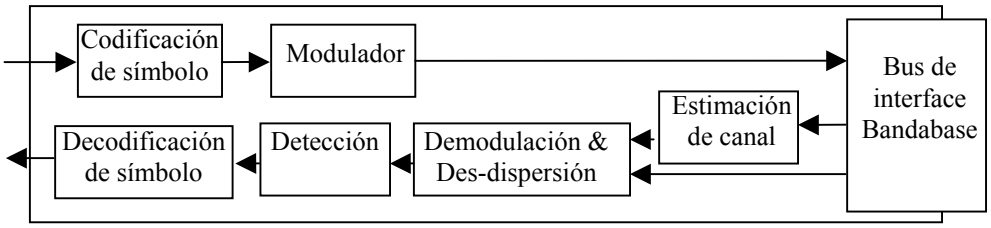

Figura 2: *Esquema de procesamiento en banda base propuesto por Xilinx.*

portadora puede emplearse una portadora piloto o se puede derivar la fase de la portadora directamente de la señal modulada [7]. El último método es el más empleado en la práctica.

Asumiendo que se transmite una portadora sin modular y considerando un canal AWGN (Additive White Gaussian Noise), la señal recibida por el receptor es dada por la ecuación 1.

$$r(t) = A\cos(2\pi f_c + \phi) + n(t) \qquad (1)$$

donde *A* es la amplitud, f_c es la frecuencia de la portadora, ϕ es la fase desconocida de la portadora y *n(t)* es ruido gausiano.

Uno de los criterios para estimar el valor de la fase desconocida ϕ es el criterio de máxima probabilidad (maximum-likelihood, ML), el cual permite encontrar la mejor estimación de ϕ al maximizar la función de probabilidad $\Lambda(\phi)$. El valor que más se aproxima a la fase desconocida está dado por la ecuación 2 [7].

$$\phi_{ML} = -\tan^{-1}\left\{\frac{\int_{To} r(t)\sin\omega_c t dt}{\int_{To} r(t)\cos\omega_c t dt}\right\} \qquad (2)$$

donde *To* es el tiempo de símbolo.

A partir de la ecuación anterior, se puede generar el diagrama de bloques de la Figura 3, el cual es empleado para estimar la fase de la portadora no modulada [7].

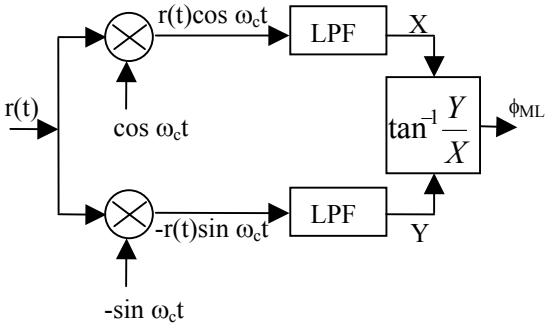

Figura 3: *Diagrama de bloques del detector de fase.*

Una de las características de este detector de fase, es que este puede ser implementado digitalmente, proporcionando la flexibilidad requerida en RS al permitir la detección y sincronización de varias técnicas de demodulación como QAM y PSK. En este caso, las señales seno y coseno son generadas por un NCO (Numerically Controlled Oscillator).

La salida del detector de fase, luego de ser procesada, se emplea para variar la frecuencia y el ángulo del oscilador local y permite la sincronización de portadora. Usualmente se emplea un PLL para procesar la salida del detector de fase y generar el offset del NCO.

En la Figura 4 se presenta un diagrama de bloques para el sincronizador de portadora. En este caso, el diagrama propuesto es una modificación del diagrama de bloques de un DPLL.

El sincronizador está conformado por un detector de fase, un NCO y una unidad que genera el offset, la cual permite variar la frecuencia o fase de la señal producida por el NCO. Si la señal *r(t)* es modulada empleando varias fases, el bloque generador de offset debe producir la diferencia entre ϕ_{ML} y el valor de fase más próximo a los valores establecidos para la comunicación.

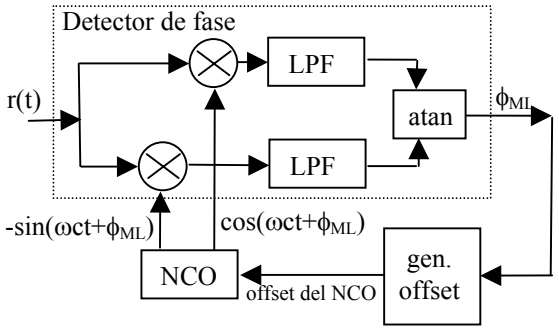

Figura 4: *Diagrama de bloques del sincronizador de portadora.*

4. DISEÑO FUNCIONAL DEL SINCRONIZADOR DE PORTADORA

En esta sección se presentan dos técnicas para diseñar el sincronizador de portadora. La primera técnica es basada en usar un DPLL, en este caso el generador de offset es un filtro. La segunda técnica es basada en usar el nuevo algoritmo propuesto en este trabajo, es decir el FACS (Fast Adquisition Carrier Synchronization), el cual genera el offset del NCO. En este caso, el algoritmo FACS presenta un menor tiempo de adquisición de portadora y permite sincronizar la frecuencia en un rango mayor que el DPLL.

4.1. Diseño del DPLL

En la Figura 5 se presenta el diagrama de bloques del DPLL. Con el propósito de simplificar el diseño del DPLL, no se tuvieron en cuenta los filtros del detector de fase para calcular la función de transferencia del DPLL.

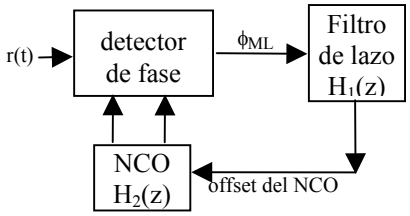

Figura 5: *Diagrama de bloques del DPLL.*

$H_1(z)$ es la función de transferencia del filtro de lazo y $H_2(z)$ es la función de transferencia del NCO, en este caso las ecuaciones del filtro y del NCO están dadas por las ecuaciones 3 y 4.

$$H_1(z) = \frac{\beta}{1-\alpha z^{-1}} \tag{3}$$

$$H_2(z) = \frac{1}{1-z^{-1}} \tag{4}$$

Entonces la función de transferencia de lazo cerrado del DPLL está dada por la ecuación 5.

$$H(z) = \frac{\beta\, z^2}{z^2(1+\beta)+(-\alpha-1)z+\alpha} = \frac{N(z)}{z^2+C_2 z+C_1} \tag{5}$$

Para el diseño del DPLL se deben considerar los parámetros ω_n (frecuencia natural), ξ (factor de amortiguamiento), *ts* (tiempo de establecimiento) y *Ts* (periodo de muestreo) empleando las ecuaciones 6 y 7.

$$C_1 = e^{-2\xi\omega_n Ts} \tag{6}$$

$$C_2 = -2C_1\cos(\omega_n Ts\sqrt{1-\xi^2}) \tag{7}$$

Para la selección del tiempo de establecimiento *ts* debe tenerse en cuenta el retardo del detector de fase, el cual depende en gran parte del número de TAPs del filtro seleccionado. Los parámetros de diseño empleados son:

- ξ=0.707
- Filtro de lazo: FIR de 50 TAPs
- ts = 50 Ts
- Número de muestras por ciclo de la portadora: 4
- Número de muestras por símbolo: 50

4.2. Diseño del algoritmo FACS

En la Figura 6 se presenta el diagrama del sincronizador de portadora para BPSK empleando el algoritmo FACS. El nuevo algoritmo FACS propuesto en este trabajo se presenta en la Figura 7.

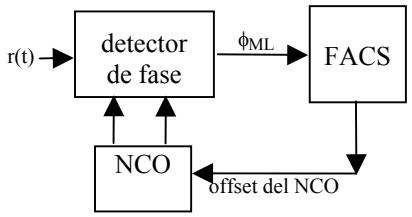

Figura 6: *Diagrama de bloques del sincronizador BPSK empleando el algoritmo FACS.*

```
if offset != 0,
offset = 0;
count = 0;
end
if count != delay,
count = count + 1;
end
if (ang == ang1) && (count == delay),
   if (ang > 20°) && (ang <= 160°),
      offset = ang;
      if offset > 90°,
         offset = 180° - offset;
      end
   end
end
if abs(offset) > 62°,
  d = d*-1;
end
offset = d * offset;
ang1 = ang;
```

Figura 7: *Descripción del algoritmo FACS.*

Las características del algoritmo son las siguientes:

- No permite que pequeñas variaciones de fase debido a ruido activen la función del sincronizador, solo si el valor de la fase estimada ϕ_{ML} se encuentra en el rango entre 20° y 160°, se genera el valor ϕ_{ML} como offset del NCO.
- Tiene en cuenta el retardo del detector de fase, es decir, el offset del NCO es cero hasta que se estabilice la salida del detector de fase, en este caso hasta que haya transcurrido un tiempo desde la última modificación del offset.
- Presenta un mecanismo para determinar cuando ϕ_{ML} es estable, lo cual permite evitar lecturas erróneas cuando ocurre ruido o cuando ocurren cambios de fase. En este caso, se asume que ϕ_{ML} es estable si el valor actual es igual al valor del periodo de muestreo anterior.

- Permite obtener el valor del ángulo más cercano a los valores esperados de 0° y 180°. En este caso, cuando ϕ_{ML} es mayor a 90°, el valor de la fase se calcula como: fase = 180° - ϕ_{ML}
- Permite la sincronización entre la frecuencia de la portadora y la frecuencia del oscilador local considerando las diferencias en frecuencia. En este caso, el valor del offset tendrá signo positivo o negativo para poder aumentar o disminuir la frecuencia del NCO, es decir, el signo del offset cambiará si el valor de la fase calculado en el punto anterior sobrepasa cierto umbral. Debido a que el valor del offset tiene un rango entre 0° y 90°, el umbral se estimó en el 77% de 90° (valor máximo), en este caso el valor del umbral es de 61.6° ≈ 62°.

Un algoritmo simple para la detección de los bits modulados en BPSK consiste en establecer un umbral para la señal entregada por el detector de fase. Si la fase es mayor a 90° se detecta un '1' lógico y si la señal es menor a 90° se detecta un '0' lógico.

4.3. Resultados de simulación

Con el propósito de verificar el funcionamiento de FACS y compararlo con el DPLL, en esta sección se presentan los resultados de las simulaciones en Matlab para la sincronización de portadora considerando: desfase de la portadora, diferencia en frecuencias (entre la portadora y el NCO) y la presencia de ruido gausiano.

La primera simulación tiene como entrada una señal BPSK con los siguientes parámetros:

- Desafase de la portadora: 60°
- Diferencia en frecuencia entre la portadora y el oscilador local: 0Hz.
- Relación señal a ruido: 100dB.
- Bits modulados: secuencia de unos y ceros.

Las señales de salida de los sincronizadores FACS y DPLL se presentan en la Figura 8. Como puede observarse desde la Figura 8, el algoritmo FACS tiene un tiempo de adquisición de portadora *tac* menor que el DPLL, logrando sincronizar la portadora durante el primer símbolo. En este caso, *tac* es de 80 muestras para FACS y de 240 muestras para el DPLL.

La segunda simulación tiene como entrada una señal BPSK con los siguientes parámetros:

- Desafase de la portadora: 60°
- Diferencia en frecuencia entre la portadora y el oscilador local: 12Hz.
- Relación señal a ruido: 7dB.
- Bits modulados: secuencia de unos y ceros.

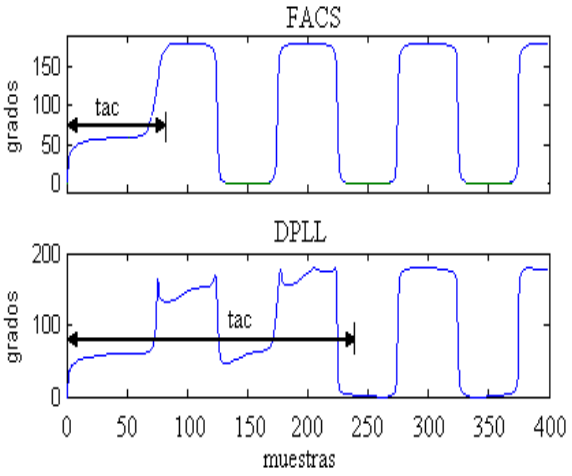

Figura 8: *Señales de salida para entrada BPSK con la portadora desfasada 60°.*

Las señales de salida de los sincronizadores FACS y DPLL se presentan en las Figuras 9 y 10 respectivamente. En la parte inferior de cada gráfica aparecen los bits detectados a partir de la señal entregada por los sincronizadores.

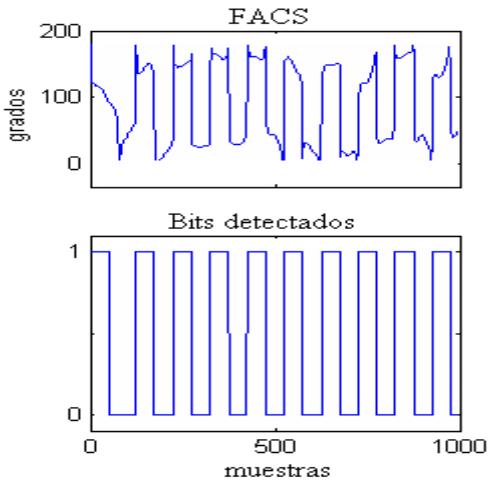

Figura 9: *Señal de salida del sincronizador FACS y bits detectados para una entrada BPSK.*

Como puede observarse desde las Figuras 9 y 10, el algoritmo FACS presenta un mejor desempeño respecto al DPLL. En este caso, permite detectar los bits modulados en BPSK, lo cual no se puede hacer con el DPLL.

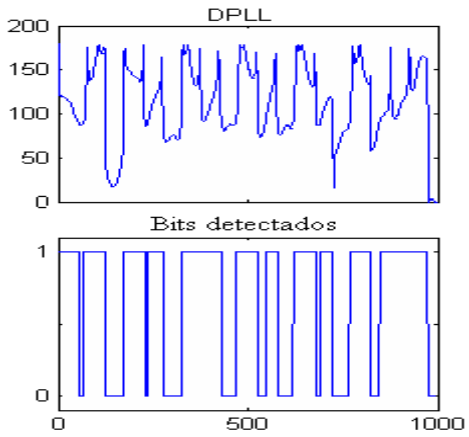

Figura 10: *Señal de salida del sincronizador DPLL y bits detectados para una entrada BPSK.*

5. DISEÑO DEL ALGORITMO FACS

Para diseñar en hardware el sincronizador de portadora usando el algoritmo FACS es necesario diseñar tres bloques funcionales (ver Figura 6): NCO, detector de fase y generador de offset empleando FACS.

El diagrama de bloques del NCO diseñado se presenta en la Figura 11.

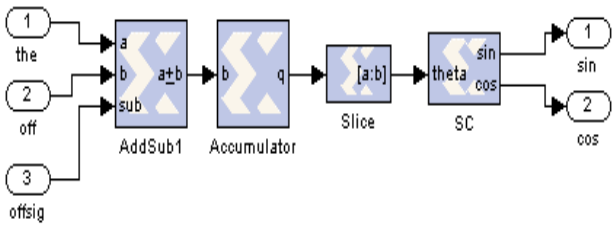

Figura 11: *Diagrama de bloques del NCO.*

La entrada *the*, es una constante que especifica la frecuencia del NCO y la entrada *off*, permite variar la fase y/o frecuencia del NCO. La entrada *offsig* determina la operación suma/resta de las entradas *the* y *off*, la cual es realizada por el bloque *AddSub1*. Tanto la señal *the* y *off* tienen 10 bits para representar la parte entera y 2 bits para la parte fraccional., el bloque *Accumulator* acumula el valor de la salida del sumador. El bloque *SC* es una ROM, la cual contiene los valores de seno y coseno. La función del bloque *Slice* es seleccionar los 10 bits de la parte entera de la salida del acumulador, los cuales se usan para direccionar la ROM. Las salidas *cos* y *sin* son los valores de coseno y seno del ángulo θ de acuerdo a la siguiente ecuación:

$$\theta = (\, the \pm off \,)\, 2\pi/1024 \;\; radianes \qquad (8)$$

Para el diseño del detector de fase se emplearon los siguientes parámetros:

- Muestras por ciclo de portadora: 8
- Muestras por símbolo: 90
- Filtro FIR pasa bajas de 50 TAPs
- Bits modulados: secuencia de unos y ceros.

El diagrama de bloques del detector de fase con el NCO se presenta en la Figura 12. La entrada *mod*, es la señal modulada y las entradas *the*, *off* y *offsig* van directamente al NCO. El cálculo de la tangente inversa se realizó empleando el algoritmo CORDIC. Las salidas del detector de fase son:

- *fase*: valor de la fase de la señal modulada
- *mag*: magnitud escalada de la señal modulada

El diagrama de bloques del algoritmo FACS se presenta en la Figura 13. En este caso FACS tiene como señal de entrada *fase*, que es una señal de salida del detector de fase. Las señales *off* y *offsig* son señales de entrada al NCO.

El algoritmo FACS fue implementado en un FPGA Spartan-3 XC3S200 empleando System Generador. En este caso, el diseño usa:

- Número de CLBs (bloques lógicos configurables): 13
- Número de flip-flops: 18
- Número de LUTs: 80

Con el propósito de verificar el diseño del sincronizador usando el algoritmo FACS se realizaron 2 simulaciones.

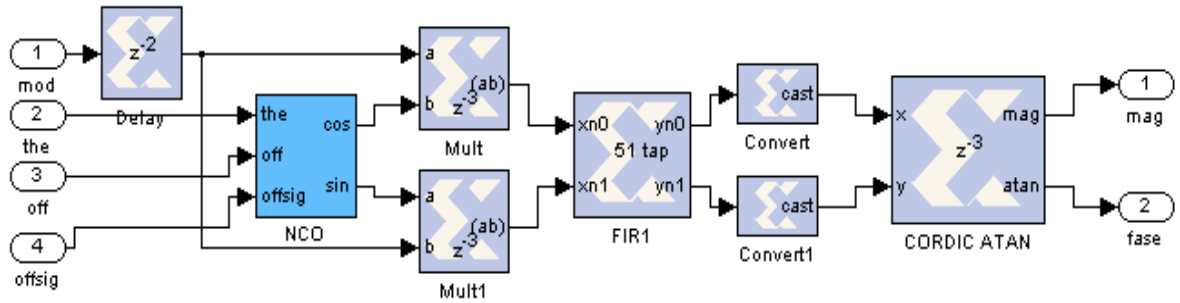

Figura 12: *Diagrama de bloques del detector de fase*

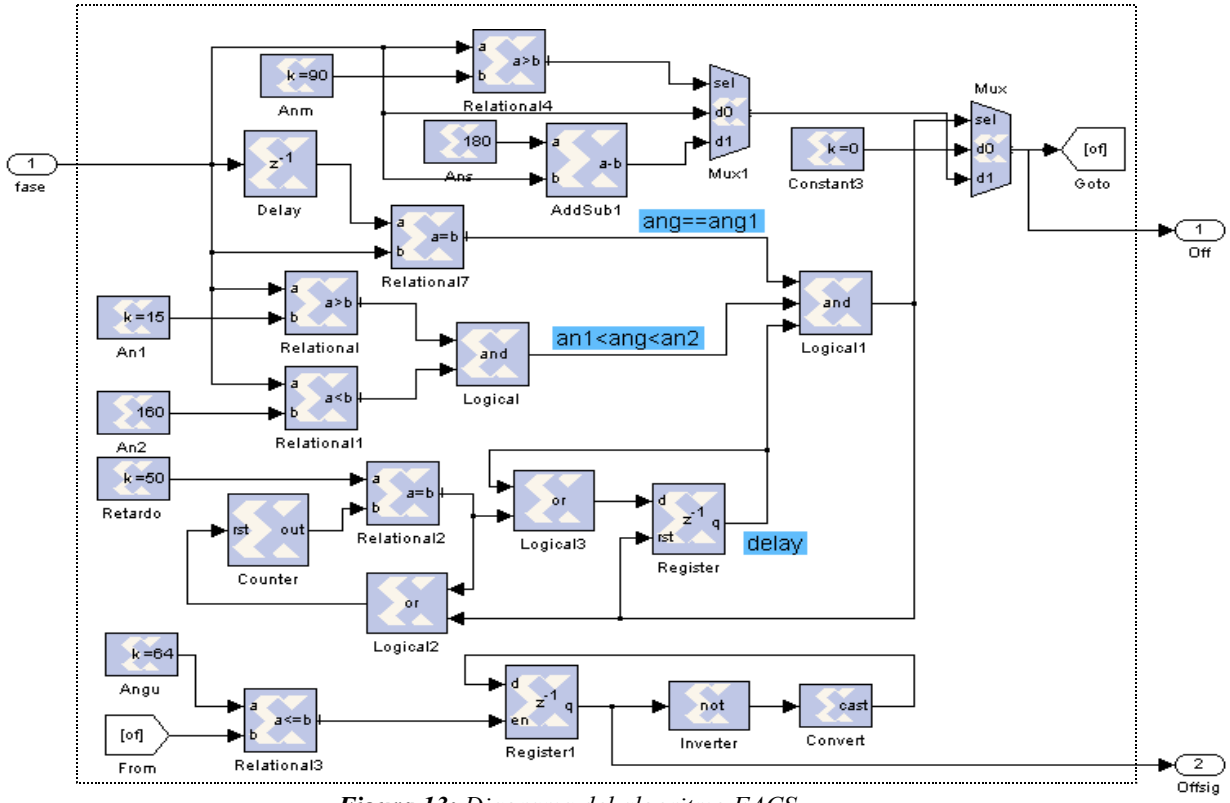

Figura 13: *Diagrama del algoritmo FACS*

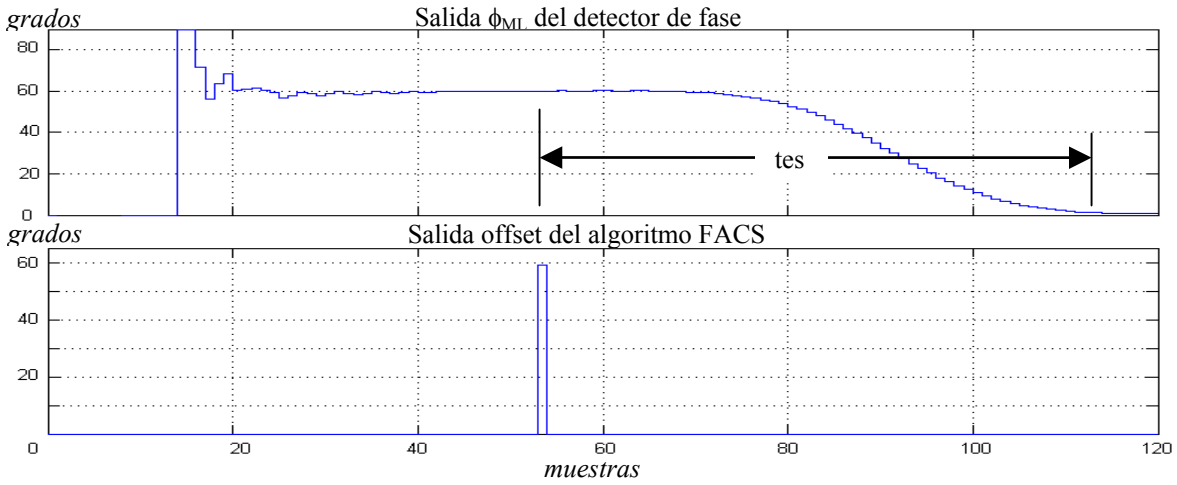

Figura 14: *Resultados de simulación para el detector de fase y FACS*

La primera simulación tiene como entrada una señal BPSK con los siguientes parámetros:

- Desfase de la portadora: 60°
- Diferencia en frecuencia entre la portadora y el oscilador local: 0Hz.
- Bits modulados: secuencia de unos y ceros.

La salida del detector de fase y la salida del algoritmo FACS se presenta en la Figura 14. Desde la Figura 14 se puede observar que el sincronizador FACS permite lograr un desfase de 0° entre la portadora y la señal del NCO. Puede observarse también que el detector de fase tiene un retardo de 60 muestras aproximadamente para responder a los cambios de fase del NCO.

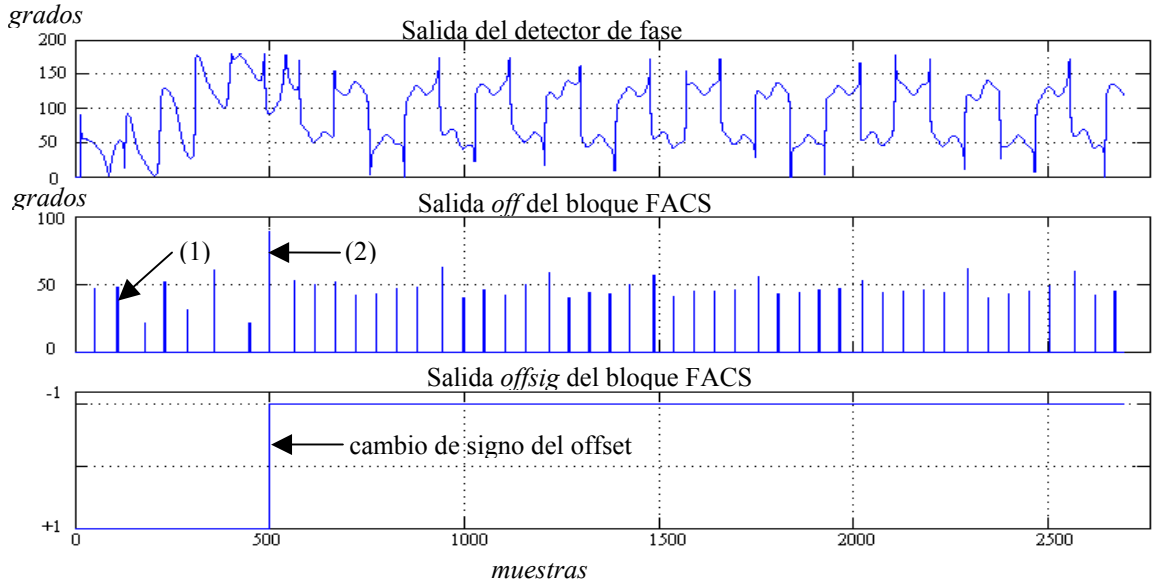

Figura 15: *Resultados de simulación para el detector de fase y FACS*

La segunda simulación tiene como entrada una señal BPSK con los siguientes parámetros:

- Desfase de la portadora: 60°
- Diferencia en frecuencia entre la portadora y el oscilador local: 12Hz.
- Bits modulados: secuencia de unos y ceros.

La salida del detector de fase y las salidas *off* y *offsig* del algoritmo FACS se presentan en la Figura 15. Desde la Figura 15 puede observarse como el algoritmo FACS mantiene una salida periódica para seguir la portadora a pesar de la diferencia en frecuencia, obteniéndose un resultado similar a la simulación realizada con Matlab. Además, puede observarse el cambio de signo del offset cuando el valor del desfase es lo suficientemente grande (mayor a 62°), permitiendo un adecuado seguimiento de la portadora. Como se puede observar desde la Figura 15, la señal off controla adecuadamente la fase del NCO, por ejemplo, en el instante señalado como (1) en la gráfica, el algoritmo FACS le indica al NCO que debe realizar un cambio de fase de 50°; en el instante señalado como (2), debido a que el offset es mayor a 62°, ocurre un cambio de signo del offset.

6. CONCLUSIONES Y TRABAJO FUTURO

En este artículo se presenta el diseño en hardware de un nuevo algoritmo usado para la sincronización de portadora en BPSK. La verificación funcional del algoritmo FACS se llevó a cabo a nivel de simulación usando Matlab, es decir, se realizaron las comparaciones entre las respectivas simulaciones de funcionamiento del algoritmo FACS y el DPLL. En este caso, el algoritmo FACS presentó un tiempo de adquisición de portadora menor y permite la sincronización de la frecuencia de la portadora en un rango de frecuencias mayor que el DPLL.

El diseño del algoritmo FACS se implemento en un FPGA Spartan-3 XC3S200 empleando System Generator. En este caso el diseño usa 13 CLBs, lo cual es muy adecuado desde el punto de vista de área. Los resultados obtenidos permiten concluir que el diseño propuesto puede ser empleado como un "core" para aplicaciones de de alto desempeño en Radio Software.

El trabajo futuro será orientado hacia el diseño de "cores" para Radio Software con el propósito de diseñar un procesador de banda base y frecuencia intermedia con características de muy alto desempeño.

7. REFERENCIAS

[1] J. Mitola, *Software Radio Architecture*,.John Wiley & Sons, Inc, 2000.

[2] Michael Rice, Chris Dick, y fred harris, "Maximum Likelihood Carrier Phase Synchronization in FPGA-Based Software Defined Radios," *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, May. 2001.

[3] Software Defined Radio. Altera Corporation. 2005. http://www.altera.com/solutions/comm/wireless/software/sdr/wir-sdr.html.

[4] Wireless Communication: Baseband. Xilinx Inc. 2005. http://www.xilinx.com/esp/wireless/baseband/index.htm.

[5] Chris Dick, Fred Harris y Michael Rice, "FPGA Implementation of carrier Synchronization for QAM receivers," *Journal of VLSI Signal Processing Systems,* Volumen 36, pp. 57-71, Ene. 2004.

[6] F.J. Harris y C.H. Dick, "On Structure and Implementation of Algorithms for Carrier and Symbol Synchronization in

Software Defined Radios," *EUSIPCO-2000, "Efficient Algorithms for Hardware Implementation of DSP Systems," Tempere, Finland,* Sept. 2000.

[7] John Proakis, *Digital Communicactions 4ed*, Mc Graw Hill. 2000.

[8] Sistemas de Control Automático. KUO, Benjamin. Prentice Hall. 1996.

[9] Introduction to phase-locked loop system modeling. LI, Wen y MEINERS, Jason. Texas Instruments: Analog Application Journal. 2000.

[10] Marvin Frerking, *Digital Signal Processing in Communication Systems*, Kluwer Academic Publishers. 2003.

[11] Xilinx Inc., System Generator for DSP, http://www.xilinx.com/xlnx/xebiz/designResources/ip_product_details.jsp?key=dr_dt_system_generator

[12] Xilinx Core Generator, http://www.xilinx.com/products/logicore/coregen/index.htm