Filtro digital reconfigurable basado en FPGA

Franco Bocalon, Luis Guanuco, Santiago Nolasco
Procesamiento Digital de Señales
Especialidad en Sistemas Embebidos
Instituto Universitario Aeronáutico

***Resumen*—La flexibilidad que proporciona el dominio digital en lo que respecta a la síntesis de filtros se ha incrementado para mejor con las nuevas tecnologías. La alta velocidad de procesamiento de los microcontroladores de bajo costo y el acceso a plataformas basadas en FPGA permiten completar el estudio en profundidad de los filtros digitales. Aquí se presenta una implementación de un filtro FIR reconfigurable con la integración de diferentes software utilizados en el módulo *Procesamiento de Señales Digitales*.**

***Palabras claves*—DSP, FIR filter, FPGA, soft-core**

I. Introducción

Con el objeto de analizar el comportamiento de los filtros digitales,se implementan cuatro tipos de filtros: *pasa-bajo*, *pasa-alto*, *pasa-banda* y *suprime-banda* según los siguientes requerimientos:

- Orden del filtro N=12
- Longitud de palabra 13 bits
- Pequeña interfaz de usuario para elegir y cargar los coeficientes en la plataforma embebida a traves del puerto serie.

II. Modulado

Para poder tener los coeficientes necesarios para la implementacion de un filtro FIR se utilizo la herramienta *FDATool* del paquete *MathWorks*. La principal característica de estos filtros es un número finito de términos no nulos como respuesta a una señal impulso.

FDATool permite seleccionar el tipo de filtro, orden y ventana a implementar. En nuestro caso el tipo de ventana seleccionada es la Kaiser. Ésta proporciona un mejor control de la pendiente y perturbación en la banda de paso.

En la Figura 1 se puede apreciar la respuesta del filtro. Claramente se observa la frecuencia de corte y los lóbulos atenuados para las frecuencias mayores en el caso del filtro pasa bajo.La amplitud de ganancia esta indicada con color azul y la fase en rojo, es muy importante de destacar que en estos filtros se pueden diseñar si asi el diseño lo requiera, una respuesta de fase lineal.

Una vez generados los coeficientes, disponibles en un array en el workspace de MATLAB, pueden ser enviados a la plataforma embebida (basada en una FPGA). Se deben cuantizar los coeficientes, es decir enviarlos en el formato de palabra de trabajo de la placa, en nuestro caso particular 16bits sin signo. Por lo tanto los generados coeficientes por FDATool tienen que ser convertidos desde punto flotante a punto fijo y cambiar el tipo de notación con signo a sin signo. Esta

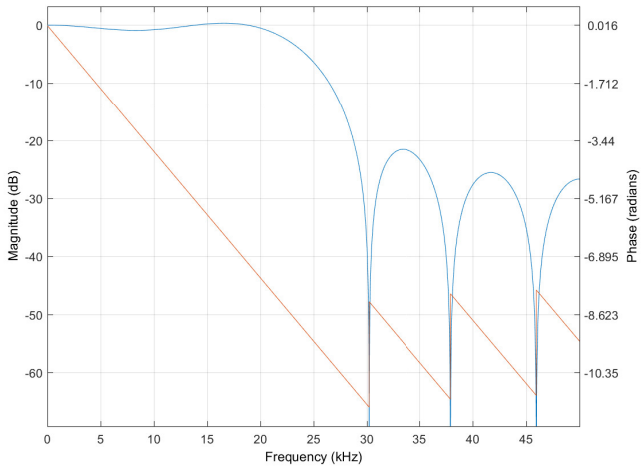

Fig. 1. Respuesta en frecuencia y fase del filtro sintetizado por TDATool.

conversión se realizo a través del siguiente algoritmo escrito en un script de MATLAB.

```
%%%%%%FiltroV1%%%%
  clc
  close all
  delete(instrfind({'Port'},{'/dev/ttyUSB1'}));

  condition = 1;
  while condition
  % do something
  nroFiltro = input ('Elija su filtro: 1_Pbajo
  2_Palto 3_Pbanda 4_Sbanda\n');
  if((nroFiltro < 5)&&(nroFiltro > 0))
  condition=0;
  end
  end
  % Signal inputs
  % u = evalin( 'base', 'Num' )
  switch nroFiltro
  case 1
  u=Num;
  case 2
  u=Num1;
  case 3
  u=Num2;
  case 4
  u=Num3;
  otherwise;
  end
```

El cual consta de una minima interfaz grafica que permite al usuario elegir que filtro quiere implementar en la FPGA y luego enviarlo a traves de un puerto estandar RS232.

```
  FM=10;
  Mayor=max(u)*FM;
```

```
  Normal=u*1000;
  Bits=12;
  Normal=round(Normal);
  Q=quantizer('fixed','round','saturate',[Bits 0]);
  hexh=num2hex(Q,Normal);
  Binario=num2bin(Q,Normal);
  Entero= num2int(Q,Normal);
  h=hex2dec(hexh);
  l=bin2dec(Binario);
  % Vector cuantizado 1x13
  % VectorCoef = quantize (Vector,1,13,0); % Con siNumgno,
  % 13bits y sin fraccional
```

```
  % Trasmision serie
  s=serial('/dev/ttyUSB1');%se crea un objeto del puerto
  s.Baudrate=115200;      % configuracion puerto
  s.DataBits=8;
  s.Parity='none';
  s.StopBits=1;
  s.FlowControl='none';
  s.RequestToSend='off';
  s.OutputBufferSize=14;  % es el numero de bytes a enviar
  s.InputBufferSize=14; % es el numero de bytes a recibir
  s.Timeout=5;            % 5 segundos de tiempon de espera
  set(s,'Terminator','CR/LF');% caracter fin de envio
  fopen(s);               % se abre el objeto
  % FIR COEFF
  % Escritura de bytes iniciales para
  % iniciar la conversion en la fpga
  K1= [];
  fwrite(s,250,'uint8'); %tx 0xFA bit menos significativos
  fwrite(s,255,'uint8'); %tx 0xFF
  K1 = [K1 fread(s,1,'uint16')];
  i=0;
  for i=1:length(l);
  fwrite(s,l(i),'uint16');
% se envia entero sin signo de 8 bits
  K1 = [K1 fread(s,1,'uint16')];
  end
  fclose(s);
  delete (s);
  clear s;
```

Se utilizaron funciones propias de MATLAB para cuantizar y convertir los coeficientes negativos a sin signo. Por último, es interesante destacar que eligiendo el número de coefcientes par o impar podemos tener una simetría o asimetría en los valores de los coeficientes, por ejemplo el filtro pasa-bajo es con todos los coeficientes positivos y con pariedad.

III. Implementación

Se utiliza la plataforma *BASYS2* y los módulos *PmodAD1* y *PmodDA2*, todos fabricados por Digilent Inc.[1].

IV. Conclusiones

Suspendisse erat mauris, nonummy eget, pretium eget, consequat vel, justo. Pellentesque consectetuer erat sed lacus. Nullam egestas nulla ac dui. Donec cursus rhoncus ipsum. Nunc et sem eu magna egestas malesuada. Vivamus dictum massa at dolor. Morbi est nulla, faucibus ac, posuere in, interdum ut, sapien. Proin consectetuer pretium urna. Donec sit amet nibh nec purus dignissim mattis. Phasellus vehicula elit at lacus. Nulla facilisi. Cras ut arcu. Sed consectetuer. Integer tristique elit quis felis consectetuer eleifend. Cras et lectus.

[1] https://digilentinc.com/

Referencias

[1] Umair Ashraf, *CS 415 N-Tier Application Development*. National University of Computer and Emerging Sciences. July 5, 2013.

[2] Bruno Pedro, *Is REST better than SOAP? Yes, in Some Use Cases*, url: http://nordicapis.com/rest-better-than-soap-yes-use-cases/.