IMPLEMENTACIÓN DE FILTRO FIR EN SISTEMAS PROGRAMABLES EN UN *CHIP*

I. Torres[1], Y. Padrón[2] , Y. Hernández[3], A. Taboada[4]

[1] *Universidad Central de Las Villas, Centro de Estudios de Electrónica y Tecnologías de la Información itrodriguez@uclv.edu.cu*

[2] *Centro Provincial de Ingeniería Clínica y Electromedicina, Sancti Spiritus ypadron@electromed.ssp.sld.cu*

[3] *Centro Provincial de Ingeniería Clínica y Electromedicina, Matanzas electromed.mtz@infomed.sld.cu*

[4] *Universidad Central de Las Villas, Centro de Estudios de Electrónica y Tecnologías de la Información ataboada@uclv.edu.cu*

RESUMEN

Los filtros (analógicos y digitales) se usan para la separación de señales combinadas y la restauración de señales que presentan alguna distorsión. Los filtros digitales alcanzan mejores resultados, con precisión limitada por errores de redondeo en la aritmética empleada. El diseño de un filtro digital con respuesta finita al impulso (FIR) y la cuantización de sus coeficientes para facilitar su implementación en PSoC es el tema de este trabajo.

Palabras claves: filtrado FIR, cuantización, PSoC.

Palabras claves: escriba un mínimo de tres palabras claves, en orden alfabético ascendente

Implementation of fir filters in programmable systems on chip (PSoC)

ABSTRACT

Filters (analogue and digital) are used for combined signal separation and restoration of signals that have some distortion. Digital filters achieve better results, with limited accuracy depending on rounding errors in arithmetic employed. The design of a digital filter with finite impulse response (FIR) and the quantization coefficients to facilitate its implementation in PSoC is the subject of this paper.

.

Key words: filter FIR, quantization, PSoC.

INTRODUCCIÓN

Distintas formas de filtrado se han implementado en la industria de los Procesadores Digitales de Señales (DSP). Sin embargo, varias aplicaciones no requieren de tan altas tasas de muestreo y costo computacional, parámetros que generalmente están asociados a los DSP [1]. Una alternativa para la implementación es el uso de microprocesadores o microcontroladores sencillos, como los sistemas programables en un chip (PSoC) de Cypress Microsystems [2].

Lamentablemente, las señales se presentan contaminadas con ruidos y artefactos. Se requiere el uso de filtros para mejorarlas, dentro de los que se destacan los filtros digitales de respuesta finita impulsiva (FIR), que se pueden implementar en los sistemas programables en un chip (PSoC).

Un filtro FIR es no recursivo dado que comprende una serie de elementos de retardo conectados en serie, donde la salida depende solo de la entrada presente y de las pasadas [3]. Cada muestra de entrada (x) y sus retardos respectivos se multiplican por los coeficientes del filtro (b), necesitándose así una multiplicación para cada elemento de retardo. Un bloque genérico de cómo se presenta un filtro FIR se muestra en la Figura 1.

Para un sistema lineal invariante en el tiempo, la respuesta finita al impulso (FIR) puede ser descrita como se muestra en la ecuación (1) [4].

$$y[n] = b_0x[n]+b_1x[n-1]+\ldots+b_{M-1}x[n-N+1] \qquad 1)$$

donde:
x[n]: Señal de entrada,
y[n]: Señal de salida y
b[i]: Coeficientes del filtro.

A diferencia de los filtros IIR, un filtro FIR siempre produce una respuesta estable a la salida. Además, provee fase lineal, tanto cuando el filtro es simétrico o no [4]. Cuando un filtro FIR es implementado en PSoC, el valor absoluto de los coeficientes debe ser menor que la unidad, $|b[i]| \leq 1$, $\forall i$[2]. Si el valor de los coeficientes no se encuentra en ese rango, pueden usarse una preescala y una postescala.

Una herramienta de diseño de filtros puede ser generada con el código de los sistemas programables en un chip (PSoC) y los coeficientes del filtro FIR [2], que han sido obtenidos para la implementación utilizando, por ejemplo, MATLAB o ScopeFIR.

La Tabla 1 posee algunas características de dos dispositivos de la familia de los PSoC, que pueden ser utilizados en la implementación de filtros FIR[5-6].

La familia de los PSoC de Cypress MycroSystems consiste en varios bloques de señal mezclada con controlador en un chip. Este dispositivo es diseñado para remplazar múltiples componentes de un sistema microcontrolador (MCU) en un solo componente, siendo esta una de sus principales ventajas. Ser un componente programable en un solo chip disminuye su costo. Incluye bloques analógicos y digitales, así como interconexiones programables. Permite crear configuraciones a la medida en función de la aplicación requerida. Incluyen una rápida unidad central de procesos (CPU), una memoria flash, una memoria de datos SRAM y terminales de entrada y salida configurables[5-6].

El PSoC ofrece una amplia gama de convertidores análogo/digitales (C A/D), de los cuales seleccionar uno depende de los requisitos de la aplicación, teniendo en cuenta: resolución, porciento de uso de la CPU, latencia, linealidad, ruido, consumo de potencia y de otros recursos (bloques analógicos y digitales, RAM, flash) [7].

MATERIALES Y MÉTODOS

Programación del PSoC

En la programación del PSoC son utilizados dos bloques analógicos y uno digital, seleccionados en color verde (Figura 2), para el C A/D sigma-delta de 11 bits (DELSIG11). Cuando se selecciona el convertidor, se conecta el Puerto 2_2 como entrada[8].

Los C A/D sigma-delta presentan mayor inmunidad al ruido, además, las señales de entrada son muestreadas a una tasa mucho mayor que la de Nyquist [9-10], lo que permite prescindir del amplificador de muestreo/retención. Están compuestos por un modulador sigma-delta, un filtro digital y un diezmador [8].

Filtro FIR en PSoC

La mayoría de los filtros FIR se implementan usando convolución circular. Un filtro FIR puede ser visto como la convolución entre los coeficientes del filtro, de longitud N, con la señal de entrada y sus retardos (ecuación 1). La organización en memoria de esta ecuación se ilustra en la Figura 3.

Con el objetivo de ver la importancia del buffering circular se computa la salida en un tiempo n+1, para encontrar y[n+1]. Entonces la organización de la memoria en un tiempo n+1 queda tal como se muestra en la Figura 4.

De las **¡Error! No se encuentra el origen de la referencia.** y **¡Error! No se encuentra el origen de la referencia.** se concluye que dado un tiempo determinado, la salida del filtro solo involucra N multiplicaciones entre la entrada y sus retardos con los coeficientes del filtro. El primer coeficiente del filtro, h[0], es multiplicado por la mayoría de las muestras de entrada, mientras que el último coeficiente, h[M-1], está multiplicado por las muestras más antiguas en el buffer.

El camino más eficiente para manejar estas muestras guardadas es a través del buffering circular. Primeramente, se asigna un puntero a las localizaciones de memoria consecutivas, como se ilustra en la Figura 5. En un tiempo t el puntero es ubicado en una localización baja de memoria (parte izquierda de la Figura 5), usualmente apuntando a la muestra más antigua, que será analizada. Cuando se obtiene la nueva muestra del C A/D, el valor se pone en este lugar y el puntero pasa a la próxima dirección (Figura 5 centro) antes de comenzar la convolución.

El puntero se localiza nuevamente en la posición de la muestra más antigua. La convolución comienza por la multiplicación de los retardos y los coeficientes del filtro, uno por uno. Entonces los coeficientes del filtro deben de ser organizados en orden descendente: h[M-1],…h[0].

Si el puntero en el buffer circular alcanza el fin de la memoria, este regresa a la posición de comienzo. Después de la convolución, el puntero se mueve a la posición más antigua de la memoria y espera la siguiente muestra. Luego el proceso se repite desde el principio (Figura 5 derecha). Las principales operaciones que se realizan en la convolución son la multiplicación y la adición. La función de multiplicación aquí usada aparece en la Nota de Aplicación AN2038 [5].

Algoritmo de Convolución en PSoC

Con el objetivo de cumplir la explicación antes expuesta, la programación del PSoC responde al diagrama de flujo de la Figura 6.

El proceso de filtrado comienza con la inicialización, donde se define la fuente de la señal de entrada (SE), que pueden ser los datos que han sido previamente guardados en un espacio de la Memoria Flash Serie del dispositivo (Filtrado

fuera de línea), o las muestras que se obtienen desde un C A/D (filtrado en línea). Es necesario definir también el destino de la señal filtrada (SF).

En dependencia de la frecuencia de corte escogida, se carga la tabla de coeficientes (TC) que responde a los parámetros del filtro previamente diseñado. Se definen las direcciones de los buffers de entrada (BE) y de salida (BS). El buffer de entrada (BE) se llena según el número de coeficientes del filtro, para así garantizar igual número de retardos que de coeficientes.

Luego se procede a realizar la convolución entre la tabla de coeficientes y las muestras que se encuentran en el buffer de entrada. Cada muestra multiplicada y sumada se envía al buffer de salida (BS) y luego al destino de la señal filtrada (SF).

En el caso del diagrama de flujo de la Figura 6, se pueden seleccionar dos juegos distintos de coeficientes: uno con frecuencia de corte igual a 40Hz y otro para 100Hz. Los datos de estos ficheros se guardan en la memoria flash del dispositivo.

Diseño del filtro

Para hacer las pruebas, se utilizaron filtros con longitud 64 y ventana de Kaiser con β igual a 6, con el objetivo de reducir el efecto de los lóbulos secundarios. Se escogió Kaiser porque solo variando el valor de β se obtiene una aproximación de las otras ventanas [11].

Una forma de mejorar la respuesta de un filtro FIR es incrementar su número de coeficientes o el número de bits en la representación de estos coeficientes [12]. Estas posibles formas de mejorar la respuesta del filtro incrementan en gran medida la complejidad computacional. Los coeficientes son reformados antes de su uso en el PSoC, estos cambios se realizan para aprovechar las potencialidades del microprocesador de 8 bits que posee el mismo (Tabla 1) y hacer la mayor cantidad de coeficientes iguales a cero para reducir el número de multiplicaciones [12]. Con este objetivo se cuantizan los coeficientes de los filtros, tratando de que sea precisamente 8 la longitud de la palabra cuantizada.

Los coeficientes pueden ser cuantizados utilizando los métodos clásicos de cuantización que brinda MATLAB, o también mediante la implementación serie o paralelo que se proponen en [12], teniendo en cuenta incluir en la ROM el valor de la flag, que indica cuando el coeficiente ha sido cuantizado a más de 8 bits. Es necesario que las condiciones iniciales especificadas para el diseño de los filtros no varíen considerablemente, al punto de afectar la respuesta de frecuencia de estos. La Figura 7 muestra la respuesta de frecuencia de un filtro paso bajo con frecuencia de corte igual a 40 Hz y la respuesta luego de efectuar la cuantización a 8, 9 y 10 bits.

La respuesta al impulso del filtro original y la de los coeficientes cuantizados a 8 bits se muestran en la Figura 8, de 64 coeficientes iniciales distintos de cero, solo quedan 49, después de cuantizar.

Estos coeficientes son transformados a formato hexadecimal para su posterior uso en el PSoC y guardados en los ficheros correspondientes, estos están ordenados en forma descendiente, comenzando por el bit más significativo.

RESULTADOS Y DISCUSIÓN

Los códigos en ensamblador correspondientes al algoritmo de convolución se muestran a continuación.

```
 ;;-----------------------------------------
FIRInit:
_FIRInit:
;limpia todos los retardos de buffering
mov A, 0
mov [cTemp+0], iDn
mov [cTemp+1], Ncdelaytaps
call BufferInit
;mueve el punter al comienzo
mov [ptr_DelayTaps],iDn
;actualiza la dirección de los retardos del FIR(16
bits)
mov [ptr_DelayTaps+1],0
;;-----------------------------------------
;;  BufferInit:
;;  Initialize buffers with constant
;;-------------------------------------------------
----------------
 BufferInit:
_BufferInit:
  LOOP1:
    cmp [cTemp+1], 0 ;compara el número
                        ;de retardos con cero
;si ZF=1 (si son iguales) salta a EXIT1
;el buffer esta limpio
    jz EXIT1
    mvi [cTemp+0], A ;limpia el buffer, mueve a A, que
tiene cero
    dec [cTemp+1]    ;decrementa el Ncdelaytaps y
vuelve a chequear que ;;quede cero
    jmp LOOP1
  EXIT1:
   Ret
```

Código 1. Inicialización del Buffer Circular

Cuando la nueva muestra es recuperada del C A/D, esta es salvada al buffer circular y el puntero es movido a la posición de la próxima muestra, siendo incrementado.

```
;;-----------------------------------------
;;FIRFiltering:
;; Generando  filtrado FIR
;;-----------------------------------------
FIRFiltering:
_FIRFiltering:
;salva la nueva muestra, 16 bits, e incre ;menta el
puntero
mov [cTemp],A ; porque esta dividida en LBM ;y HBM
mov A,X ; ;porque esta dividida en LBM y HBM
mvi [ptr_DelayTaps],A ; ; mueve X para ;ptr_delaytaps e
incrementa la ;;posición del ;puntero
mov A, [cTemp]
mvi [ptr_DelayTaps],A
IncrementwithCircular
(ptr_DelayTaps+0,(iDn+0), NUMFIRDELAY
; Convolución
mov [ctr_Hn], 0 ;reset el contador del fil ;tro
mov [cTemp+0], 0 ;reset 32 bits del ACC,
mov [cTemp+1], 0
mov [cTemp+2], 0
mov [cTemp+3], 0
mov [ctr1],NUMFIRDELAY ;cargo el número de
;multiplicaciones, 65
STARTConvolution:
cmp [ctr1], 0 ;comparo si es cero salgo, no ;hace nada
jz EXITConvolution
mvi A, [ptr_DelayTaps] ;carga 4 bits(Parte ;alta)
mov [icurXn],A
mvi A, [ptr_DelayTaps] ;carga 4 bits (Parte ;baja)
mov [icurXn+1],A
IncrementwithCircular
(ptr_DelayTaps+0),(iDn+0), NUMFIRDELAY ;chequeo si
necesito retornar el ;cursor
mov [ptr_curHn],icurHn
```

```
mov A, [ctr_Hn] ;lo que está en ctr_Hn lo ;pongo en A
index FIRCoefficients ; Apunto a la tabla de
;coeficientes
mvi [ptr_curHn], A ; load 4 bit (HIGH)
add [ctr_Hn], 1
mov A, [ctr_Hn]
  cmp [b4], 0
jnz FIR100
index FIRCoefficients40
jmp Cont
FIR100:
index FIRCoefficients100
Cont:
mvi [ptr_curHn], A ; carga 4 bits (Parte ba ;ja)
add [ctr_Hn], 1  ;esto lo hago para cargar ;el próximo
coeficiente
MultiplyAndSum8s_8s_8s (cTemp), (icurXn),(icurHn) //
Multiplica 8 //bits con 8 bits y se guardan en 8
dec [ctr1] ; decrementa el número de retar ;dos del
filtro
jmp STARTConvolution
EXITConvolution:
call ShiftLeft32BitsBuffer
mov X, [cTemp]
mov A, [cTemp+1]
ret
;-----------------------------------------
;;  ShiftLeft32BitsBuffer:
;;  Para correr las muestras a la izquierda
ShiftLeft32BitsBuffer:
_ShiftLeft32BitsBuffer:
   mov  A,0
   asl  [cTemp+1];la carpeta con las mues ;tras
temporales se corre hacia ;la izquierda
   adc  A,0
   asl  [cTemp+0]
   add  [cTemp+0],A
   mov  A,0
   asl  [cTemp+2]
   adc  A,0
   add  [cTemp+1],A
   mov  A,0
   asl  [cTemp+3]
   adc  A,0
   add  [cTemp+2],A
ret
```

Código 2. Filtro FIR Genérico

```
;-----------------------------------------------
macro IncrementwithCircular
push A
inc [@0+1] ;incrementa el puntero
mov A,@2
sub A, [@0+1]
jnz ENDIncrementwithCircular
mov [@0],@1 ;si alcanza el fin, se pone de nuevo en el
comienzo
mov [@0+1],0 ;carga el contador con cero
ENDIncrementwithCircular:
pop A
endm
```

Código 3. Incremento del puntero

En la Figura 9 se muestra una de las tablas de coeficientes obtenidas para su uso en el PSoC. En ellas quedan especificadas diferentes variables que se usan en la programación de este dispositivo. Dentro de estas variables podemos mencionar el Número de Coeficientes del Filtro (NUMFIRCOFF), que es igual al Número de Retardos (NUMFIRDELAY), que a su vez es igual al número de multiplicaciones y sumas que se realizan en la convolución. Con STARTDELAYINPUT se define el lugar donde se ubica el puntero para comenzar a entrar las muestras a filtrar y STARTDELAYCTR es un contador usado para el buffering circular.

Los datos de esta tabla se guardan en la memoria flash del dispositivo. Esto queda definido al utilizar la palabra ROM en la directiva area. La palabra REL, en la directiva, permite al dispositivo reubicar los códigos o datos.

CONCLUSIONES

En este trabajo, presentamos la implementación de un filtro FIR en PSoC. Los coeficientes de los filtros que cumple con las características de diseño deseadas se obtienen con la ayuda del MATLAB. Los coeficientes obtenidos y mostrados en la Figura 9 pueden ser usados directamente en proyectos con PSoC DesignerTM [5].

La implementación de filtros FIR en PSoC puede encontrar distintas aplicaciones, como la detección de frecuencias instantáneas. Además, se pueden usar en el diseño de equipos portátiles y de bajo costo, como los equipos electrocardiográficos de monitoreo constante (Holter), aprovechando las potencialidades de la gran gama de C A/D que posee este dispositivo.

REFERENCIAS

1. **A. Z. B. Varnagiryte, O. Olsen, P. Koch, O. Wolf, E. Kazanavicius,** "A practical approach to DSP code optimization using compiler/architecture," vol. 2, 2002.
2. **P. D. Somsak Sukittanon, Stephen G. Dame,** "Application Note AN2328: FIR Filtering in PSoC™ with Application to Fast Hilbert Transform," Cypress Perform, 2005.
3. **S. Winder,** Analog and Digital Filter Design, Second ed. United States of America: Newnes, 2002.
4. **W. J. Tompkins,** Biomedical Digital Signal Processing New Jersey: Prentice Hall, 2000.
5. **C. S. Corporation,** "CY8C24094, CY8C24794,CY8C24894 and CY8C24994 PSoC™ Mixed-Signal Array Final Data Sheet," Cypress Perform, March 30, 2010.
6. **C. S. Corporation, "**CY8C27143, CY8C27243, CY8C27443, CY8C27543, CY8C27643 PSoC™ Mixed Signal Array Final Data Sheet," Cypress Perform, February 10, 2010.
7. **D. Seguine,** "Application Note AN2239: ADC Selection," Cypress MicroSystems, January 2005.
8. **J. Y. Saavedra-Arramendi,** "Control y adquisición de señales del ECG-6101 desde una computadora," CEETI, Central Marta Abreu de Las Villas, Santa Clara, 2009.
9. **H. Baher,** Analog & Digital Signal Processing: Wiley, 2001.
10. **W. A. Kester and A. Devices,** Data conversion handbook: Newnes, 2005.
11. **I. Torres-Rodríguez,** "Implementación de filtrado FIR en PSoC para Procesamiento de Señales Biomédicas," CEETI, Central Marta Abreu de las Villas, Santa Clara, 2010.
12. **Z. Shen,** "Improving FIR Filter Coefficient Precision," IEEE Signal Processing Megazine, vol. 27, p. 120, 2010.

AUTORES

Idileisy Torres Rodríguez, Ingeniera Biomédica, Centro de Estudios de Electrónica y Tecnologías de la Información (CEETI), Carretera a Camajuaní, km 5 ½, Santa Clara, VC, CP 54830, 0142-281157, itrodriguez@uclv.edu.cu. Graduada de Ingeniería Biomédica en La Universidad Central de Las Villas en julio del 2010, actualmente se desempeña como Reserva Científica en el CEETI, investiga en el area del Procesamiento Digital de Señales e Imágenes.

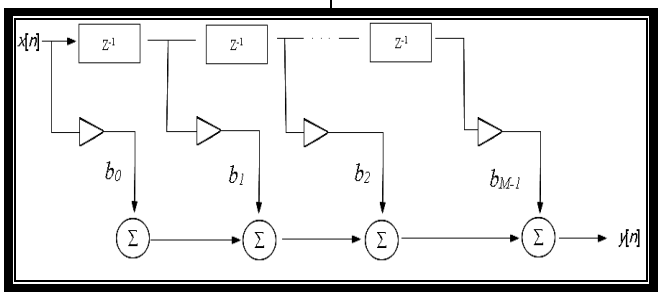

Figura 1. Estructura común de un filtro FIR.

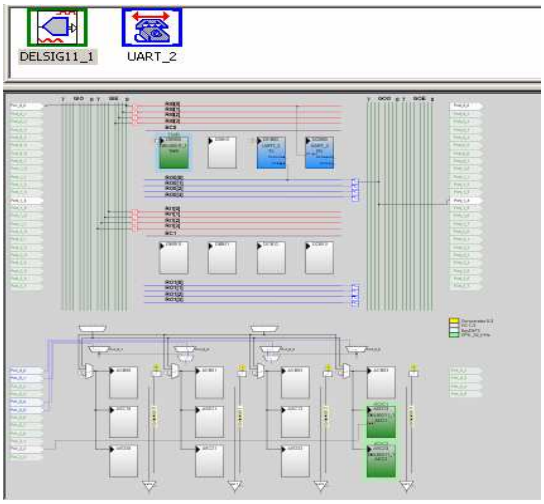

Figura 2. Bloques seleccionados para el CA/D y comunicación serie.

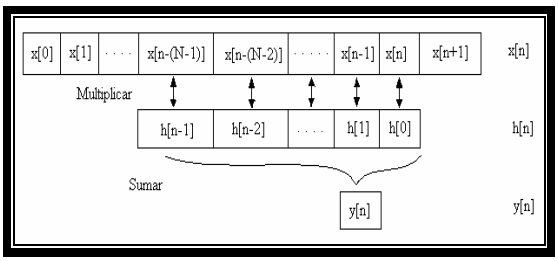

Figura 3. Organización en memoria de la salida del filtro en un tiempo n.

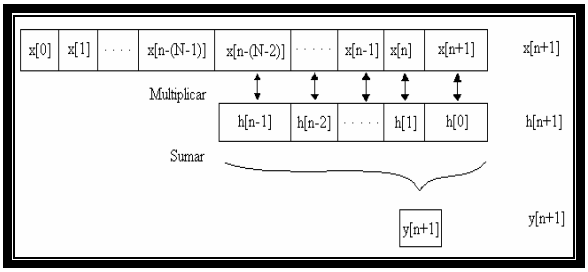

Figura 4. Organización en memoria de la salida del filtro en un tiempo n+1.

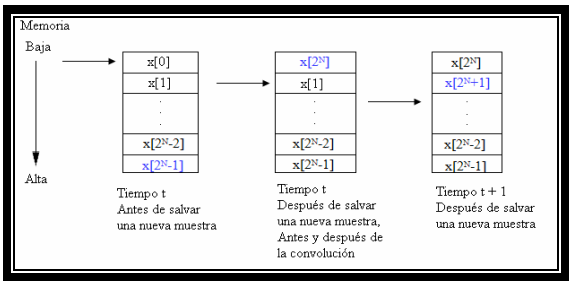

Figura 5. Buffering circular aplicado a las entradas retardadas.

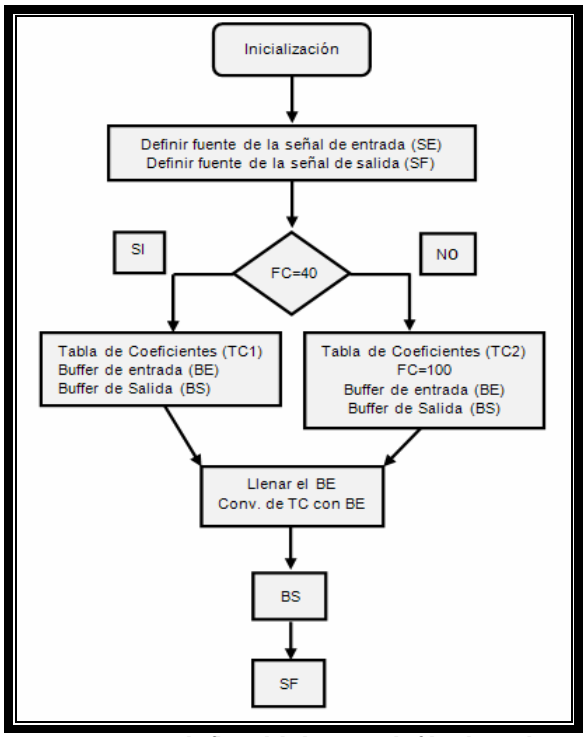

Figura 6. Diagrama de flujo del algoritmo de filtrado en el PSoC

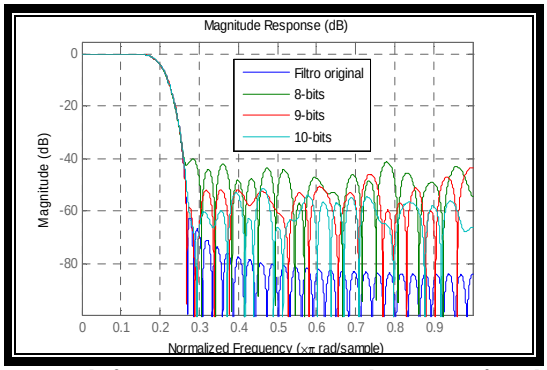

Figura 7. Respuesta de frecuencia (40Hz) cuantizada con aritmética de punto fijo.

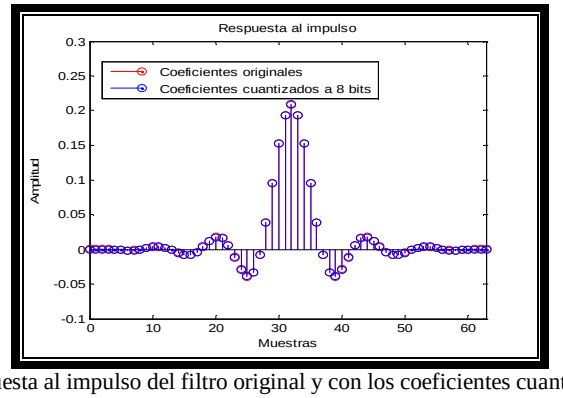

Figura 8. Respuesta al impulso del filtro original y con los coeficientes cuantizados a 8 bits.

```
NUMFIRCOFF:      equ     65
NUMFIRDELAY:     equ     NUMFIRCOFF
;Only use for hilbert transform
ALPHA:           equ     32
STARTDELAYINPUT:   equ     2*(ALPHA+1)
STARTDELAYCTR:     equ     ALPHA+1


area text(ROM,REL)
.LITERAL


FIRCoeficientes:
0,  0,  0,  0,  0,  0,  1,  1,  0,  0,  0,  0,  0,  0,  F,  F,  F,
1,  0,  1,  2,  2,  1,  F,  F,  F,  F,  F,  5,  C,  1,  1,  1,  1,
1,  C,  5,  F,  F,  F,  F,  F,  1,  2,  2,  1,  0,  1,  F,  F,  F,
0,  0,  0,  0,  0,  0,  1,  1,  0,  0,  0,  0,  0,  0,
.ENDLITERAL
```

Figura 9. Tabla de coeficientes para el PSoC

Tabla 1. Características principales de CY8C27443 y CY8C24894

Características	CY8C24894	CY8C27443
Microprocesador empotrado	M8C, Arquitectura Harvard	
Velocidad del procesador	24MHz (4MIPs)	
Bloques analógicos	6	12
Bloques digitales	4	8
Multip. por hardware	1 de 8x8	2 de 8x8
Memoria flash	16k	

RAM	1k	256 Bytes
Interfaz USB	sí	no
Voltaje de trabajo	3V - 5,25V	
Rango de temperat.	-40°C a +85°C	
No. de terminales	56	28