

# **Teoría de los circuitos II - Ampliado**

**Trabajo final**

Filipuzzi, Fernando Rafael

6 de febrero de 2023

# **Índice general**

# **Índice de figuras**

# 1 Introducción

El objetivo de este trabajo es llevar los aspectos teóricos de diseño y realización de filtros digitales a la práctica.

En las primeras secciones se busca reunir las definiciones, expresiones y criterios que sirvan de referencia en el desarrollo de los script de diseño de los filtros en Matlab.

En la secciones finales se trata la implementación de los filtros, describiéndose el hardware utilizado y las especificaciones de los filtros realizados.

## 2 Representación de las funciones del circuito

### 2.1. Comentarios sobre la bibliografía

Weinberg. Network analysis and synthesis

- *En el cap6, pag 134. Repaso de transformada de Laplace*
- *En el cap8, pag 215. Bode, Diagrama de polos y ceros*

# **3 Teoría de cuadripolos**

## **3.1. Comentarios sobre la bibliografía**

Lam Harry. Analog and digital filters design and realization.  
*Desarrollo general.*

W. L. Everitt. G. E. Aner. Ingeniería de comunicaciones  
*Desarrollo sobre teoría imagen*

## 4 Filtros por Teoría imagen

### 4.1. Comentarios sobre la bibliografía

Lam Harry. Analog and digital filters design and realization.  
*Desarrollo general.*

W. L. Everitt. G. E. Anero. Ingeniería de comunicaciones  
*Desarrollo sobre teoría imagen*

# 5 Filtros por funciones de aproximación

## 5.1. Introducción

### Filtro ideal

Un filtro ideal es una red circuital que transmite libremente las señales con componentes en frecuencias dentro de su banda de transmisión libre, y rechaza a las componentes frecuenciales fuera de esa banda. Un filtro ideal no introduce distorsión y debería mantener la relación de fase. El filtro puede introducir un retraso temporal en la señal resultante, pero todas las componentes de frecuencia deberán retrasarse igualmente.

En la banda de rechazo (o suprimida) los filtros no absorben potencia, es decir que rehúsan admitir potencia en sus terminales de entrada. Estando dada la respuesta ideal en frecuencia por la función magnitud tal como

$$|H(j\omega)|^2 = \begin{cases} A & \text{para } \omega_1 \leq \omega \leq \omega_2 \\ 0 & \text{de otro modo} \end{cases}$$

El intervalo o banda de frecuencia  $\omega_1 \leq \omega \leq \omega_2$  se conoce como banda libre o banda de paso y el desplazamiento en dicha banda es cero o  $180^\circ$ , o es lo mismo que decir que la función de fase es proporcional a la frecuencia, Fig. ??.

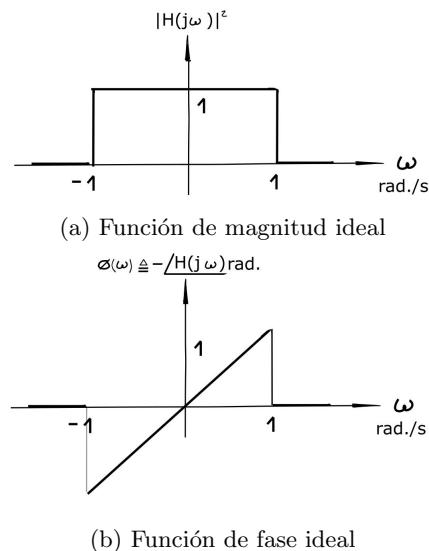


Figura 5.1: Características en frecuencia de un Filtro ideal pasa bajo

Entonces, el problema de diseño radica en que ningún circuito lineal de elementos concentrados puede producir tal función de transferencia exacta. En resumen tenemos que:

- 1- En un filtro realizado con elementos pasivos se tiene como respuesta en frecuencia una función racional.
- 2- Una función racional no puede tener una valor constante en ninguna banda.

### El problema de la aproximación.

En la práctica se permiten ciertas tolerancias, Fig ??, y esto nos lleva a que además de las bandas libres y las bandas de rechazo van a haber bandas de transición.

De esta manera, el problema se reduce en encontrar una función  $f_a(x, \alpha_1, \dots, \alpha_n)$  que aproxime a otra función  $f(x)$  de un filtro ideal, en un intervalo  $x_1 \leq x \leq x_2$ , donde los parámetros  $\alpha_1, \dots, \alpha_n$ , en la función aproximación, son

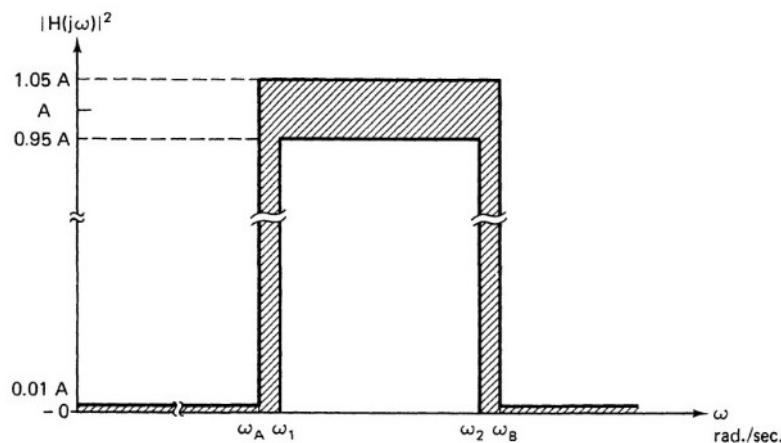


Figura 5.2: Especificaciones de un filtro.

encontrados por medio de algún criterio que contemple el error  $\epsilon$  que surge entre la diferencia entre ambas funciones

$$\epsilon = f(x) - f_a(x; \alpha_1, \dots, \alpha_n)$$

La selección de la función de aproximación se pueden dividir en los siguientes criterios.

1. Mínimos cuadrados. El valor de  $I(\alpha_1, \dots, \alpha_n)$  definido por:

$$I(\alpha_1, \dots, \alpha_n) = \int_{x_1}^{x_2} |\epsilon|^2 w(x) dx$$

es minimizado, donde  $w(x)$  es una función de peso, la cual atenúa el error en cierto subintervalos.

2. Máxima planicidad. Las primeras  $n - 1$  derivadas del error,  $\epsilon$ , se hacen cero en  $x = x_0$ .

3- Chebyshev. El valor de  $\mu$  (constante de riple) es minimizado en el intervalo  $x_1 \leq x \leq x_2$  donde  $\mu = |\epsilon|_{max}$ .

4. Interpolación. Se busca que el valor del error,  $\epsilon$ , se reduzca para un conjunto de  $n$  puntos en el intervalo  $x_1 \leq x \leq x_2$ .

Una vez que se ha tomado alguno de estos criterio se debe determinar la forma de la función aproximación, ya sea en el dominio del tiempo o en el dominio de la frecuencia.

#### a- Selección de la función de aproximación en el dominio del tiempo.

Consiste en aproximar una respuesta al impulso  $h(t)$  desde el dominio del tiempo con una función  $h^*(t)$  tal que el error sea mínimo.

$$\epsilon = \int_0^\infty (h(t) - h^*(t))^2 dt$$

La función de aproximación va a tener la forma:

$$f_a(x; \alpha_1, \dots, \alpha_n) = \alpha_1 e^{\alpha_2 x} + \alpha_3 e^{\alpha_4 x} + \dots$$

Un procedimiento efectivo es usar en el dominio del tiempo funciones ortogonales,  $\phi(t)$ , donde la función  $h^*(t)$  toma la forma

$$h^*(t) = \sum_{k=1}^n \alpha_k \phi_k(t)$$

queda el error expresado como

$$\epsilon = \int_0^\infty \left[ h(t) - \sum_{k=1}^n \alpha_k \phi_k(t) \right]^2 dt$$

y es minimizado cuando

$$\alpha_k = \int_0^{\infty} h(t)\phi_k(t)dt \quad k = 1, 2, \dots, n$$

Si se eligen funciones ortogonales como una suma de exponentiales  $e^{s_k t}$ , la respuesta al impulso será  $h^*(t) = \sum_{k=1}^n \alpha_k e^{s_k t}$  y finalmente, se tendrá la transformada  $H^*(s) = \sum_{k=1}^n \frac{\alpha_k}{s - s_k}$ .

b- Selección de la función de aproximación en el dominio del frecuencia.

La selección de la función de aproximación trata en que hay que encontrar una función racional  $H(s)$  que tenga alguna característica tal como máxima planicidad o igual riple. Estas características se pueden buscar en la función de magnitud, o en la función de fase. La forma de la función en general va ser

$$f_a(x; \alpha_1, \dots, \alpha_n) = \frac{\alpha_1 + \alpha_3 x + \alpha_5 x^2 + \dots}{\alpha_2 + \alpha_4 x + \alpha_6 x^2 + \dots}$$

El filtro ideal no es realizable porque la respuesta en el impulso no es cero para  $t < 0$ . Por ejemplo, en el caso de buscar la máxima planicidad, la función de aproximación va a ser una función racional, donde se asume que los ceros están en infinito (filtro pasa bajos) y la función de magnitud tendrá la forma:

$$M(\omega) = \frac{K_0}{[1 + f(\omega^2)]^{1/2}}$$

donde  $K_0$  es la constante de corriente DC ( $M(j0)$ ) y  $f(\omega^2)$  es el polinomio a ser seleccionado.

Por último, esta función elegida debe cumplir con los criterios de que sea realizable con elementos pasivos o elementos activos en el caso de filtros analógicos( o elementos computacionales para el caso de los filtros digitales).

## 5.2. Filtro de Butterworth

En este tipo de filtro el parámetro característico es que el orden es dependiente de las atenuaciones elegida para cada banda.

### Función de aproximación

La función de aproximación queda dada por

$$B_n(\omega) = \omega^n$$

### Respuesta en frecuencia

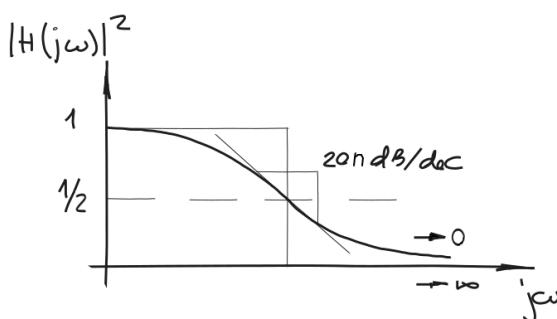


Figura 5.3: Respuesta en frecuencia del filtro de Butterworth

### Orden del filtro y atenuación

En base a las características enunciadas anteriormente se llega a que el orden queda dado en función de la atenuación requerida según la banda de interés:

$$n = \frac{\log_{10}(10^{A_{db}/10} - 1)}{2\log_{10}\omega}$$

### Desarrollo

$$\begin{aligned} |H(j\omega)|_{db}^2 &= 10\log_{10} \frac{1}{1 + \omega^{2n}} = -A_{db} \\ \omega^{2n} &= 10^{A_{db}/10} - 1 \\ 2n\log_{10}\omega &= \log_{10}(10^{A_{db}/10} - 1) \longrightarrow n = \frac{\log_{10}(10^{A_{db}/10} - 1)}{2\log_{10}\omega} \end{aligned}$$

### Función de transferencia y polos de la función de transferencia

La función de transferencia tiene la forma:

$$\begin{aligned} |H(s))|^2 &= \begin{cases} \frac{1}{1+s^{2n}}, & \text{si } n \text{ es par} \\ \frac{1}{1-s^{2n}}, & \text{si } n \text{ es impar} \end{cases} \\ H(s) &= \prod_{k=1}^n \frac{1}{s - s_k} = \begin{cases} \prod_{k=1}^{n/2} \frac{1}{s^2 + 2\sin\theta_k + 1}, & \text{si } n \text{ es par} \\ \frac{1}{s+1} \prod_{k=1}^{n/2} \frac{1}{s^2 + 2\sin\theta_k + 1}, & \text{si } n \text{ es impar} \end{cases} \end{aligned}$$

La expresión para determinar los polos queda dada por

$$\hat{s}_k = -\sin\left(\frac{2k-1}{2n}\pi\right)\pi + j\cos\left(\frac{2k-1}{2n}\pi\right)$$

## Teoría de los circuitos II

donde  $k$  y  $n$  es un número entero dado entre  $k = 1, 2, \dots, n$ .

### Desarrollo

Para el caso de que  $n$  sea par se tiene:

$$1 + s^{2n} = 0$$

$$s^{2n} = -1 = e^{j(\frac{\pi - 2k\pi}{2n})} \quad k=1, \dots, 2n$$

$$s_k = e^{j(\frac{2k-1}{2n}\pi)} \Rightarrow \theta_k = \frac{2k-1}{2n}\pi \Rightarrow \hat{s}_k = \cos\left(\frac{2k-1}{2n}\pi\right) + j \sin\left(\frac{2k-1}{2n}\pi\right)$$

$$\cos \theta = -\sin\left(\frac{\pi}{2}\right)$$

$$\left(\frac{2(k+\frac{n}{2})-1}{2n}\right) = \frac{2k+n-1}{2n} = \frac{(2k-1)\pi}{2n} - \frac{\pi}{2n} = \pi\left(\frac{2k-1}{2n}\right) - \frac{\pi}{2}$$

$$\Rightarrow \hat{s}_k = -\sin\left(\frac{2k-1}{2n}\pi\right) + j \cos\left(\frac{2k-1}{2n}\pi\right)$$

Y en el otro caso, donde  $n$  es impar el desarrollo consiste en:

$$1 - s^{2n} = 0$$

$$s^{2n} = 1 = e^{j(\frac{\pi(2k-2n)}{2n})} = e^{-j(\frac{(1-k)\pi}{n})} \quad k=1, \dots, (2n-1)$$

$$\Rightarrow \theta_k = \frac{(k-1)\pi}{n} \Rightarrow s_k = -\sin\left(\frac{2k-1}{2n}\pi\right) + j \cos\left(\frac{2k-1}{2n}\pi\right)$$

$$\cos \theta = -\sin\left(\frac{\pi}{2}\right)$$

$$\frac{k-1}{n}\pi = \frac{k+2n-1}{n}\pi = \frac{2k+n-1}{2n}\pi = \frac{2k-1}{2n}\pi + \frac{\pi}{2}$$

Finalmente, construyendo las expresiones correspondientes con cada polo y su conjugado tendrá la siguiente forma:

$$(s - s_k)(s - \bar{s}_k) = (s - (\alpha + j\beta))(s - (\alpha - j\beta)) = 0$$

$$s^2 - (\alpha + j\beta)s - (\alpha - j\beta)s + (\alpha^2 + \beta^2) = 0$$

$$s^2 - 2\alpha s + (\alpha^2 + \beta^2) = 0$$

$$\boxed{s^2 + 2 \sin(\theta_k)s + 1 = 0}$$

**Ejemplo 1 con Matlab:** Determinando los polos y los ceros para orden 3, Fig. ??.

En este ejemplo se construye la función de transferencia a partir de los polos conjugados.

```

1 clear all; clc; close all;
2
3 n=3 % orden del filtro
4
5 syms k integer;
6 theta(k)=(2*k-1)/(2*n) *pi;
7 m=floor(n/2);
8
9 syms s;
10 H(s)=1/prod( s^2 + 2*sin(theta(1:m))*s + 1 ) * 1/(mod(n,2)*s+1);
11 H(s)=collect(H(s));
12
13 [num, den]= numden(H(s));
14 B=sym2poly(num); A=sym2poly(den);
15
16 fig=figure;
17 zplane( [zeros(1, n-length(B)+1) B], [zeros(1, n-length(A)+1) A] );
18 grid on; title('Filtro Butterworth');
19 saveas(fig, './butterworth_pz_ej1', 'jpg');

```

**Ejemplo 2 con Matlab:** Determinando polos y ceros para orden 3, Fig. ??.

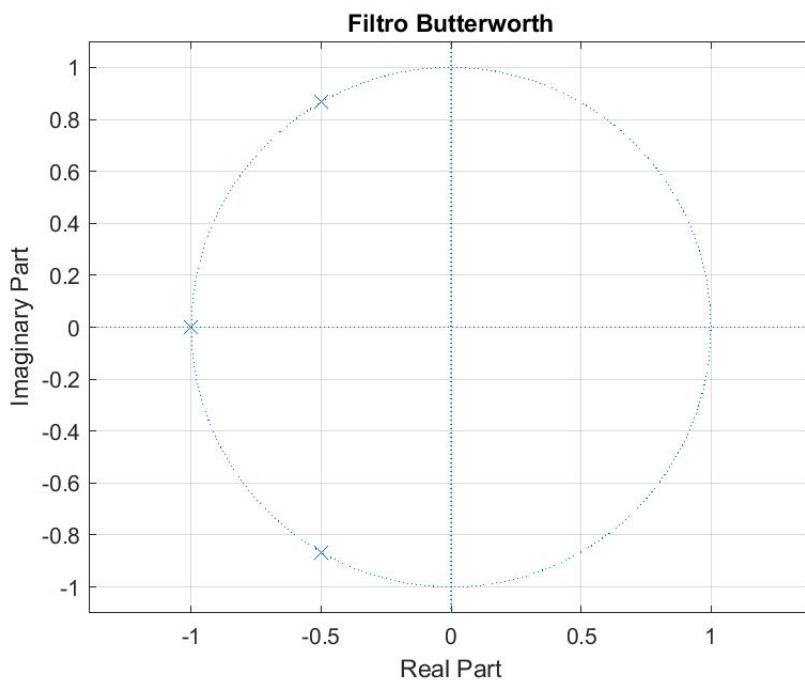


Figura 5.4: Diagrama de polos y ceros de la función de transferencia,  $H(s)$ , del filtro de Butterworth

A diferencia del ejemplo anterior, se determina la función de transferencia calculando los polos individuales en el semiplano-izquierdo y luego se hace el producto de cada expresión correspondiente.

```

1 clear all; clc; close all;
2
3 n=3 % orden del filtro
4
5 syms k integer;
6 theta(k)=(2*k-1)/(2*n)*pi;
7
8 sk(k)=-sin(theta(k)) + j*cos(theta(k));
9
10 syms s;
11 H(s)=collect( prod( 1./(s-sk(1:n)) ) );
12
13 [num, den]= numden(1/H(0)*H(s));
14 B=sym2poly(num); A=sym2poly(den);
15
16 fig=figure;
17 zplane( [zeros(1, n-length(B)+1) B], [zeros(1, n-length(A)+1) A] );
18 grid on; title('Filtro Butterworth');
19 saveas(fig,'./butterworth_pz_ej2','jpg');
```

### 5.3. Filtro de Chebyshev

La característica principal es la de tener una buena aproximación uniforme en toda la banda de paso. Los parámetros característicos son la constante de riple y el orden, que dependen de los requerimientos de atenuación. Otra característica importante es que la frecuencia de corte es mayor a uno (cuando se tiene la frecuencia normalizada).

#### Función de aproximación:

Queda definida por

$$C(\omega) = \begin{cases} \cos(n \cdot \cos^{-1}(\omega)) & , \text{ si } |\omega| \leq 1 \\ \cosh(n \cdot \cosh^{-1}(\omega)) & , \text{ si } |\omega| > 1 \end{cases}$$

y para el caso de que la expresión dada se defina en forma recursiva sería:

$$C(\omega) = \begin{cases} 1 & , \text{ si } n = 0 \\ \omega & , \text{ si } n = 1 \\ 2\omega C_{n-1} - C_{n-2}(\omega) & , \text{ para otro valor de } n \end{cases}$$

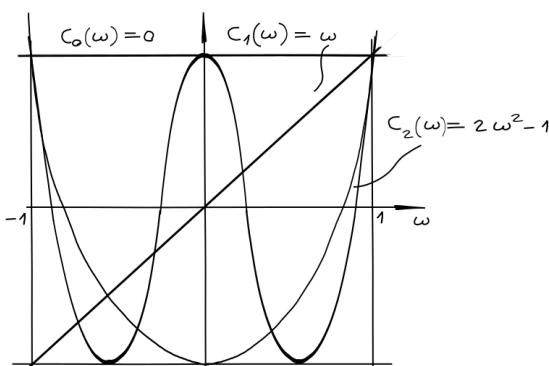


Figura 5.5: Polinomios de Chebyshev

Resumen de propiedades:

1- Los valores característicos son:

$$C_n(0) = \begin{cases} 0 & , \text{ si } n \text{ es impar} \\ 1 & , \text{ si } n \text{ es par} \end{cases}$$

2- Los ceros están en el intervalo  $|\omega| < 1$ .

3- Dentro del intervalo,  $|\omega| \leq 1$ , no excede la unidad.

4- El valor absoluto de la función se incrementa rápidamente para  $|\omega| > 1$ .

#### Función de respuesta en frecuencia

$$|H(j\omega)|^2 = \frac{1}{1 + \epsilon^2 C_n^2(\omega)}$$

donde  $\epsilon$  es la constante de riple, y  $C_n(\omega)$  es la función de Chebyshev de orden  $n$ . En el intervalo  $|\omega| \leq 1$  oscilará desde un máximo, en 1, a un mínimo, en  $\frac{1}{1+\epsilon^2}$ , y fuera de este intervalo se aproxima a cero rápidamente.

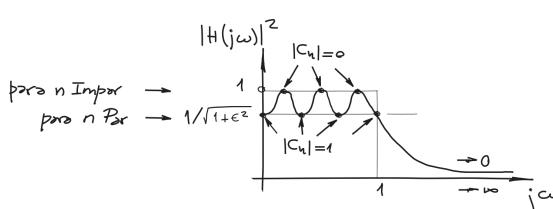


Figura 5.6: Función respuesta en frecuencia del filtro de Chebyshev

Resumen de Propiedades:

1- Los valores característicos son:

$$|H(j1)|^2 = \frac{1}{1 + \epsilon^2} \text{ para todo } n$$

$$|H(j0)|^2 = \begin{cases} 1 & \text{para } n \text{ impar} \\ \frac{1}{1+\epsilon^2} & \text{para } n \text{ par} \end{cases}$$

2- Dentro de la banda de paso,  $0 \leq \omega \leq 1$ ,  $H(j\omega)$  oscilará entre 1 y  $\frac{1}{\sqrt{1+\epsilon^2}}$ , entonces la amplitud del riple queda definida por:

$$\text{Ripple} = 1 - \frac{1}{\sqrt{1 + \epsilon^2}}$$

### Orden del filtro, constante de riple y atenuación:

La atenuación esta dada por:

$$\alpha_n = -10 \log_{10} |H_n(j\omega)|^2 = 10 \log |1 + \epsilon^2 C_n^2(\omega)| \text{ dB}$$

En la banda de rechazo,  $\omega > \omega_c$ , siendo  $\omega_c$  la frecuencia de corte, la función de aproximación de Chebyshev es  $C_n(\omega) = \cosh(n \cdot \cosh^{-1}\omega)$  y por lo tanto la atenuación en esta banda es:

$$\alpha_n = 10 \log |1 + \epsilon^2 \cosh^2(n \cdot \cosh^{-1}\omega)| \text{ dB} \quad (5.1)$$

donde es dependiente de la constante de riple  $\epsilon$  y del orden  $n$ .

Además, el orden  $n_s$  requerido para la mínima atenuación deseada en la banda de rechazo se despeja evaluando la ecuación ?? en  $\omega_s$ :

$$n_s = \frac{\cosh^{-1} \left( \frac{\sqrt{10^{A_{dB}/10} - 1}}{\epsilon} \right)}{\cosh^{-1}(\omega_s)}$$

De igual forma, el orden  $n_{3db}$  requerido en la frecuencia de corte (con atenuación 3dB),  $\omega_c$ , es:

$$n_{3dB} = \frac{\cosh^{-1} \left( \frac{\sqrt{10^{3dB/10} - 1}}{\epsilon} \right)}{\cosh^{-1}(\omega_c)}$$

El orden del filtro se estima como el máximo orden que cumpla los requerimientos de atenuación dados.

$$n = \max(n_s, n_{3dB})$$

El riple se despeja de la atenuación máxima deseable en la banda de paso, en  $H(j1)$ , y es:

$$\alpha_{max} = 10 \log_{10}(1 + \epsilon^2)$$

quedando que el riple definido por:

$$\epsilon = \sqrt{10^{\alpha_{max}/10} - 1}$$

Desarrollo:

$$10 \log \left| \frac{1}{1 + \epsilon^2 \cosh^2(n \cosh^{-1}\omega)} \right| = - A_{dB} \quad \cancel{\cosh^{-1}(n \cosh^{-1}\omega_s)} = \sqrt{\frac{A_{dB}/10}{\epsilon^2} - 1}$$

$$n = \frac{\cosh^{-1} \left[ \sqrt{\frac{A_{dB}/10}{\epsilon^2} - 1} \right]}{\cosh^{-1}\omega_s}$$

### Función de transferencia a partir de los polos

La función de transferencia va a ser una función racional de variable compleja  $s$  y va a tener la forma

$$H(s) = \prod_{k=1}^n \frac{1}{s - s_k}$$

donde  $s_k$  son polos ubicados en el semiplano izquierdo,  $k$  son valores enteros definidos desde  $k = 1, \dots, n$ , donde  $n$  es el orden del filtro y  $s_k = (\sigma'_k + j\omega'_k) \cosh(\beta_k)$  evaluada en  $k = 1, 2, \dots, n$  da las raíces del denominador, para la cual se tiene que:

$$\sigma'_k = \tanh(\beta_k) \sin\left(\frac{2k-1}{2n}\pi\right), \omega'_k = \cos\left(\frac{2k-1}{2n}\pi\right), \beta_k = \frac{\sinh^{-1}(1/\epsilon)}{n}$$

Geométricamente hablando, los polos en el filtro Chebyshev se ubicaran sobre una elipse en el plano s, con el eje mayor sobre el eje imaginario, Fig.???. Así que cuanto más delgada sea la elipse más influencia tendrán los polos sobre el eje imaginario, y mayor ripple.

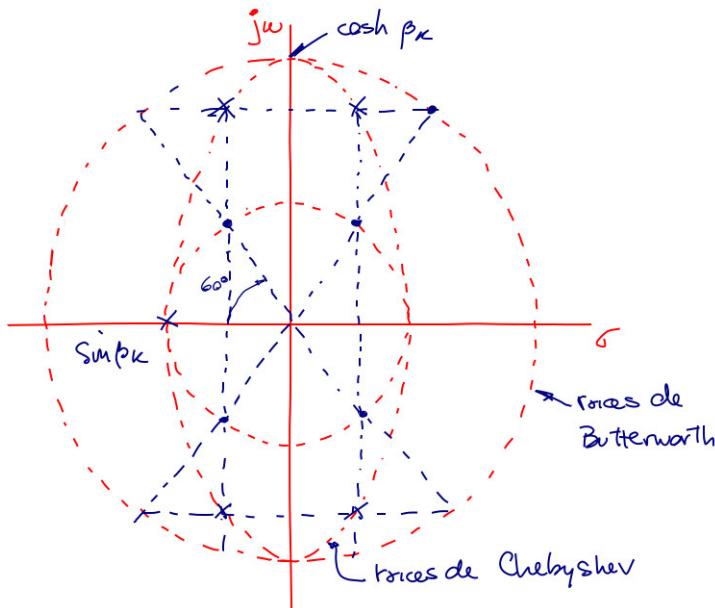


Figura 5.7: Ubicación de los polos en el plano imaginario.

#### Desarrollo del cálculo de las raíces del denominador:

Para encontrar los polos se buscan los ceros del denominador, así que primero hay que desarrollar el denominador como sigue:

$$\begin{aligned}
 1 + [\epsilon C_n(\omega)]^2 &= 0 \\
 [1 + j \epsilon C_n(\omega)][1 - j \epsilon C_n(\omega)] &= 0 \Rightarrow C_n(\omega) = \pm \frac{1}{\epsilon} j \\
 \cos n \underbrace{\cos^{-1}\left(\frac{\omega}{\epsilon}\right)}_{\alpha_k + j \beta_k} &= \pm \frac{1}{\epsilon} j \\
 \cos(n\alpha_k + j n \beta_k) &= \underbrace{\sin n \alpha_k \cdot \cosh n \beta_k}_{=0} - j \underbrace{\cos n \alpha_k \cdot \sinh n \beta_k}_{\pm 1/\epsilon} = \pm \frac{1}{\epsilon} j
 \end{aligned}$$

encontrando como soluciones a

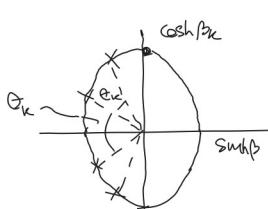
$$\begin{aligned}
 \bullet \quad &\cosh n \alpha_k \cdot \cosh n \beta_k = 0 \Rightarrow \alpha_k = \frac{\pi(2k-1)}{2n} \Rightarrow \alpha_k = \frac{\pi(2k-1)}{2n} = \frac{\pi}{2n}, \frac{3}{4n}\pi, \dots \\
 \bullet \quad &\sin n \alpha_k \cdot \sinh n \beta_k = \pm \frac{1}{\epsilon} \\
 &\Rightarrow \beta_k = \frac{\sinh^{-1}\left(\frac{(-1)^k}{\epsilon}\right)}{n} \\
 &k = 1, 2, 3, \dots
 \end{aligned}$$

## Teoría de los circuitos II

Despejando  $s_k$  se tiene:

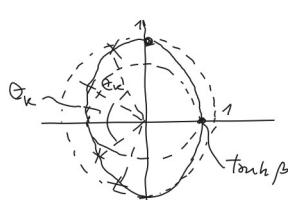
$$\begin{aligned} \cos^{-1} \frac{s_k}{\beta} = \alpha_k + j\beta_k \Rightarrow s_k &= j \cos(\alpha_k + j\beta_k) \\ s_k &= j [\cos \alpha_k \cosh \beta_k - j \sin \alpha_k \sinh \beta_k] \\ s_k &= \frac{\sin \alpha_k \sinh \beta_k}{\omega_n} + j \frac{\cos \alpha_k \cosh \beta_k}{\omega_n}, \end{aligned}$$

y por identidad trigonométrica, la ubicación geométrica de los polos queda dada en una elipse, tal como se muestra en Fig.??



$$\begin{aligned} \sigma_k &= \sin\left(\frac{i(2k-1)\pi}{2n}\right) \cdot \underbrace{\sinh\left[\frac{\sinh^{-1}\left(\frac{(\pm 1)^k}{e}\right)}{n}\right]}_A = |A| \sin \theta_k \quad \Rightarrow \quad \frac{\sigma^2}{|\sinh \beta_k|^2} + \frac{\omega^2}{|\cosh \beta_k|^2} = 1 \\ \omega_k &= \cos\left(\frac{i(2k-1)\pi}{2n}\right) \cdot \underbrace{\cosh\left[\frac{\sinh^{-1}\left(\frac{(\pm 1)^k}{e}\right)}{n}\right]}_B = |B| \sin \theta_k \end{aligned}$$

Para compararlo con la geometría de una circunferencia de radio igual a uno, se normalizan las raíces tal como sigue:



$$\begin{aligned} s'_k &= \frac{s_k}{\cosh \beta_k} = \frac{\sigma_k}{\cosh \beta_k} + j \frac{\omega_k}{\cosh \beta_k} = \sigma'_k + j \omega'_k \\ \sigma'_k &= \frac{\sinh \beta_k}{\cosh \beta_k} \cdot \sin\left(\frac{2k-1\pi}{2n}\right) \Rightarrow \boxed{\sigma'_k = \tanh \beta_k \cdot \sin\left(\frac{2k-1\pi}{2n}\right)} \\ \omega'_k &= \frac{\cosh \beta_k}{\cosh \beta_k} \cdot \cos\left(\frac{2k-1\pi}{2n}\right) \Rightarrow \boxed{\omega'_k = \cos\left(\frac{2k-1\pi}{2n}\right)} \\ \beta'_k &= \frac{\sinh^{-1}\left(\frac{(\pm 1)^k}{e}\right)}{n} \quad \boxed{\beta'_k} \\ s_k &= (\sigma'_k + j \omega'_k) \cosh \beta_k \quad \boxed{s_k} \end{aligned}$$

**Ejemplo con Matlab 1:** Determinando los polos y ceros para orden 3, Fig. ??.

```

1 clear all; clc; close all;
2
3 n=3           % orden del filtro
4 e=0.2;        % constante del riple
5
6 syms k integer;
7 theta(k)=(2*k-1)/ (2*n) *pi;
8 beta(k)=asinh((1)^k/e)/(n);
9
10 sk(k)=tanh(beta(k)).*(-1).*sin(theta(k)) + j*cos(theta(k));
11
12 syms s;
13 H(s)=collect(prod(1./ (s-sk(1:n))));
14
15 [num, den]= numden(1/H(0)*H(s));
16 B=sym2poly(num); A=sym2poly(den);
17
18 fig=figure;
19 zplane([zeros(1, n-length(B)+1) B], [zeros(1, n-length(A)+1) A]);
20 grid on; title('Filtro de Chebyshev');
21 saveas(fig, './chebyshev_ej1_pz', 'jpg');

```

En el script se observa que una vez que se tienen los polos, según el orden y constantes de riple dados, se construye muy fácilmente la respuesta en frecuencia del filtro.

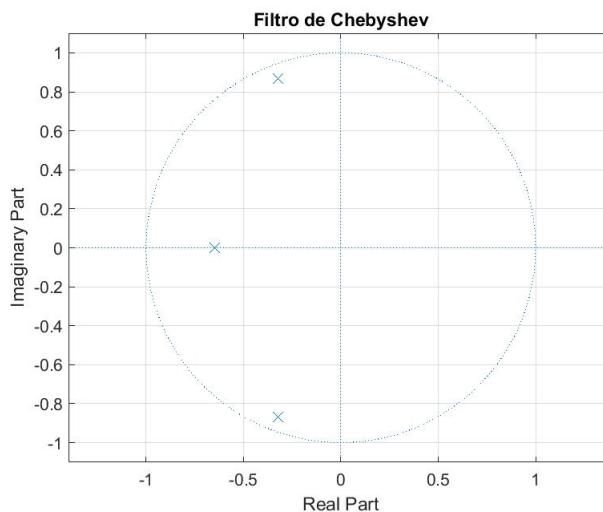


Figura 5.8: Diagrama de polos y ceros del filtro de Chebyshev del ejemplo 1

**Ejemplo con Matlab 2:** Filtro pasa bajo de Chebyshev a partir de las especificaciones dadas, Fig. ??.

Primero, se define una función que determine el orden y la constante de riple requerido para las características prefijadas del filtro a diseñar.

```

1 function [N,e]=mi_cheb1ord(Wc, Ws, Ap, As)
2
3 e=sqrt(10^(Ap/10)-1);
4 ns=acosh( sqrt(10^(As/10)-1)/e )/acosh(Ws);
5 A3db=3;
6 n3db=acosh( sqrt(10^(A3db/10)-1)/e )/acosh(Wc);
7 n=ceil(n3db);
8 if ns>n
9     n=ceil(ns);
10 end
11 N=n;
12

```

El procedimiento para el diseño del filtro es similar al realizado en el Ejemplo 1, la diferencia radica en que se determina el orden y la constante de riple a partir de los requerimientos propuestos para este ejemplo.

```

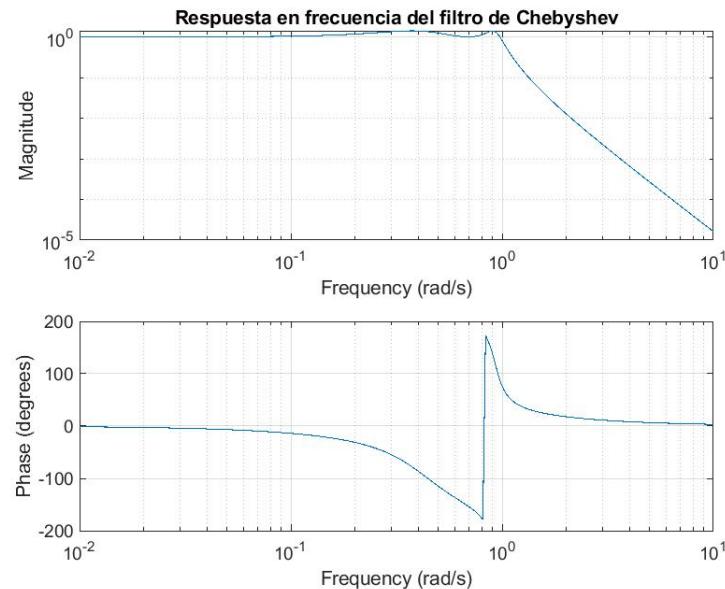
1 clear all; clc; close all;
2
3 % Especificaciones del filtro normalizadas
4 Wc=1.2;
5 Ws=1.3;
6 Ap=3; % atenuacion en dB
7 As=20; % atenuacion de dB
8
9 %% Calculo del orden y la constante de riple
10 [n,e] = mi_cheb1ord(Wc, Ws, Ap, As)
11
12 % Definicion de la expresion los polos
13 syms k integer;
14 theta(k)=(2*k-1) / (2*n) *pi;
15 beta(k)=asinh((1)^k/e)/(n);
16
17 sk(k)=tanh(beta(k)).*(-1).*sin(theta(k)) + j*cos(theta(k));
18
19 % Definicion de la funcion de transferencia a partir de los polos
20 syms s;
21 H(s)=collect( prod( 1./ (s-sk(1:n)) ) );
22
23 % respuesta en frecuencia.
24 fig1=figure;
25 [num, den]= numden(1/H(0)*H(s));

```

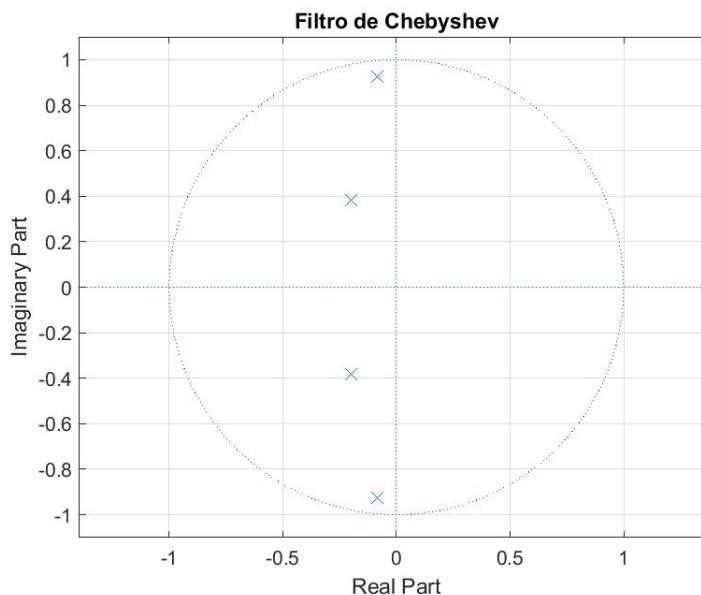
```

26 B=sym2poly(num); A=sym2poly(den);
27 freqs(B,A);
28 title('Respuesta en frecuencia del filtro de Chebyshev');
29 saveas(fig1,'./chebyshev_ej2_freqs','jpg');
30
31 % polos y ceros
32 fig2=figure;
33 zplane( [zeros(1, n- length(B)+1) B ], [zeros(1, n- length(A)+1) A] );
34 grid on; title('Filtro de Chebyshev');
35 saveas(fig2,'./chebyshev_ej2_pz','jpg');

```



(a) Respuesta en frecuencia del filtro Chebyshev del Ejemplo 2



(b) Diagrama de polos y ceros del filtro de Chebyshev del Ejemplo 2

Figura 5.9: Ejemplo 2

## 5.4. Filtro de Bessel

Este tipo de filtro consiste en relacionar las especificaciones de retardo de fase lineal o de grupo constante del filtro ideal con los polinomios de la función de transferencia.

El retardo de grupo es de particular importancia cuando es relevante las formas de onda de la señal. Por ejemplo para señales de audio, no es perceptible los cambio de fase, pero para señales de video es importante conservar la forma de onda. Las funciones obtenidas de esta manera tienen un retraso plano máximo en el origen.

La función de Bessel aproxima a una constante de retardo de un filtro pasa bajo.

### Función de aproximación:

El polinomio de Bessel definido en forma cerrada es:

$$B_n(\tau_0 s) = s^n + b_{n-1}s^{n-1} + \dots + b_1s + b_0$$

donde es  $b_k = \frac{(2n-k)!}{2^{n-k}k!(n-k)!}$  y n es el orden del polinomio.

Por otro lado, la forma recursiva esta dada por:

$$B_n(s) = \begin{cases} 1 & , \text{ para } n = 0 \\ s + 1 & , \text{ para } n = 1 \\ (2n - 1)B_{n-1}(s) - s^2B_{n-2}(s) & , \text{ otro valor} \end{cases}$$

### Función de transferencia

$$H(s) = \frac{K}{e^{\tau_0 s}} = \frac{K}{s^n + b_{n-1}s^{n-1} + \dots + b_1s + b_0}$$

donde K se elige tal que  $H(0) = 1$  y por lo tanto  $K = b_0$ ;  $\tau_0$  es la constante de fase y los coeficientes del polinomio se pueden obtener de dos formas:

- 1- Igualación de parámetros, o
- 2- Desarrollo de polinomios de Bessel.

#### 1- Método por igualación de parámetros

Planteando la función de transferencia en s tal como:

$$H(s) = \frac{K}{e^{\tau_0 s}} = \frac{K}{\cosh(s) + \sinh(s)} = \frac{K}{s^n + b_{n-1}s^{n-1} + \dots + b_1s + b_0} = \frac{K}{M(s) + N(s)} = K \left( \frac{M(s) - N(s)}{M^2(s) - N^2(s)} \right)$$

donde el par transformado de  $H(s)$  viene dado por

$$h(t) = K\delta(t - \tau_0) \xleftrightarrow{\mathcal{L}} H(s) = Ke^{-\tau_0 t}$$

así  $H(s)$  se relaciona con la función de transferencia a partir de los polinomios en s, siendo  $s = j\omega$ , la fase queda definida por

$$\varphi(\omega) = -\operatorname{atanh} \left( \frac{N(\omega)/j}{M(\omega)} \right) = -\operatorname{atan} \left( \frac{b_1\omega}{-\omega^2 + b_0} \right) \quad (5.2)$$

El retardo de grupo,  $\tau(\omega)$ , es la derivada de la fase  $\varphi(\omega)$

$$\tau(\omega) \triangleq -\frac{d\varphi(\omega)}{d\omega} = -\frac{u'}{1+u^2} = T \text{ siendo } u = \frac{N(\omega)/j}{M(\omega)} \text{ y } T \text{ es una constante.}$$

desarrollando lo anterior para una retardo de grupo normalizado, con  $T = 1$ ,

$$\tau(\omega) = -\frac{-\operatorname{atan} \left( -\frac{b_1\omega}{-\omega^2 + b_0} \right)}{d\omega} = \frac{b_1\omega^2 + b_1b_0}{\omega^4 + (-2b_0 + b_1^2)\omega^2 + b_0^2} = 1 \quad (5.3)$$

## Teoría de los circuitos II

---

Entonces, la máxima planicidad de retardo de grupo implica que  $\tau(0) = 1$ , de la Ec. ?? se despeja

$$\frac{b_1 b_0}{b_0^2} = 1 \rightarrow b_1 = b_0 \quad (5.4)$$

Operando la Ec. ??

$$b_1 \omega_1^2 + b_1 b_0 = \omega^4 + (-2b_0 + b_1^2)\omega^2 + b_0^2$$

y de esta ultima expresión reemplazando con lo obtenido en la Ec. ?? se tiene

$$\omega^4 + (-3b_0 + b_0^2)\omega^2 = 0$$

el segundo término debe ser cero, entonces

$$(-3b_0 + b_0^2) = 0 \rightarrow b_0 = 3 \quad (5.5)$$

por lo tanto,  $b_1 = 3$ , la función de transferencia es

$$H(s) = \frac{b_3}{s^3 + 3s + 3} \quad (5.6)$$

### 2- Desarrollo en polinomios de Bessel

Partiendo de que

$$H(s) = \frac{K}{e^{\tau_0 s}} = \frac{K}{\cosh(s) + \sinh(s)}$$

la fase se podría plantear mediante el cociente de  $\cosh(s)$  y de  $\sinh(s)$ , y luego con los desarrollos de serie de Taylor se puede llegar a

$$\varphi_2(s) = \frac{\cosh(s)}{\sinh(s)} = \frac{1 - \frac{s^2}{2!} + \frac{s^4}{4!} - \frac{s^6}{6!} + \dots}{1 - \frac{s^3}{3!} + \frac{s^5}{5!} - \frac{s^7}{7!} + \dots} = \frac{1}{s} + \frac{1}{\frac{3}{s} + \frac{1}{\frac{5}{s} + \frac{1}{\frac{7}{s} + \dots}}} \quad (5.7)$$

el desarrollo de la división continuada queda como sigue:

$$\begin{aligned} & \frac{\frac{1}{s}}{s + \frac{s^3}{3!} + \frac{s^5}{5!} + \frac{s^7}{7!} + \dots} = \frac{\frac{1}{s}}{1 + \frac{s^2}{2!} + \frac{s^4}{4!} - \frac{s^6}{6!} + \dots} \\ & \frac{\frac{1}{s}}{1 + \frac{s^2}{2!} + \frac{s^4}{4!} - \frac{s^6}{6!} + \dots} = \frac{\frac{1}{s}}{1 + \frac{s^2}{3!} + \frac{s^4}{5!} - \frac{s^6}{7!} + \dots} \\ & \frac{\frac{1}{s}}{1 + \frac{s^2}{3!} + \frac{s^4}{5!} - \frac{s^6}{7!} + \dots} = \frac{\frac{1}{s}}{1 + \frac{s^2}{3!} + \frac{s^4}{5!} - \frac{s^6}{7!} + \dots} \\ & \frac{\frac{1}{s}}{1 + \frac{s^2}{3!} + \frac{s^4}{5!} - \frac{s^6}{7!} + \dots} = \frac{\frac{1}{s}}{1 + \frac{s^2}{3!} + \frac{s^4}{5!} - \frac{s^6}{7!} + \dots} \\ & \frac{\frac{1}{s}}{1 + \frac{s^2}{3!} + \frac{s^4}{5!} - \frac{s^6}{7!} + \dots} = \frac{\frac{1}{s}}{1 + \frac{s^2}{3!} + \frac{s^4}{5!} - \frac{s^6}{7!} + \dots} \\ & \frac{\frac{1}{s}}{1 + \frac{s^2}{3!} + \frac{s^4}{5!} - \frac{s^6}{7!} + \dots} = \frac{\frac{1}{s}}{1 + \frac{s^2}{3!} + \frac{s^4}{5!} - \frac{s^6}{7!} + \dots} \end{aligned}$$

Para el caso de un orden 2, de la Ec. ?? se puede asociar el numerador de la fase a la parte par del denominador del polinomio de la función de transferencia, y de igual forma para con el denominador de la fase, quedando que

$$\varphi_2(s) = \frac{1}{s} + \frac{1}{\frac{1}{s}} = \frac{3 + s^2}{3s} = \frac{M(s)}{N(s)}$$

Entonces la función de transferencia queda como;

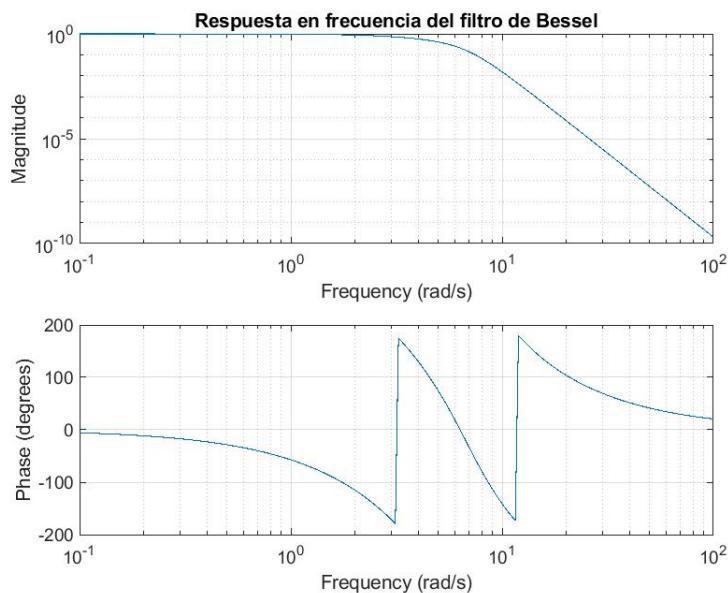
$$H(s) = \frac{K}{M(s) + N(s)} = \frac{K}{1 + s^2 + 3s} \text{ donde } k = 3 \text{ para que } H(0) = 1$$

Ejemplo con Matlab 1: Filtro pasa bajo de Bessel, Fig. ??.

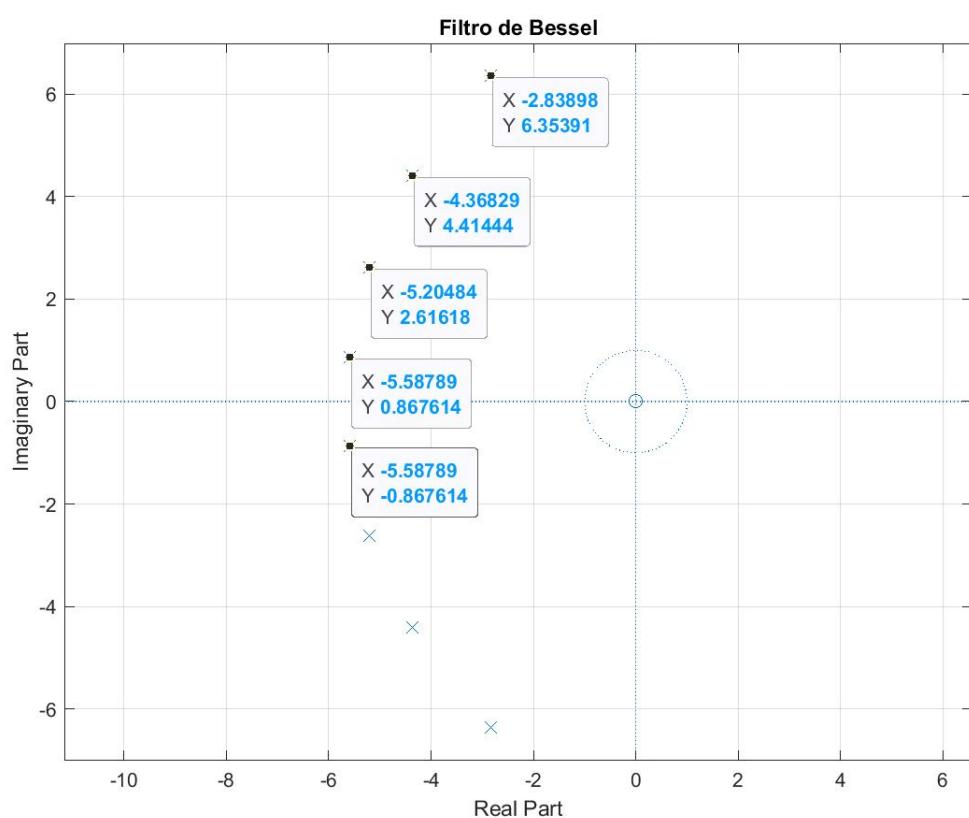
```

1 clear all; clc; close all;
2
3 % coeficientes del polinomio de Bessel
4 syms k n integer;
5 b(k,n) = (factorial(2*n-k)) ./ ...
6             ( 2.^(n-k).*factorial(k) .*factorial(n-k) );
7
8 % especificaciones del filtro
9 N=8; % orden del filtro
10
11 % funcion de transferencia
12 syms s;
13 m=N:-1:0;
14 bm=b(m,N);
15 T(s)=bm*s.^m; % polinomio de Bessel
16
17 H(s)=collect( bm(end)/T(s) ) % funcion de transferencia
18
19 [num, den]= numden(H(s));
20 B=sym2poly(num); A=sym2poly(den);
21
22 % respuesta en frecuencia.
23 fig1=figure;
24 freqs(B,A);
25 title('Respuesta en frecuencia del filtro de Bessel');
26 saveas(fig1,'./bessel_ej1_freqs','jpg');
27
28 % polos y ceros
29 fig2=figure;
30 zplane( [zeros(1, N- length(B)) B], [zeros(1, N- length(A)) A] );
31 grid on; title('Filtro de Bessel');
32 saveas(fig2,'./bessel_ej1_pz','jpg');
33

```



(a) Respuesta en frecuencia del filtro Bessel del Ejemplo 1



(b) Diagrama de polos y ceros del filtro de Bessel del Ejemplo 1

Figura 5.10: Ejemplo 1. Filtro pasa bajo de Bessel

## 5.5. Comentarios sobre la bibliografía

Lam Harry. Analog and digital filters design and realization.

*Desarrollo general.*

Pactitis S. A. Active Filters: Theory and design

*Desarrollo del Filtro de Chebyshev.*

Weinberg. Network analysis and synthesis

*En la pág. 365, tiene mejor desarrollado la introducción de funciones de aproximación.*

*Las expresiones para calcular los polos del filtro de Chebyshev.*

Kendal Su. Analog Filters.

*En la pág.66, el desarrollo por igualación de parámetros de Bessel.*

*En la pág.69, la expresión cerrada de Bessel.*

Van Valkenburg, Mac Elwyn. Analog filter design.

*En la pág 294. Función de transferencia y retardo en el tiempo*

Arthur Williams, Fred J. Taylor. Electronic Filter Design Handbook. Fourth Edition.

*En la página 52, menciona sobre la ubicación geométrica de los polos.*

Van Valkenburg, Mac Elwyn. Introduction to modern network synthesis.

*En la pág 391, Fig. 13-17 Comparación de la ubicación geométrica de los polos entre filtros de máxima planicidad de retardo y de magnitud*

Daryanani G. Principles of active network synthesis and design.

*Ejemplo de filtros de Bessel*

# 6 Transformación de frecuencia

## 6.1. Introducción

Un procedimiento para el diseño de filtros en otras estructuras diferentes a un filtro pasa bajo normalizado se consigue al trasladar los requerimientos de algunas estructuras de filtros no normalizadas a un filtro pasa bajo normalizado, para luego aplicar el procedimiento de diseño de un filtro pasa bajo normalizado, y finalmente, por medio de alguna función de transformación, se transforma la función de transferencia del filtro obtenido a la función de transferencia del filtro no normalizado de interés. Este procedimiento se resumen en la Fig.??

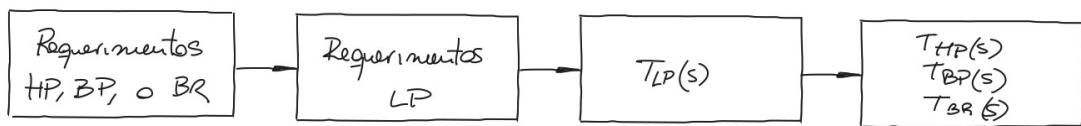


Figura 6.1: Procedimiento de diseño de un filtro no normalizado.

## 6.2. Transformación de paso bajo a paso alto

### Función de transformación

La función de transformación a un filtro pasa alto (HP) no normalizado,  $T_{HP}(s)$ , hacia un filtro pasa bajo normalizado (LP),  $T_{LP}(s)$ , es:

$$S = \frac{\omega_p}{s} \quad (6.1)$$

o bien, siendo  $S = j\Omega$  y  $s = j\omega$ , con la siguiente función de transformación

$$\Omega = \frac{-\omega_p}{\omega} \quad (6.2)$$

### Procedimiento de resolución de filtros paso alto.

1- Al conjunto de requerimientos dados de un filtro pasa alto:

$$(A_{max}, \omega_p, A_{min}, \omega_s)$$

donde,

$A_{max}$ , Atenuación máxima dada en la frecuencia en la banda de paso,  $\omega_p$

$\omega_p$ , Frecuencia dada en la banda de paso.

$A_{min}$ , Atenuación mínima dada en la frecuencia en la banda de rechazo,  $\omega_s$ .

$\omega_s$ , Frecuencia dada en la banda de rechazo.

Estos requerimientos se transforman con la expresión dada en Ec. ?? a los requerimientos equivalentes del filtro paso bajo normalizado

$$(A_{max}, 1, A_{min}, \Omega_s = \omega_p/\omega_s)$$

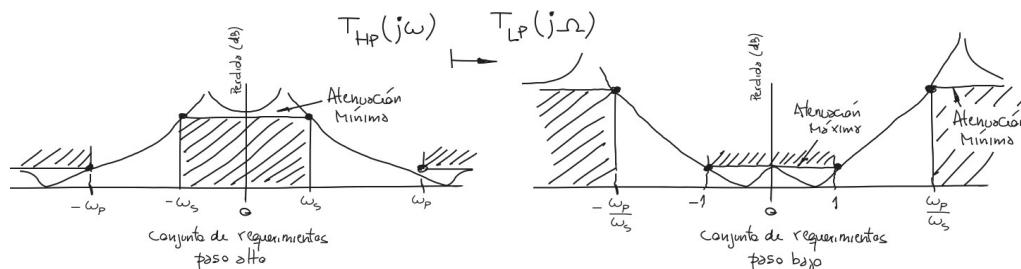


Figura 6.2: Funciones de atenuación. A la izquierda el filtro pasa alto. A la derecha el filtro pasa bajo

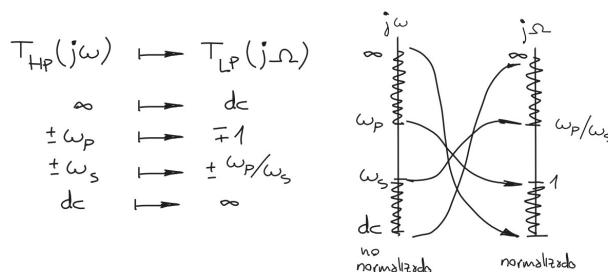


Figura 6.3: Relaciones entre las frecuencias de un filtro pasa alto y un filtro pasa bajo.

2- Seguido, se determina la función que aproxime los requerimientos pasa bajos, según la aplicación requerida por el filtro ( Butterworth, Chebyshev o Bessel ).

3- Por último, la función  $T_{LP}(S)$  se transforma a la función  $T_{HP}(s)$  con la Ec. Ec. ??, quedando.

$$T_{HP}(s) = T_{LP}(S)|_{S=\omega_p/s}$$

**Ejemplo con Matlab:** Creando una función de Matlab que permita realizar la transformación de paso alto a paso bajo.

```
1 function [B,A]=mi_lp2hp(Bs,As, wc)
2 syms s;
3 H(s)=poly2sym(Bs,s)/poly2sym(As,s);
4 % funcion de transformacion f(z)=wc/s
5 H(s)=collect(H(wc/s));
6
7 [num,den]=numden(H(s));
8 B=sym2poly(num);
9 A=sym2poly(den);
10
11
12 end
```

### 6.3. Transformación de paso bajo a pasa banda

#### Función de transformación

La función de transformación de un filtro pasa banda (BP) no normalizado,  $T_{BP}(s)$ , hacia un filtro pasa bajo normalizado (LP),  $T_{LP}(S)$ , es:

$$S = \frac{s^2 + \omega_0^2}{B \cdot s} \quad (6.3)$$

o bien

$$\Omega = \frac{\omega^2 - \omega_0^2}{B \cdot \omega} \quad (6.4)$$

donde,

$B = |\omega_x - \omega_{-x}|$ , es el ancho de banda de paso del filtro.

$\omega_0^2 = \omega_1 \cdot \omega_2 = \omega_3 \cdot \omega_4$ , es la frecuencia central (media geométrica) del filtro pasa banda.

#### Procedimiento de resolución de filtros pasa banda

1-Al conjunto de requerimientos de un filtro pasa banda:

$$(A_{max}, \omega_1, \omega_2, A_{min}, \omega_3, \omega_4)$$

se transforman con la función dada en la Ec. ?? a los requerimientos equivalentes del filtro pasa bajo normalizado (LP), quedando que

$$(A_{max}, \Omega_P = 1, A_{min}, \Omega_S = \frac{\omega_2 - \omega_1}{\omega_4 - \omega_3})$$

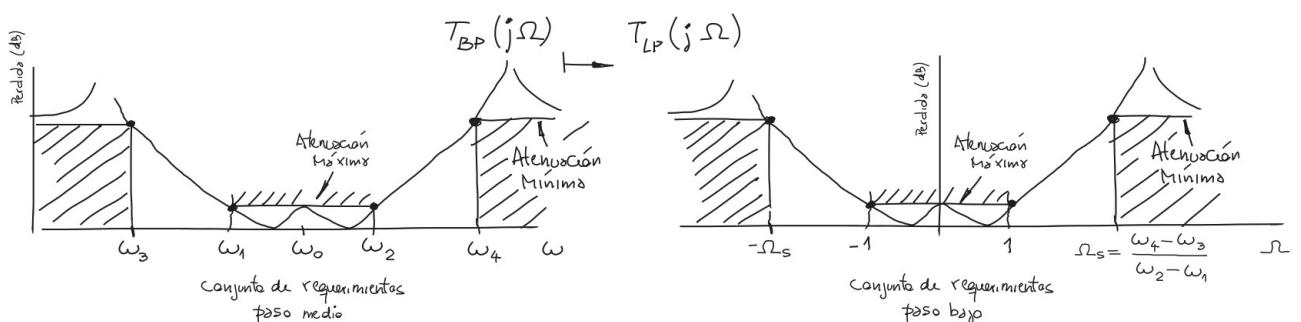


Figura 6.4: Funciones de atenuación. A la izquierda el filtro pasa banda. A la derecha el filtro pasa bajo normalizado

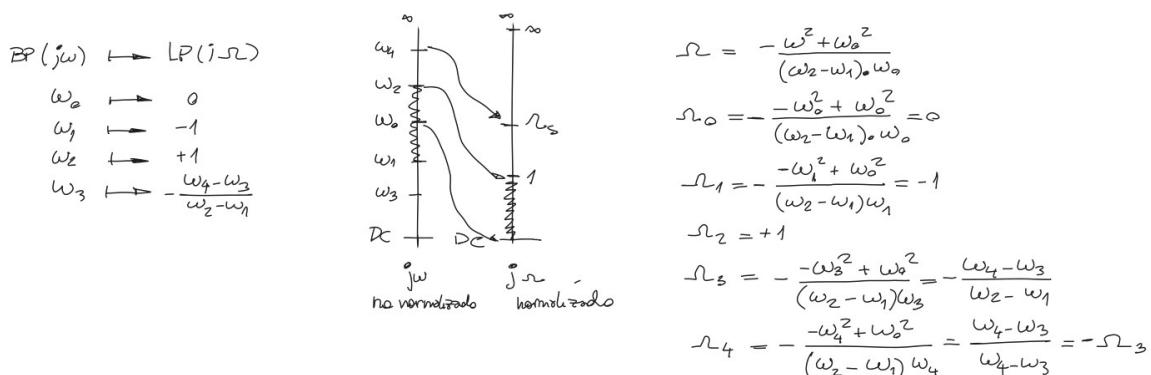


Figura 6.5: Relaciones entre las frecuencias de un filtro pasa banda y un filtro pasa bajo normalizado

## Teoría de los circuitos II

2- Seguido, se determina la función que aproxime los requerimientos pasa bajos según la aplicación (Butterworth, Chebyshev o Bessel).

3- Por último, la función  $T_{LP}(S)$  se transforma a la función  $T_{BP}(s)$  talque

$$T_{BP}(s) = T_{LP}(S)|_{S=(s^2+\omega_0^2)/B}$$

**Ejemplo con Matlab:** Filtro chebyshev pasa medio - diseño por transformación de frecuencia, Fig. ??.

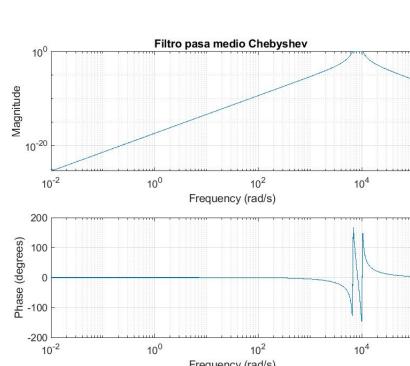
```
1 function [B,A]=mi_lp2bp(Bs,As, w0, B)
2     syms s;
3     H(s)=poly2sym(Bs,s)/poly2sym(As,s);
4
5     % funcion de transformacion f(z)=wc/s
6     H(s)=collect( H( (s^2+w0^2)/(B*s) ) );
7
8     [num,den]=numden(H(s));
9     B=sym2poly(num);
10    A=sym2poly(den);
11
12 end
```

```
1 clear all; clc; close all;
2
3 syms w;
4
5 %% funcion de transformacion
6 f = @(w) (- w^2 + w0^2)/(B*w0);
7
8 %% requerimientos pasa medio (anchos de banda)
9 w0=2*pi*(1326); %rad/s frecuencia en rad/s
10 Bp=2*pi*(625); %rad/s frecuencia en rad/s
11 Bc=2*pi*(865); %rad/s frecuencia en rad/s
12 Bs=2*pi*(2652); %rad/s frecuencia en rad/s
13 Ap=0.1; %dB en w=1
14 As=15; %dB en los extremos de Bs
15 %% represando los requerimientos pasa medio a A_max, w1, w2, Amin, w3, w4
16
17 % Encontrando las frecuencias paso - en el chebyshev es la frecuencia de
18 % borde - w=1
19 [s1]=vpasolve(w^2+Bp*w-w0^2==0,w);
20 f1=s1(2)/2/pi;
21 f2=(Bp+f1*2*pi)/2/pi;
22
23 % Encontrando las frecuencias de corte
24 [s3]=vpasolve(w^2+Bc*w-w0^2==0,w);
25 f3=s3(2)/2/pi;
26 f4=(Bc+f3*2*pi)/2/pi;
27
28 % Encontrando la frecuencias en la banda suprimida
29 [s5]=vpasolve(w^2+Bs*w-w0^2==0,w);
30 f5=s5(2)/2/pi;
31 f6=(Bs+f5*2*pi)/2/pi;
32
33 % Normalizando - chebyshev normaliza con la de paso - es la frecuencia 1
34 % en el butterworth es la frecuencia de corte la frecuencia 1
35 Wc=(f4-f3)/(f2-f1);
36 Ws=(f6-f5)/(f2-f1);
37
38 %% Calculo del orden y la constante de triple
39 [n,e] = mi_chebiord(Wc, Ws, Ap, As)
40
41 %% diseno del filtro pasa bajo
42
43 % Definicion de la expresion los polos
44 syms k integer;
45 theta(k)=(2*k-1) / (2*n) *pi;
46 beta(k)=asinh((1)^k/e)/(n);
47
48 sk(k)=tanh(beta(k)).*(-1).*sin(theta(k)) + j*cos(theta(k));
```

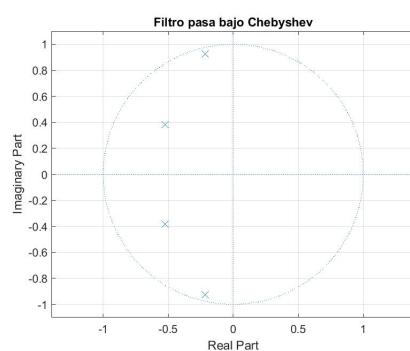
```

49 % Definicion de la funcion de transferencia a partir de los polos
50 syms s;
51 H(s)=collect( prod( 1./(s-sk(1:n)) ) );
52
53 [num, den]= numden(1/H(0)*H(s));
54 BS=sym2poly(num); AS=sym2poly(den);
55
56 % transformacion de paso bajo a paso alto
57 [Bs, As] = mi_lp2bp(BS, AS, w0, Bp)
58
59 % respuesta en frecuencia.
60 fig1=figure;
61 freqs(BS,AS);
62 grid on; title('Filtro pasa bajo Chebyshev ');
63 saveas(fig1,'./chebyshev_ej2_freqs','jpg');
64
65 %% respuesta en frecuencia
66 n=double(n);
67 fig2=figure;
68 freqs(Bs,As);
69 grid on; title('Filtro pasa medio Chebyshev ');
70 saveas(fig2,'./lp2bp_ej1_chebyshev_freqs','jpg');
71
72 % polos y ceros bp
73 fig3=figure;
74 zplane( [zeros(1, n- length(Bs)+1) Bs ], [zeros(1, n- length(As)+1) As] );
75 grid on; title('Filtro pasa medio Chebyshev');
76 saveas(fig3,'./lp2bp_ej1_chebyshev_pz_bp','jpg');
77
78 % polos y ceros lp normalizado
79 fig4=figure;
80 zplane( [zeros(1, n- length(BS)+1) BS ], [zeros(1, n- length(AS)+1) AS] );
81 grid on; title('Filtro pasa bajo Chebyshev');
82 saveas(fig4,'./lp2bp_ej1_chebyshev_pz_lp','jpg');
83

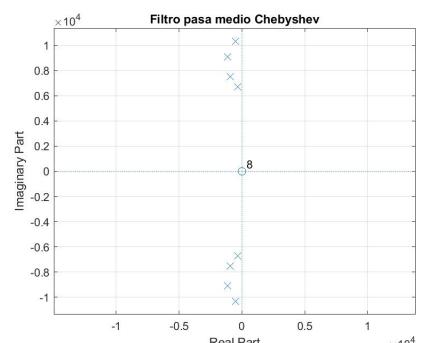
```



(a) Respuesta en frecuencia, filtro pasa banda de Chebyshev



(b) Diagrama de polos y ceros, filtro pasa bajo normalizado de Chebyshev



(c) Diagrama de polos y ceros, filtro pasa banda de Chebyshev

Figura 6.6: Ejemplo 1. Filtro pasa banda de Chebyshev

## 6.4. Comentarios sobre la bibliografía

Daryanani. Principles of active network synthesis and design.

*En la Pág.127. Desarrollo de las transformaciones de frecuencia del filtro pasa alto.*

*En la Pág.129. Desarrollo de las transformaciones de frecuencia del filtro pasa banda.*

# **7 Realización de redes sin perdidas**

## **7.1. Funciones de impedancia y admitancia**

## 7.2. Función de transferencia

# 8 Diseño de Filtros digitales

## 8.1. Introducción

Una forma de caracterizar un filtro digital en tiempo discreto es por medio de una ecuación de diferencias o por una función racional en la variable compleja  $z$ :

$$H(z) = \frac{\alpha \prod_{i=1}^M (1 - z_i z^{-1})}{\prod_{k=1}^N (1 - p_k z^{-1})} = \frac{\prod_{i=0}^M a_i z^{-i}}{1 + \prod_{i=1}^N b_k z^{-k}} \quad (8.1)$$

donde  $N > M$  y

$z_i$  , son los ceros del numerador.

$p_k$  , son los ceros de polinomio del denominador.

Similar a los filtros analógicos, las funciones realizables tendrán la forma de Ec. ?? y el problema que se presenta es el mismo que se encuentra en los filtros analógicos, el cual es que en este tipo de funciones no presentan un valor constante en ninguna banda. Por ende se busca una función que aproxime algunos aspectos de las características deseadas.

Teniendo en cuenta la expresión  $H(z)$  vista en la Ec. ?? se pueden presentar dos enfoques de diseños de filtros.

- Filtros de Respuesta al Impulso infinita, IIR. Cuando  $b_k \neq 0$  , la respuesta al impulso en tiempo discreto consiste de una serie infinita de términos. Para que el filtro sea estable requiere que todos sus polos estén dentro del círculo unitario en el plano  $z$ .

- Filtros de respuesta al impulso finita, FIR: en este caso, la Ec. ??, con  $b_k = 0$ , tomará la forma:

$$H(z) = \alpha \prod_{i=1}^M (1 - z_i z^{-1})$$

Aquí,  $M$  es una cantidad finita y de ahí surge su nombre de respuesta al impulso finito, ya que la respuesta al impulso  $h(n)$  consiste de una serie finita de valores. También se desprende que sus características principales son la estabilidad y la causalidad, es decir:  $\sum_{n=-\infty}^{\infty} |h(n)| < \infty$ .

## 8.2. Filtros IIR

Por la similitud que hay entre la función de transferencia realizable de un filtro digital y la función de transferencia realizable de un filtro analógico (funciones racionales, la primera en  $z^{-1}$ , y la segunda en  $s$ ), los filtros IIR son versiones digitales de los diseños de filtros analógico.

Aprovechando esta similitud, una técnica de diseño consiste en mapear la función racional en  $s$  a otra función racional en  $z$ .



Figura 8.1: Pasos para el diseño de filtros digitales

Por lo tanto el procedimiento, ilustrado en la Fig. ??, se puede resumir en que las especificaciones del filtro se suelen dar en la frecuencia analógica para el tipo de filtro deseado, luego se sigue el procedimiento de diseño para un filtro analógico y por último, se mapea la función de transferencia desde el plano  $s$  al plano  $z$ . Todo esto es siempre que se cumplan dos condiciones siguientes.

1- Condición 1: Preservar las características en frecuencia de los filtros analógicos. Significa que el eje  $j\omega$  se mapea en el círculo unitario en el plano  $z$ , Fig. ??.

$$\{s = j\omega | -\infty < \omega < \infty\} \mapsto \{z = e^{j\theta} | -\pi < \theta \leq \pi\}$$

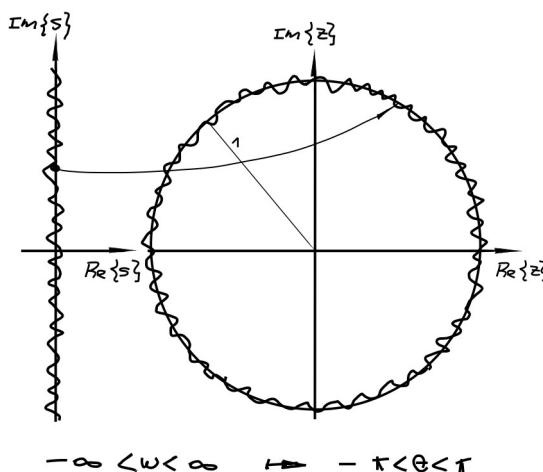


Figura 8.2: Mapeo del eje  $j\omega$ , en el plano  $s$ , en el círculo unitario, en el plano  $z$

2- Condición 2: Preservar las propiedades de estabilidad. Para un sistema estable en  $s$ , no debería haber polos en el semiplano derecho, de forma equivalente para un sistema digital, los polos no deberían estar fuera del círculo unitario, Fig. ??

$$\{s / \operatorname{Re}[s] < 0\} \mapsto \{z / |z| < 1\}$$

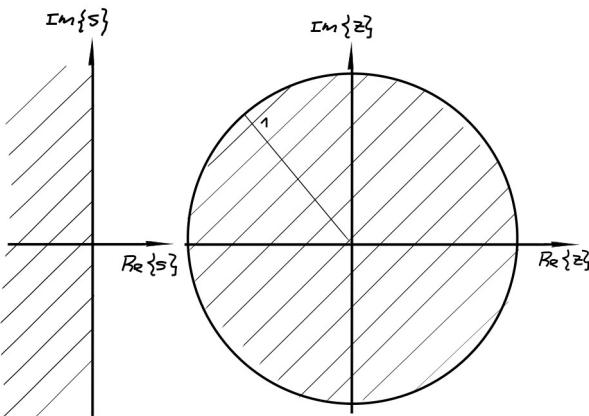


Figura 8.3: Mapeo del semiplano izquierdo del plano s dentro del circulo unitario en el plano z

### 8.2.1. Transformación por Método de Euler

Este método de mapeo es uno de los métodos por integración numérica, surge de aproximar la derivada en tiempo continuo mediante algunas diferencias finitas. El efecto resultante es reemplazar una ecuación diferencial (caracterización de un filtro analógico) con una ecuación de diferencias (caracterización de un filtro digital).

#### Función de mapeo

El método de aproximación de Euler consiste en aproximar la derivada en tiempo continuo con un diferencia de primer orden

$$\left. \frac{dy(t)}{dt} \right|_{t=nT} = \frac{y(n) - y(n-1)}{T}$$

quedando como resultado la función de mapeo,  $s = f(z)$ , desde el plano s al plano z como:

$$s = \frac{1 - z^{-1}}{T} \triangleq f(z)$$

donde  $T$  es el periodo de muestreo, y para el caso de expresar la relación inversa sería

$$z = \frac{1}{1 - sT}$$

Desarrollo:

$$\begin{aligned} \left. \frac{d\hat{y}(t)}{dt} \right|_{t=nT} &= \frac{y(n) - y(n-1)}{T} \xleftrightarrow{\mathcal{L}} sY(s) = \frac{Y(z^{-1}) - z^{-1}Y(z^{-1})}{T} \\ y(k) &= \hat{y}|_{t=nT} \xleftrightarrow{\mathcal{L}} sY(s) = \frac{1 - z^{-1}}{T} Y(z) \end{aligned}$$

#### Verificación de las condiciones de mapeo:

1- Condición 1: conservación de las características de frecuencia. Mapear el eje  $j\omega$  en el plano z resulta en la siguiente expresión

$$(z - 1/2) = 1/2 \cdot e^{j2 \cdot \text{atan}(\omega T)}$$

Como se ve en la Fig. ?? no se cumple la condición, porque no queda circunscrito en el circulo unitario. Pero también se observa que para un  $\Delta\theta = \omega T$  pequeño el mapeo resulta en una buena aproximación de la condición 1.

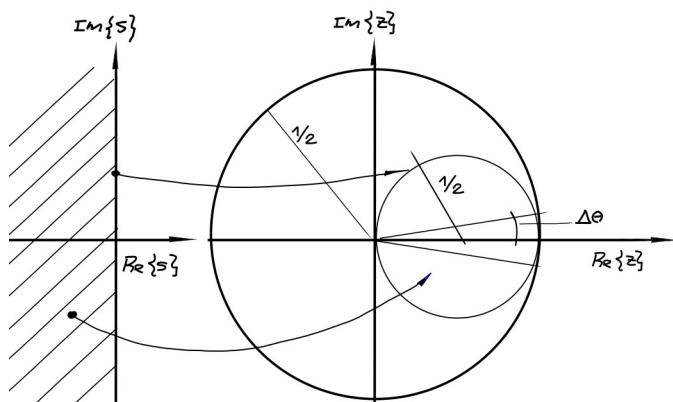


Figura 8.4: Transformación por método de Euler. Mapeo del eje  $j\omega$  en el plano  $z$

2- La condición 2: para preservar la características de estabilidad implica que el semiplano izquierdo  $\text{Real}[s] \leq 0$  quede mapeado dentro del círculo unitario,  $|z| < 1$ . Se verifica ésta condición mediante la siguiente expresión:

$$|z| = \left| \frac{1}{1 - sT} \right|_{s=\sigma+j\omega} = \frac{1}{\sqrt{(1 + T\sigma)^2 + (T\omega)^2}} < 1$$

### 8.2.2. Transformación invariante al impulso

Este procedimiento asegura que la respuesta al impulso del filtro digital  $h(n)$  resultante sea la versión muestreada de la respuesta al impulso correspondiente filtro analógico  $\hat{h}(t)$

$$h(n) \triangleq \hat{h}(t) \Big|_{t=nT}$$

donde  $T$  es el periodo de muestreo.

#### Función de mapeo

La función de mapeo queda establecida como

$$z = e^{sT} \rightarrow \theta = \omega T$$

donde  $\theta$  es la frecuencia digital, y esta comprendida entre  $|\theta| \leq \pi$ .

La función de mapeo surge de comparar ambas respuestas en frecuencia

$$\sum_{i=0}^{\infty} \frac{\hat{\xi}_i}{s + \hat{p}_i} \longrightarrow \underbrace{\sum_{i=0}^{\infty} \frac{\hat{\xi}_i}{1 - e^{\hat{p}_i T} z^{-1}}}_{p_i = e^{\hat{p}_i T}} = \sum_{i=0}^{\infty} \frac{\hat{\xi}_i}{1 - p_i z^{-1}}$$

resultando que el polo en el plano s,  $\hat{p}_i$ , queda mapeado en el plano z como  $p_i = e^{\hat{p}_i T}$ .

Desarrollo:

Para un par transformado de la respuesta al impulso  $\hat{h}(t)$

$$\hat{h}(t) = \sum_{i=1}^N \hat{\xi}_i e^{\hat{p}_i t} \hat{u}(t) \xleftrightarrow{\mathcal{L}} \hat{H}(s) = \sum_{i=1}^N \frac{\hat{\xi}_i}{s + \hat{p}_i} \quad (8.2)$$

muestreando la  $\hat{h}(t)$  a intervalos regulares  $T$  se tiene que

$$h(n) = \hat{h}(t) \Big|_{t=nT} = \sum_{i=1}^N \hat{\xi}_i e^{\hat{p}_i nT} u(n) \xleftrightarrow{\mathcal{Z}} H(z) = \sum_{i=1}^N \frac{\hat{\xi}_i}{1 - p_i z^{-1}} \quad (8.3)$$

Comparando ambos resultados de ?? y de ?? se llega a

$$p_i = e^{\hat{p}_i T}$$

donde  $p_i$  es el polo  $i$  en el plano z correspondiente al polo equivalente  $\hat{p}_i$  en el plano s.

#### Verificación de las condiciones de mapeo:

La característica de frecuencia de la secuencia muestreada,  $x(n)$ , es una suma escalada de un número infinito de copias desplazadas en frecuencia de la característica de frecuencia de la correspondiente señal de tiempo continuo  $x(t)$

$$X(e^{j\theta}) = \frac{1}{T} \sum_{k=-\infty}^{\infty} \hat{X} \left( j \left( \frac{\theta}{T} + \frac{2\pi k}{T} \right) \right) \Big|_{\omega T = \theta} = \frac{1}{T} \sum_{k=-\infty}^{\infty} \hat{X} \left( j \left( \omega + \frac{2\pi k}{T} \right) \right)$$

Gráficamente se puede ver en la Fig. ?? que el mapeo del plano s al plano z resulta en tiras del plano s dentro del círculo unitario, quedando estas en una misma región en z. Por lo tanto, el mapeo no es lineal, es decir que hay más de un punto del plano s que se mapea en un mismo punto en el plano z.

Por lo tanto solo es aplicable a señales de banda limitada que cumplen la condición  $|\hat{H}(j\omega)| \simeq 0$  para  $|\omega| > \omega_B$ . Si no se cumple esa condición ocurre un solapamiento (o aliasing) de las regiones.

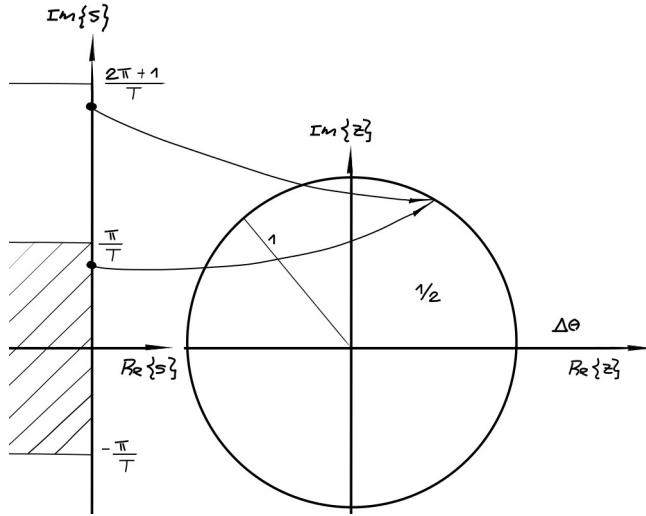


Figura 8.5: Transformación invariante al impulso. Mapeo resultante en el plano z

Desarrollo:

Primeramente, para ver la relación entre la respuesta en la frecuencia digital,  $X(e^{j\theta})$  y la respuesta en frecuencia analógica,  $\hat{X}(j\omega)$  se plantea la transformada de Fourier de la señal muestreada  $\hat{x}(nT)$ . Así que primero se presenta la señal muestreada como un tren de impulsos equis espaciados y escalado de  $x(nT)$  y luego a ésta se le aplica la transformada de Fourier.

$$\begin{aligned}
 x(n) &= \hat{x}(nT) = \hat{x}(t) \cdot \underbrace{\sum_n \delta(t - nT)}_{SF} \xrightarrow{\text{TF}} \hat{x}(j\omega) * \frac{1}{T} \sum_n \delta(\omega - \omega_0) \\
 &\xrightarrow{\text{se si deformar } \left\{ \omega_k = \omega_0 \right.} \hat{x}(t) \cdot \underbrace{\frac{1}{T} \sum_n e^{jn\omega_0 t}}_{\text{TF}} \xrightarrow{\text{transformada de Fourier}} \hat{x}(j\omega) * \frac{1}{T} \sum_n \delta(\omega - \omega_0) \\
 &\xrightarrow{\text{transformado inverso de Fourier}} \frac{1}{T} \sum_n \hat{x}(j\omega) * \delta(\omega - \omega_0) = \boxed{\frac{1}{T} \sum_{-\infty}^{+\infty} \hat{x}(j(\omega - \omega_0))} \\
 &\xrightarrow{\text{transformado inverso de Fourier}} x(nT) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} \frac{1}{T} \sum_{-\infty}^{+\infty} \hat{x}(j(\omega - \omega_0)) d\omega = \boxed{\frac{1}{T} \hat{x}(j(\omega - \omega_0))} = X(e^{j\theta}) \\
 x(n) &= \frac{1}{2\pi} \int_{-\infty}^{+\infty} X(e^{j\theta}) e^{jn\theta} d\theta
 \end{aligned}$$

así se llega a que la respuesta en frecuencia digital, que es la sumatoria de las respuestas analógicas escaladas en  $1/T$  y desplazada en múltiplos enteros de  $\omega_0 = \frac{2\pi}{T}$

$$\frac{1}{T} \hat{X}(j(\omega - \omega_0)) = X(e^{j\theta}) \text{ para } |\theta| \leq \pi$$

Segundo, si se parte de la respuesta al impulso analógica  $\hat{h}(t)$  se debería llegar a la misma relación. Esto es,

$$x(t)|_{t=nT} = \frac{1}{2\pi} \int_{-\infty}^{+\infty} \hat{X}(j\omega) e^{jn\omega t} d\omega \Big|_{t=nT}$$

y luego desarrollando esta expresión tal como

$$\begin{aligned}
 x(\omega) &= \sum_{k=-\infty}^{\infty} \frac{1}{2\pi} \left( \int_{-\frac{2\pi k}{T}}^{\frac{2k+1}{T}\pi} \hat{X}(j\omega) e^{j\omega_n T} d\omega \right) \Big|_{\omega=\omega+\frac{2\pi k}{T}} = \sum_{k=-\infty}^{\infty} \frac{1}{2\pi} \int_{-\frac{\pi}{T}}^{\frac{\pi}{T}} \hat{X}(j(\omega + \frac{2\pi k}{T})) e^{j\omega' n T} d\omega' \\
 \omega &= \omega' + \frac{2\pi k}{T} \\
 \omega_2 &= \frac{2\pi k + \pi}{T} = \omega'_2 + \frac{2\pi k}{T} \\
 \frac{2\pi k + \pi}{T} &= \omega'_2 + \frac{2\pi k}{T} \rightarrow \omega'_2 = \frac{\pi}{T} \\
 \omega_1 &= -\frac{2\pi k - \pi}{T} = \omega'_1 - \frac{2\pi k}{T} \rightarrow \omega'_1 = -\frac{\pi}{T}
 \end{aligned}$$

mediante un cambio de variable queda que

$$\begin{aligned}
 x(n) &= \frac{1}{2\pi} \sum_{k=-\infty}^{\infty} \int_{k=\frac{2k-1}{T}\pi}^{\frac{2k+1}{T}\pi} \hat{X}(j(\omega + \frac{2\pi k}{T})) \Big|_{\omega T = \theta} \\
 &\quad \omega_1 T = \theta_2 \\
 &\quad -\omega_1 T = \theta_1 \\
 &= \frac{1}{2\pi} \sum_{k=-\infty}^{\infty} \int_{-\pi}^{\pi} \hat{X}(j(\frac{\theta}{T} + \frac{2\pi k}{T})) e^{jn\theta} \frac{d\theta}{T}
 \end{aligned}$$

y finalmente se pueden relacionar ambas características en frecuencia como:

$$\begin{aligned}
 X(u) &= \frac{1}{2\pi} \int_{-\pi}^{\pi} \underbrace{\frac{1}{T} \sum_{k=-\infty}^{+\infty} \hat{X}(j(\frac{\theta}{T} + \frac{2\pi k}{T}))}_{\chi(e^{j\theta})} e^{ju\theta} d\theta \\
 X(u) &= \frac{1}{2\pi} \int_{-\pi}^{\pi} \chi(e^{j\theta}) e^{ju\theta} d\theta
 \end{aligned}$$

### 8.2.3. Transformación por mapeo bilineal

#### Función de mapeo:

Se define a partir de la siguiente función de mapeo

$$s = f(z) = \frac{2}{T} \frac{1 - z^{-1}}{1 + z^{-1}}$$

y la relación inversa

$$z = \frac{2+sT}{2-sT} \text{ o de forma equivalente } z^{-1} = \frac{2-sT}{2+sT}$$

y la relación entre la frecuencia digital y analógica

$$\theta(\omega) = \operatorname{atan} \left( \frac{4\omega T}{4 - (\omega T)^2} \right)$$

#### Verificación de las condiciones de mapeo:

De la condición 1, el eje  $j\omega$  queda mapeado en el círculo unitario, así que cumple dicha condición de conservar las características en frecuencia. Y de la condición 2, la región del semiplano izquierdo del plano s queda dentro del círculo unitario en el plano z, Fig. ??

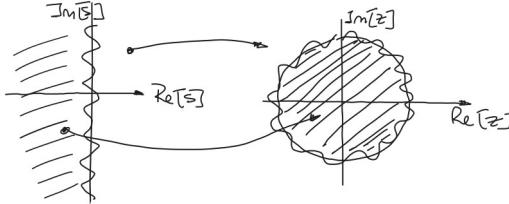


Figura 8.6: Transformación bilineal. Mapeo resultante en el plano z

#### Desarrollo:

1- La condición de que se conservan las características de frecuencia sale de reemplazar  $s = j\omega$  en

$$z = \frac{2 + j\omega T}{2 - j\omega T} = e^{j\operatorname{atan}\left(\frac{4\omega T}{4 - (\omega T)^2}\right)} = e^{j\theta(\omega)}$$

quedando finalmente

$$\theta(\omega) = \operatorname{atan} \left( \frac{4\omega T}{4 - (\omega T)^2} \right)$$

2- De igual manera, pero ahora con  $\sigma \neq 0$  se tiene que  $s = \sigma + j\omega$  y por lo tanto

$$z = \frac{2 + j\omega T}{2 - j\omega T} \Big|_{s=\sigma+j\omega} = \frac{2 + \sigma T + j\omega T}{2 - \sigma T - j\omega T}$$

si  $\operatorname{Real}(s) = \sigma < 0$  entonces  $|z| < 1$ , de esta manera se concluye que también cumple la condición 2. Quedando representado el mapeo en z como se muestra en la Fig. ??.

#### Función de transferencia y respuesta en frecuencia

Como se vio en el desarrollo de las condiciones de mapeo, para preservar las características de frecuencia y estabilidad éste mapeo bilineal cumple con ambas propiedades, quedando relacionadas de la siguiente manera ambas respuestas en frecuencia.

$$H(z) = \hat{H}(s) \Big|_{s=(2/T)(1-z^{-1})/(1+z^{-1})}$$

donde  $T$  es el período de muestreo.

Pero la relación entre las frecuencias digitales  $\theta$  y la frecuencia analógica  $\omega$  no es uniforme, Fig. ??, esto es

$$s = \sigma + j\omega = \frac{2}{T} \frac{1 - e^{-j\theta}}{1 + e^{-j\theta}} = j \frac{2}{T} \tan \frac{\theta}{2} \begin{cases} \omega = \frac{2}{T} \tan \frac{\theta}{2} \\ \sigma = 0 \end{cases}$$

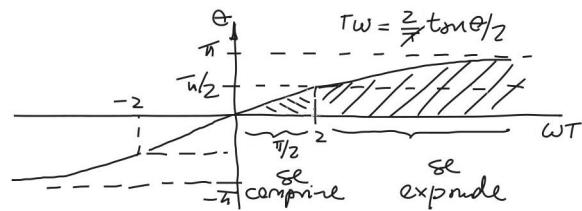


Figura 8.7: Transformación bilineal. Relación entre la frecuencia analógica  $\omega$  y la frecuencia digital  $\theta$

### Predistorsión – prewarping

La predistorsión es un procedimiento que consiste en encontrar una constante o una frecuencia de muestreo  $\omega_s = \frac{2\pi}{T}$  equivalente, tal que dada una frecuencia analógica,  $\omega_x$ , coincida con el mapeo inverso de una frecuencia digital relacionada linealmente con esta frecuencia analógica dada.

$$\omega_x T'/2 = \tan\left(\frac{\theta}{2}\right) \Big|_{\theta=\omega_x T/2}$$

Ya que para la frecuencia de muestreo utilizada,  $\omega T < \pi/2$ , las frecuencias analógica se comprimen, y  $\omega T > \pi/2$  las frecuencias expanden respecto a la frecuencia analógica, Fig. ??.

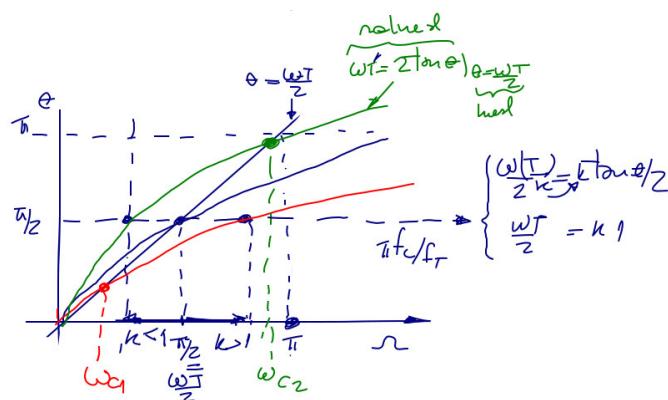


Figura 8.8: Transformación bilineal. Relación entre la frecuencia analógica  $\omega$  y la frecuencia digital  $\theta$

Entonces, se busca un punto donde coincide la relación lineal con la relación binomial, Fig. ??.

A continuación se busca una frecuencia equivalente de muestreo tal que se relacionen esta dos relaciones lineal y no lineal (intersección de los dos puntos.)

$$\begin{aligned} \omega_x T' &= 2 \tan \frac{\omega_x T}{2} \\ \omega_x T' &= 2 \tan \frac{\omega_x T}{2} \\ \omega_s &= \frac{\omega_x T}{2 \tan \frac{\omega_x T}{2}} \\ \boxed{\omega_s = \frac{\omega_x T}{2 + \tan \frac{\omega_x T}{2}}} \end{aligned}$$

la frecuencia de muestreo equivalente para que se cumpla que

$$\omega_x \frac{T}{2} = \theta_x$$

## 8.3. Filtros FIR

### Definición

La función de transferencia de un filtro FIR tiene la forma:

$$H(z) = \sum_{n=0}^N h(n)z^{-n}$$

$h(n)$  es una secuencia que representa la respuesta al impulso en tiempo discreto, de longitud finita y causal.

Hay varias formas de diseñar un filtro FIR, una forma es por truncamiento de la respuesta al impulso, y la otra forma puede ser por muestreo de frecuencia.

Aquí se desarrolla solo el primer método. En este método se especifica la respuesta en frecuencia,  $H(j\omega)$ , y luego se plantea la serie de Fourier de ésta (en  $j\omega$ ), la secuencia resultante representa la respuesta al impulso en tiempo discreto,  $h_d(n)$ . Como esta secuencia tiene duración infinita y es no causal, se multiplica por una ventana tal que trunque la secuencia fuera de los límites de dicha ventana, y finalmente se desplaza de tal modo que quede causal.

En base a lo dicho al párrafo anterior,  $h(n)$  va ser una secuencia de tiempo de discreto causal y de longitud  $N$  finita, y esto hace que se den cuatro tipos o casos de simetrías.

- Tipo I: Simetría respecto a  $\pi$  con una cantidad  $N$  impar de muestras.
- Tipo II: Simetría respecto a  $\pi$  con una cantidad  $N$  par de muestras.
- Tipo III: Anti simetría respecto a  $\pi$  con una cantidad  $N$  impar de muestras.
- Tipo IV: Anti Simetría respecto a  $\pi$  con una cantidad  $N$  par de muestras.

Para el tipo II y IV solo es aplicable a filtros pasa bajos y pasa medios, y los otros tipos no tienen restricciones. Para que el filtro fuese de fase lineal  $h(n)$  debe cumplir la condición:

$$h(n) = h(N - 1 - n)$$

y esto asegura también que sea causal.

Otra característica de la aproximación por truncamiento de la respuesta del impulso ideal es que se da el fenómeno de convergencia no uniforme de Gibbs en la discontinuidades. En este sentido se utilizan diferentes ventanas para atenuar dichos sobre-impulsos.

### Desarrollo

Para comenzar con la aproximación por truncamiento de la respuesta del impulso se especifica el filtro ideal en el dominio de la frecuencia,

$$H(j\omega T = j\Omega) = \begin{cases} 1 & |\omega T| < \omega_c T = \Omega_c \\ 0 & \text{para otro valor} \end{cases}$$

donde  $\Omega = \omega T$  es la frecuencia normalizada de  $\omega$  con respecto a la frecuencia de muestreo  $1/T$ , entonces  $\Omega$  va variar entre  $-\pi$  y  $\pi$ ;  $\Omega_c$  es la frecuencia de corte normalizada, y  $\omega_c$  es la frecuencia de corte no normalizada.

Se supone que  $H(j\Omega)$  es periódica de período  $2\pi$ , quedando el desarrollo en serie de Fourier de la respuesta en frecuencia  $H(j\Omega)$ , representada en la Fig. ??, tal como

$$H(j\omega T = j\Omega) = \sum_{n=-\infty}^{n=\infty} b_n e^{-j \cdot 1 \cdot n}$$

donde la frecuencia fundamental es 1, entonces  $\omega_0 = \frac{2\pi}{2\pi} = 1$ . En este caso,  $b_n$  se asocia directamente con la secuencia en tiempo discreto  $h_d(n)$ , Fig. ??, y que corresponde a los coeficientes de  $H(j\omega)$ .

$$h_d(n) = b_k = \frac{1}{2\pi} \int_{-\Omega_c}^{\Omega_c} 1 \cdot e^{-j1 \cdot n \Omega} = \frac{\sin(\Omega_c n)}{\pi n}$$

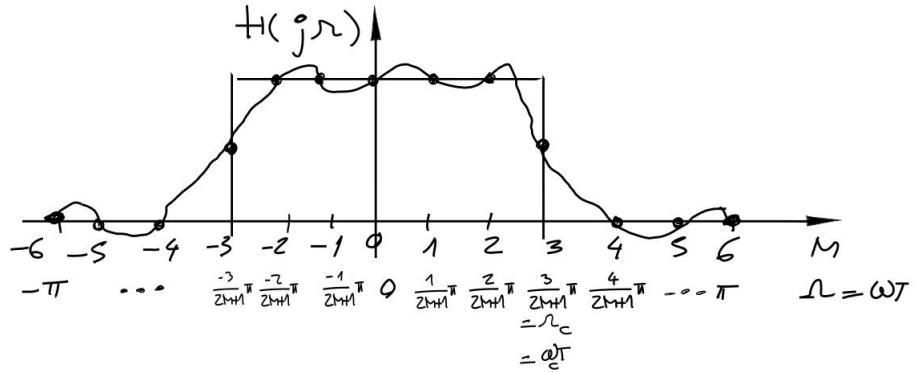


Figura 8.9: Representación gráfica de  $H(j\omega)$  y de su desarrollo serie de Fourier en  $j\omega$

Esta secuencia,  $h(n)$  tiene una cantidad infinita de términos y además no es causal. Una forma de resolver la causalidad

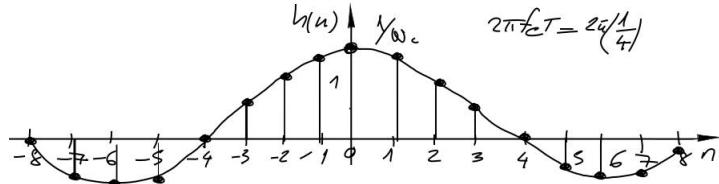


Figura 8.10: Representación gráfica de la respuesta al impulso  $h_d(n)$ .

es multiplicar esta secuencia por una ventana que trunque la secuencia a una cantidad de términos diferentes de cero y luego desplazar esta secuencia de tal forma que quede causal. En primera instancia la ventana  $w(n)$  puede ser una ventana rectangular de longitud  $N$ , tal como se muestra trazada en rojo en la Fig. ??.

$$h(n) = h_d(n) \cdot w(n) \quad (8.4)$$

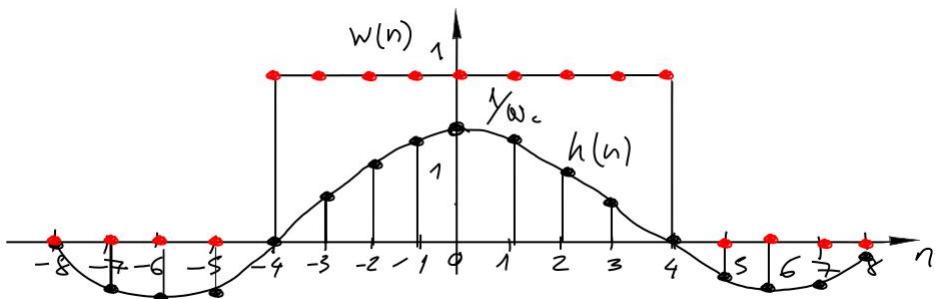


Figura 8.11: Representación gráfica de  $h_d(n)$  y  $w(n)$  (trazado en rojo)

Finalmente, a partir de la Ec. ??,  $h(n)$  será una secuencia de longitud  $N$  finita y causal, lo que para un  $N$  par la expresión quedaría definida como:

$$h(n) = \begin{cases} \frac{\sin(\Omega_c \pi (n - \frac{N-1}{2}))}{\pi (n - \frac{N-1}{2})} & \text{para } n! = \frac{N-1}{2} \\ \Omega_c & \text{para } n = \frac{N-1}{2} \end{cases} \quad (8.5)$$

En la Fig. ?? representa el caso de longitud impar dado en la Ec. ??.

Y para una secuencia  $h(n)$  de  $N$  par se tiene:

$$h(n) = \frac{\sin(\Omega_c \pi (n - \frac{N-1}{2}))}{\pi (n - \frac{N-1}{2})} \quad (8.6)$$

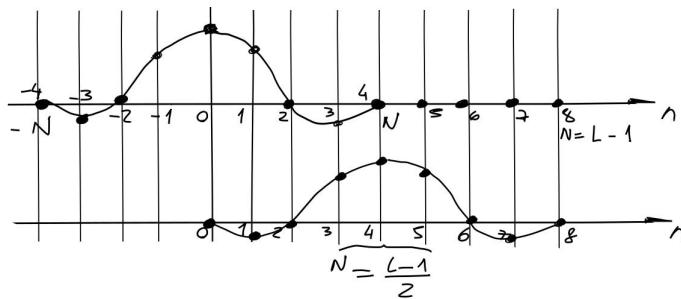


Figura 8.12: Representación de  $h_d(n)$  y de  $h(n)$  para  $N$  impar.

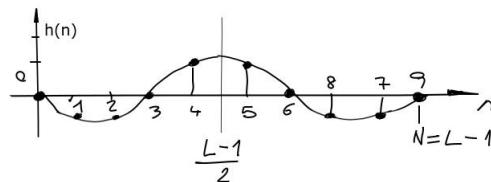


Figura 8.13: Representación de  $h_d(n)$  y de  $h(n)$  para  $N$  par.

En la Fig. ?? representa el caso de longitud par dado en la Ec. ?? .

**Ejemplo con Matlab 1:** Filtro pasa bajos FIR, Fig. ??.

```

1 clear all; clc; close all;
2
3 N=15;      % longitud
4 Wc=0.3;    % frecuencia de corte normalizada wct
5 esLP=1;    % es filtro pasa bajos
6
7 hd0=Wc;
8 if esLP==0; Wc=2-0.3; hd0=1-hd0; end;
9
10 syms n;
11 m=(N-1)/2 ;
12 h= @(n) sin(Wc*pi*( n - m ) )./(pi*( n - m));
13
14
15 if(mod(N,2)==1)
16     hd= [h(0:(N-3)/2) hd0 h((N+1)/2:(N-1)) ];
17 else
18     hd=[h(0:N-1) ];
19 end
20
21 fig=figure;
22 freqz(hd,1); title('Filtro FIR');
23 saveas(fig,'./fir1_ej1','jpg');
```

**Ejemplo con Matlab 2:** Filtro pasa altos FIR, Fig. ??.

En este ejemplo se construyó una función que permite determinar el filtro y posibilite elegir el tipo de filtro y el tipo de ventana a aplicar al mismo.

```

1 % [Bz,Az]=mi_fir1(N, Wn*2, TIPO_FILTRO, TIPO_VENTANA);
2 function [B,A]=mi_fir1(varargin)
3
4     % N, W, VENTANA
5     % number of arguments check
6     narginchk(2,4);
7     N=double(varargin{1});
8     Wc=double(varargin{2});
9     TIPO="paso bajo";
10    if nargin>=3
11        TIPO=varargin{3};
```

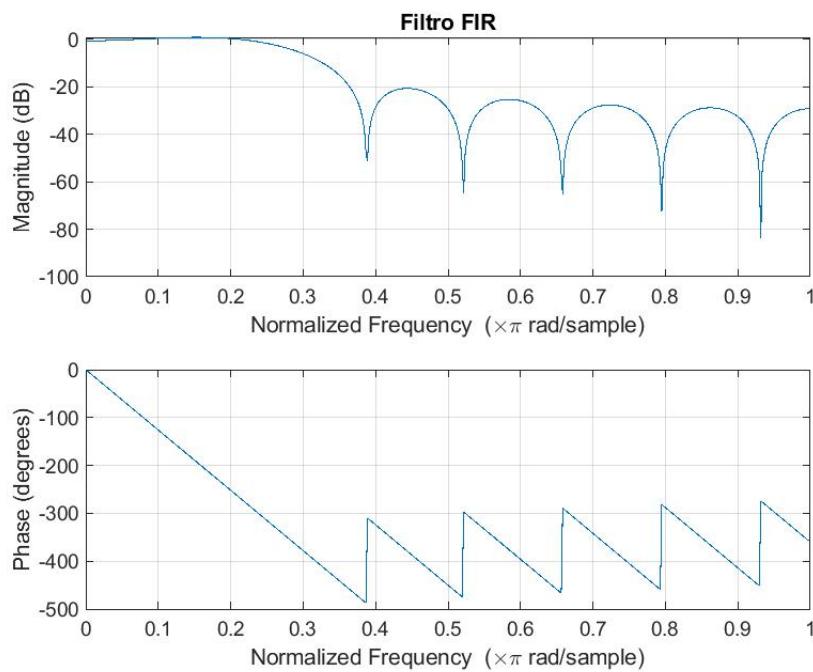


Figura 8.14: Respuesta en frecuencia.

```

12
13 end
14 VENTANA="rectangular";
15 if nargin>=4
16     VENTANA=varargin{4};
17 end
18
19 % seleccion de la ventana
20 w = @(m) 1;
21 switch VENTANA
22     case "rectangular"
23         w = @(m) 1;
24     case "hanning"
25         w = @(m) 1/2*[1-cos(2*pi*m/(N-1))];
26     case "hamming"
27         w = @(m) 0.54 - 0.46*cos(2*pi*m/(N-1));
28     case "blackman"
29         w = @(m) 0.42 - 0.5*cos(2*pi*m/(N-1))- 0.08*cos(4*pi*m/(N-1));
30 end
31
32 %seleccion del tipo de filtro
33 hd0 = Wc(end);
34 switch TIPO
35     case "pasa alto"
36         Wc=2-Wc(end);
37         hd0=1-hd0;
38 end
39
40 syms n ;
41 m =(N-1)/2 ;
42 h = @(n) sin(Wc(end)*pi*(n - m ))./(pi*(n-m));
43
44 if ( mod (N ,2) ==1)
45     B=[ h(0:(N-3)/2) hd0 h((N+1)/2:(N-1))] .*w(0:N-1);
46 else
47     B=[ h(0:N-1) ].*w(0:N-1);
48 end
49 A=1;
50 end

```

A continuación se muestra el uso de esta función con un ejemplo:

```

1 clear all; clc; close all;
2
3 N=61;      % longitud
4 Wn=0.3;    % frecuencia de corte normalizada wcT
5 esLP=1;    % es filtro pasa bajos
6
7 TIPO_FILTRO="pasa alto";
8 TIPO_VENTANA="hamming";
9
10 [Bz,Az]=mi_fir1(N, Wn*2, TIPO_FILTRO, TIPO_VENTANA);
11
12 fig=figure;
13 freqz(Bz,Az); title('Filtro FIR');
14 saveas(fig,'./fir1_ej2','jpg');

```

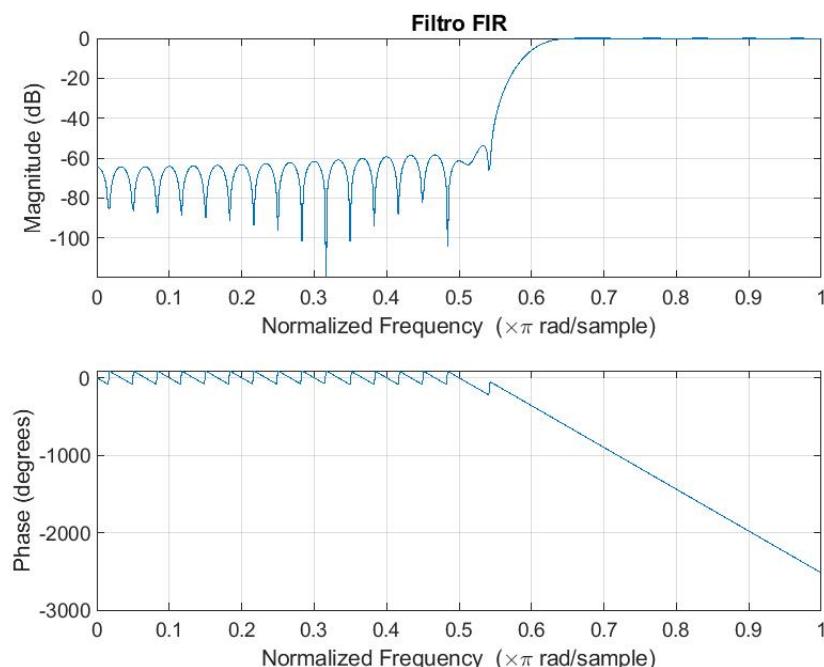


Figura 8.15: Respuesta en frecuencia.

## 8.4. Comentarios sobre la bibliografía

Lam Harry. Analog and digital filters design and realization.

*Desarrollo general.*

Oppenheim y Schafer. Digital Signal Processing. 1975.

*Desarrollo de los filtros FIR.*

Schlichthärle, Dietrich. Digital Filters: Basics and Design. Second Edition

*Las expresiones de las Ventanas para los filtros FIR.*

Gaydecki P. Foundations of digital signal processing theory.

*En la página 327. Tabla 11.1 tiene las ecuaciones de respuesta al impulso bien desarrolladas.*

# 9 Realización de filtros Digitales

## 9.1. Realización

Los filtros digitales que son realizables mediante elementos computacionales estarán dados por una función racional en  $z^{-1}$

$$H(z) = \frac{\sum_{k=0}^M b_k z^{-k}}{1 - \sum_{k=1}^N b_k z^{-k}} = \frac{Y(z)}{X(z)}$$

o de forma equivalente

$$Y(z) = \sum_{k=1}^N a_k Y(z) \cdot z^{-k} + \sum_{k=0}^M b_k X(z) \cdot z^{-k}$$

y en la caracterización correspondiente en tiempo discreto estará dada por una ecuación de diferencias.

$$y(n) = \sum_{k=1}^N a_k y(n-k) + \sum_{k=0}^M b_k x(n-k) \quad (9.1)$$

La realización del filtro digital se expresa mediante un algoritmo, que puede a su vez puede ser construido en hardware dedicado o hardware de propósito general mediante software. En ambos casos, realizados a través de elementos computacionales sencillos, tales como, sumas (Fig. ??), productos de constantes (Fig. ??), y retardos temporales (Fig. ??).

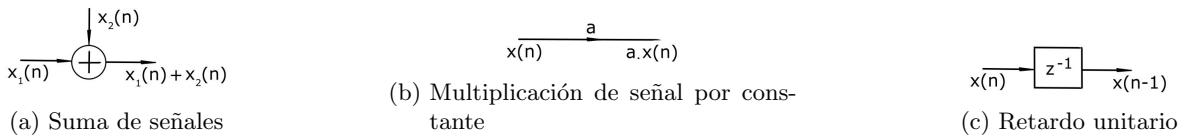


Figura 9.1: Representación en bloques de elementos computacionales básicos

Estos algoritmos se dan en diferentes estructuras o tipologías que representan en diagramas de bloques, diagramas de flujo o notación matricial. Entre estas estructuras se encuentran:

1- Formas directas: la  $H(z)$  se realiza en una sola pieza. Se clasifican en Formas directas I y II, estructura de escalera, estructura de celosías, extracción de multiplicadores, y formas modulares de filtros de ondas.

2- Formas indirectas: la  $H(z)$  se descompone en varias secciones de primer y segundo orden. Mediante las formas directas se consiguen estructuras de primer y segundo orden, y luego se combinan de algún modo, como en cascada.

### Forma directa I y II

La Ec. ?? puede ser representada por el diagrama de bloques como muestra la Fig. ?? y puede ser reorganizado con el diagrama de la Fig. ??, esta última forma se denomina forma directa I.

El flujo de las señales que se dirigen hacia los bloques de retardo que representa la forma directa I se pueden cambiar de tal forma que permitan reorganizar los bloques combinando dos retardos en un solo bloque. Esto permitiría ahorrar unidades de memoria al minimizar los registros de desplazamiento paralelos utilizados para realizar para realizar los retardos, esta forma se denomina segunda forma canónica o forma directa II y se muestra en la Fig. ??.

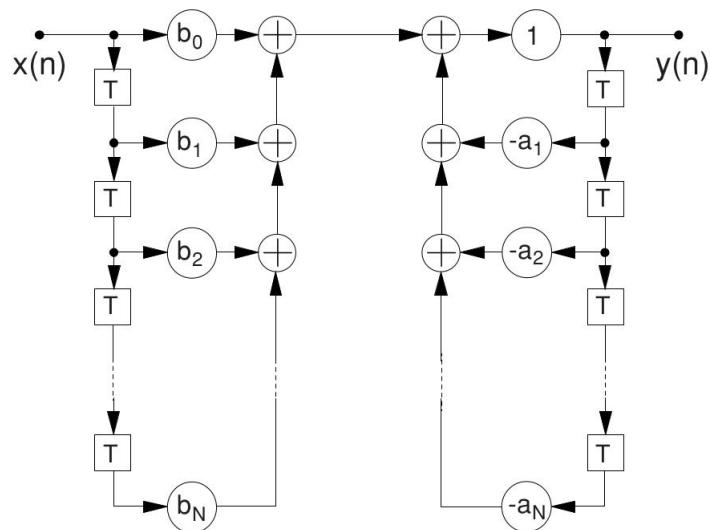


Figura 9.2: Representación de la Ec. ??

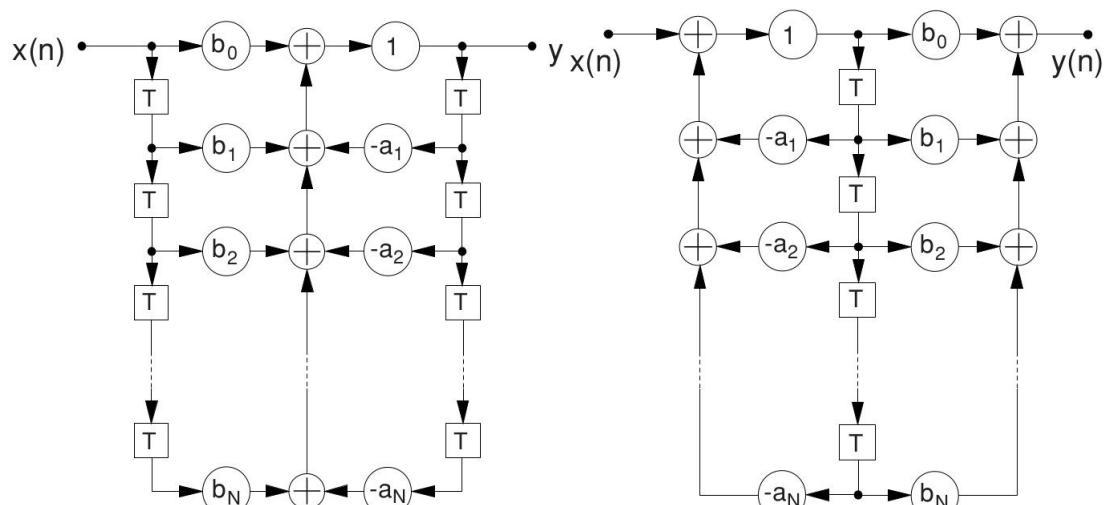


Figura 9.3: Forma directa I y II

## **9.2. Comentarios sobre la bibliografía**

Rabiner, L. R., and Gold, B. Theory and Application of Digital Signal Processing. Englewood Cliffs, N.J.: Prentice-Hall, Inc., 1975.

*Desarrollo general.*

Schlichthärle, Dietrich. Digital Filters: Basics and Design. Second Edition

*Descripción de la formas directa I y II.*

# 10 Procesamiento digital-construcción de filtros digitales

## 10.1. Realización

En general, un sistema de procesamiento de señales digitales se puede representar como se muestra en el diagrama, Fig. ???. El bloque de conversión a digital, ADC, muestrea y cuantifica la señal analógica  $\tilde{x}(t)$  en una secuencia de tiempo discreto  $x(n)$ , seguido se trata la señal digital mediante algún algoritmo de procesamiento digital, DSP, y finalmente se reconstruye la secuencia discreta resultante  $y(n)$  en una señal de analógica de tiempo continuo  $\tilde{y}(t)$ . Adicionalmente se incluyen filtros limitadores de frecuencia en cada uno de los extremos, uno en la entrada con el fin de rechazar el ingreso de frecuencias indeseadas, que van a estar por encima de la mitad de la frecuencia muestreo del sistema, y otro filtro limitador en la salida para descartar las componentes de frecuencia superiores y así suavizar la salida.



Figura 10.1: Diagrama de bloques, procesamiento digital de señales

El algoritmo computacional de un filtro digital puede ser construido mediante un hardware de propósito general o un hardware de dedicado. En un hardware dedicado, los elementos básicos computacionales, como las sumas, multiplicación por constantes y retardos se implementan mediante compuertas y unidades de memorias, ya sea con componentes discretos o integrados en un microcircuito. Otra forma de representar estos algoritmos es por medio de software, ejecutado en algún hardware de propósito general, como puede ser un microcontrolador o una computadora personal.

### Construcción mediante software, ejecutado por medio de un microcontrolador

La implementación del algoritmo de un filtro digital puede haber ligeras variaciones entre una y otra, dependiendo del lenguaje de programación, la técnica de bufferización que ofrezca el hardware, etc. Por ejemplo, los microcontroladores avanzados traen instrucciones especiales de procesamientos de señales, conocidos como DSP. Estas instrucciones pueden ser como, por ejemplo, la convolución de señales.

Una representación general del sistema en bloques se muestra en la Fig. ???. Los bloques ADC, y DAC son hardware específico, que en la mayoría de los casos se incluye junto al microcontrolador, en el mismo microcircuito. El bloque de procesamiento es implementado mediante rutinas de software.

En la Fig. ??, se muestra de forma simplificada las rutinas asociadas al bloque DSP de un filtro FIR con su lista de coeficientes, y superpuesto a este diagrama se asocia otro diagrama de bloques que representa el algoritmo completo del filtro.

### Construcción mediante hardware específico

En la construcción de un filtro con componentes discretos o un microcircuito que integre el diseño con componentes discretos se pueden presentar dificultades prácticas, esto es porque para el primero se necesita cablear una enorme cantidad de componentes y para el segundo, las dificultades pueden ser simplemente por la imposibilidad de acceso a la tecnología necesaria. Una alternativa viable es la utilización de FPGAs (o tecnologías similares).

Las FPGAs y similares consisten en un microcircuito con una enorme cantidad de celdas lógicas configurables. Así se pueden construir mediante algún lenguaje de especificación de hardware directamente o indirectamente (por medio

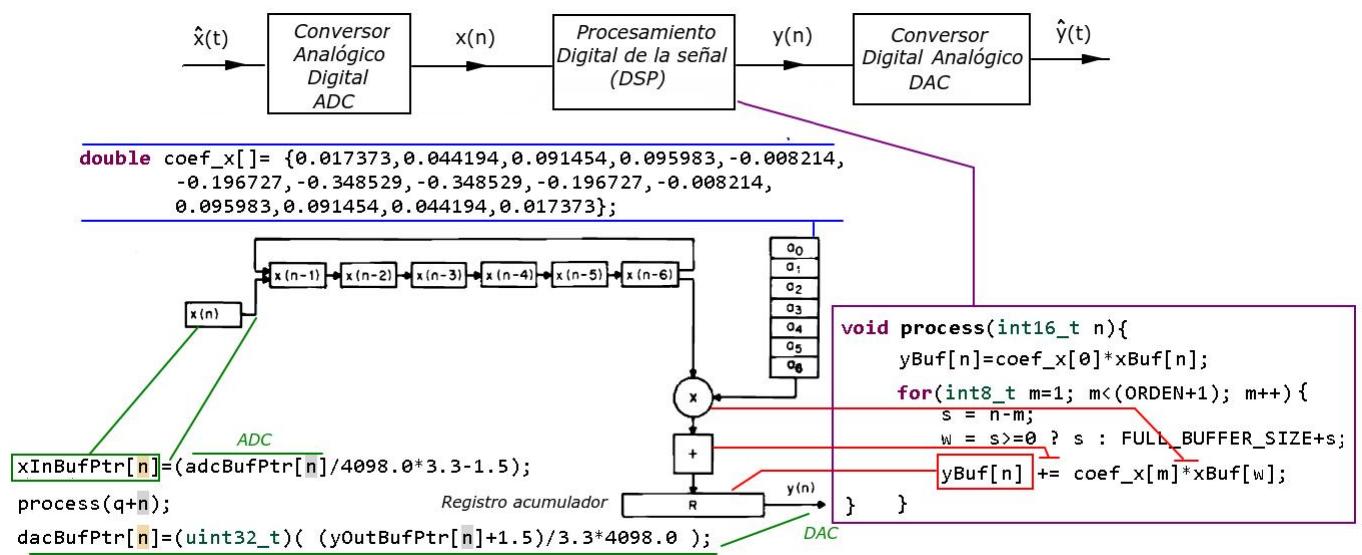
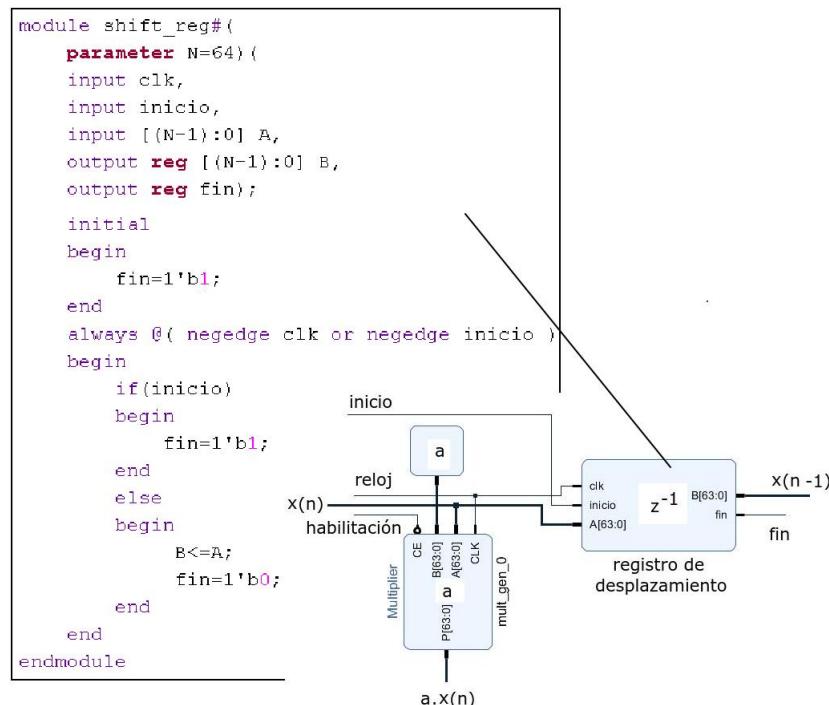


Figura 10.2: Representación del algoritmo de un filtro FIR, rutinas en c correspondientes al proceso, asociado al diagrama general de procesamiento digital, ??

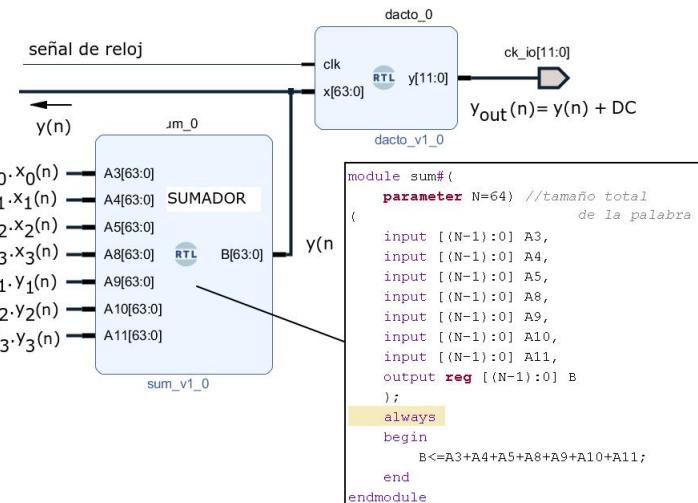
de un esquemático) circuitos digitales de diversas complejidades.

Dependiendo del entorno de desarrollo que se utilice se puede ir definiendo los bloques fundamentales del filtro y luego ir interconectándolos hasta lograr el filtro de las características deseadas.

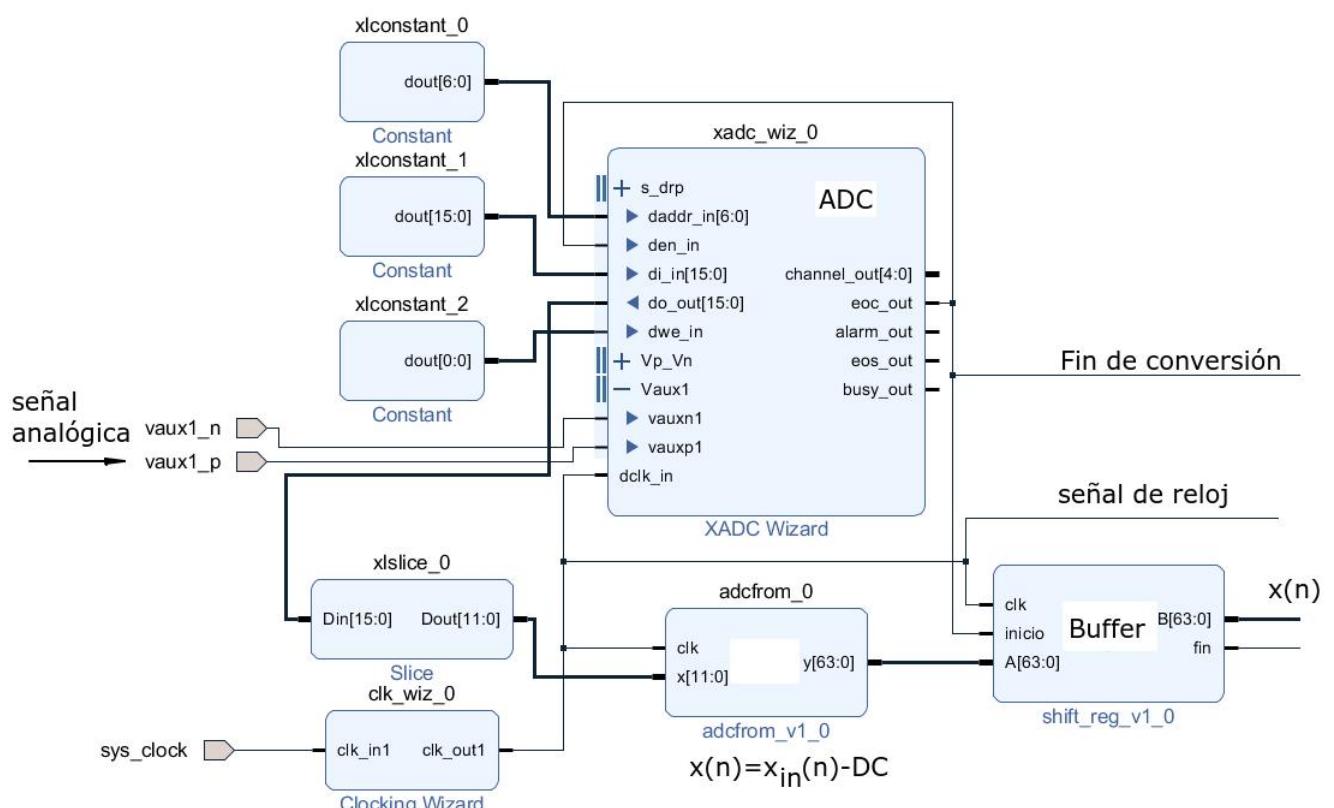
En la Fig. ?? se muestra cada uno del conjunto de bloques representativos junto con el código de descripción de hardware para cada una de las operaciones básicas necesarias para la implementación del hardware. Estos bloques forman parte del filtro construido que se muestra en la Fig. ??.



(a) Representación del bloque de retraso y del bloque de multiplicación de una constante por la señal



(b) Implementación del bloque sumador de señales y representación del bloque DAC



(c) Representación del bloque ADC

Figura 10.3: Implementación de los bloques de operaciones mediante FPGA

## **10.2. Comentarios sobre la bibliografía**

Rabiner, L. R., and Gold, B. Theory and Application of Digital Signal Processing. Englewood Cliffs, N.J.: Prentice-Hall, Inc., 1975.

# 11 Especificaciones de las implementaciones de los filtros realizados

## 11.1. Especificaciones generales del hardware utilizado

Implementación realizada utilizando Microcontrolador

El montaje completo consiste de una placa embebida de bajo costo, una interfaz para la grabación y depuración del código en el microcontrolador, un circuito para proteger las entrada analógica, y un circuito de salida que actúa como carga, Fig. ??.

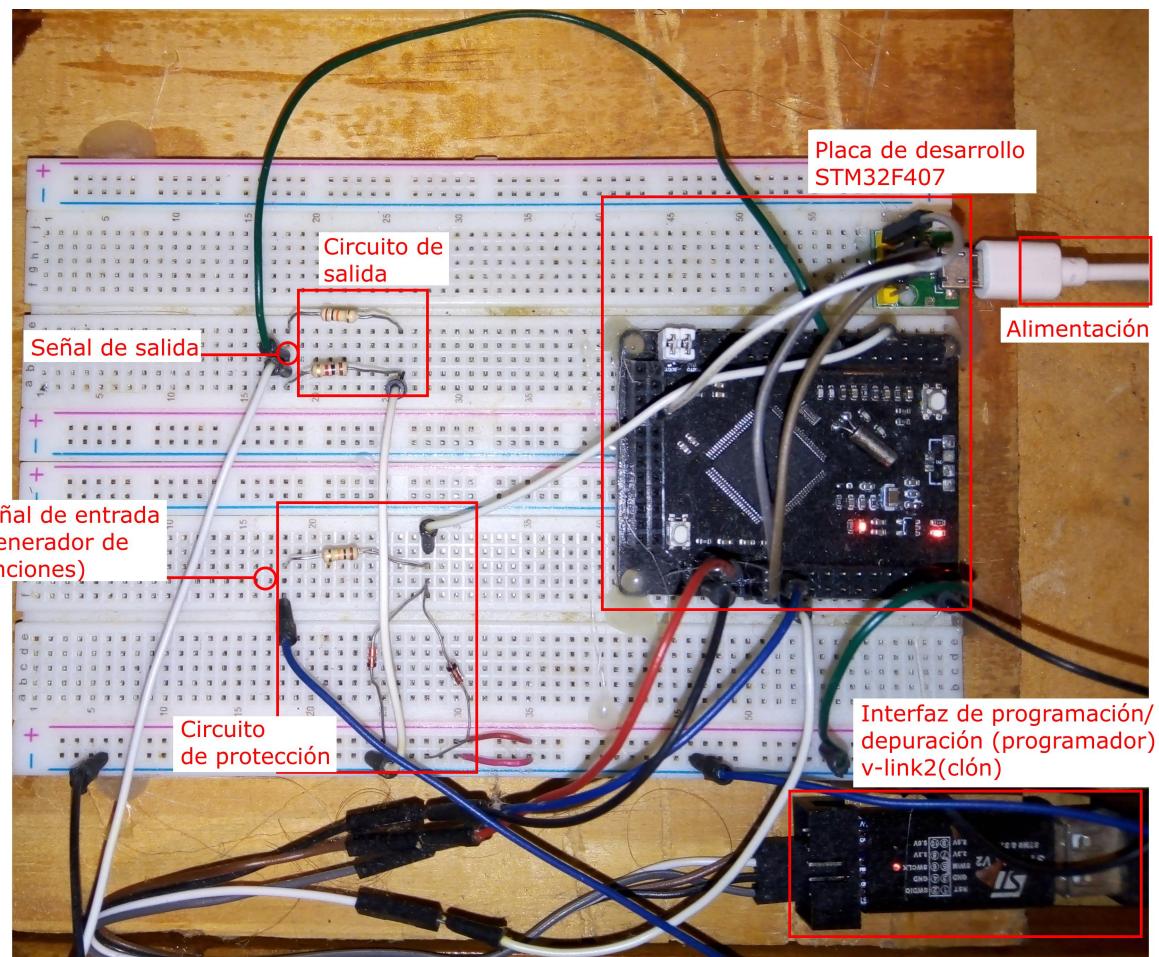


Figura 11.1: Montaje del sistema embebido con microcontrolador

Las características generales del microcontrolador de interés que posee la placa embebida para esta aplicación son:

- La denominación comercial del microcontrolador es STM32F407VGT.
- La CPU es de arquitectura ARM, Corte-M4-32 bits, con una señal de reloj de sistema máxima de 168 MHz.
- El rango de voltaje de alimentación es de 1.8 de 3.3V (valor típico).
- El módulo de conversión analógica-digital, ADC, incluido: tiene 3 unidades de 12-bit, con un voltaje máximo de entrada menor o igual al voltaje de alimentación, y la tasa de muestreo ronda entre 2 a 6 MSPS, este valor de operación es dependiente de la alimentación y de su configuración; la impedancia de entrada es de 50 kΩ, el valor de resistencia de muestreo es de 6 kΩ y el capacitor de muestreo es aproximadamente de 4 pF.

- El módulo de conversión digital-analógica, DAC, incluido: tiene dos unidades. Cada uno tiene una resistencia de carga de  $5K\Omega$ , eso es si tiene el buffer de salida activado, una impedancia de salida de  $15 k\Omega$  y una capacitancia parásita de  $50 pF$ .

- El módulo de temporización incluido, Timer: tiene 17 unidades, hay de 16 bits y de 32 bits, con frecuencia de operación máxima de 168 MHz.

La arquitectura de la implementación del sistema embebido para el desarrollo de los filtros digitales se puede resumir en la Fig. ??.

El Timer es simplemente un contador que toma la señal de reloj del bus interno del microcontrolador al que está

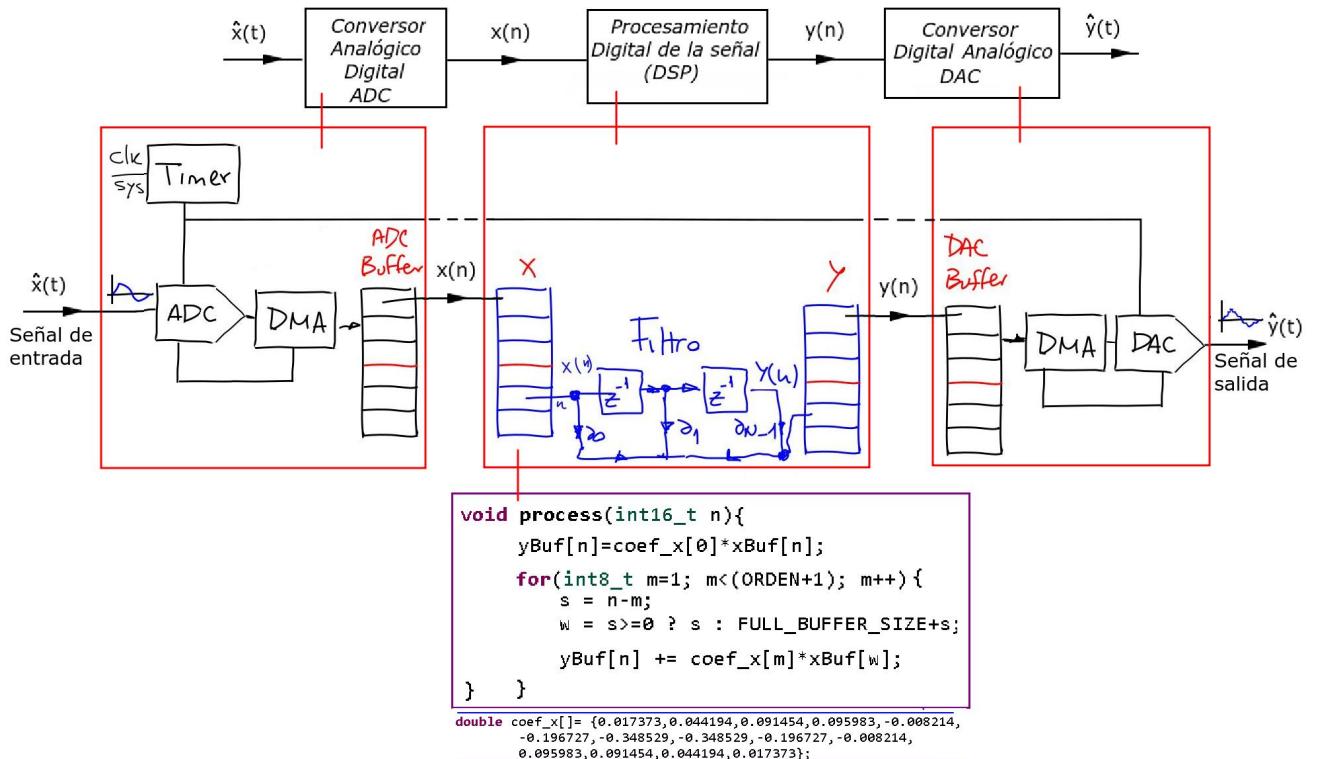


Figura 11.2: Representación en bloques de la arquitectura desarrollada para los filtros digitales por medio de microcontrolador

conectado, y a partir del valor de la cuenta cargado en el registro de comparación genera señales de menor frecuencia. El principio de funcionamiento consiste en iniciar un registro de comparación y cuando el contador desborda o llega al valor de comparación precargado genera una señal de activación y luego se reinicia para generar otro ciclo de cuenta. En este caso, se genera una señal para disparar el inicio de conversión del módulo ADC y del módulo DAC. La frecuencia de esta señal de control se considera la frecuencia de muestreo.

El cálculo del valor que configurará en el registro de comparación del Timer va a determinar la frecuencia de muestreo utilizada. Éste valor de registro va a depender de la frecuencia de la señal de reloj del bus que alimenta al Timer en cuestión (en este caso  $freq_{bus} = 84MHz$ ), a esta frecuencia se la divide por la cantidad de cuentas que va a realizar menos uno ( $reg_{contador} = 1252$  cuentas) y también, se divide por el valor del registro divisor de frecuencia mas uno (el valor del divisor es  $prescaler = 2$ ). Con todo esto la frecuencia de muestreo queda determinada por:

$$F_m = \frac{freq_{bus}}{(reg_{contador} - 1) * (1 + prescaler)} = \frac{168MHz}{(1252 - 1) \cdot (1 + 2)} = 22,418kHz$$

Ambos módulos para las conversiones de analógico a digital y viceversa son configurados para que trabajen junto con el bloque de Acceso Directo a Memoria, DMA. Este último bloque gestiona la transferencia de datos entre los módulos de conversión y los buffers de memoria de entrada y salidas correspondientes. El uso del módulo DMA libera a la CPU de las rutinas de copiado de los valores de los registros de entradas y salidas hacia, o desde, los buffers, aumentando las posibilidades de procesamiento del CPU.

Tanto el buffer de entrada, como el de salida, se dividen cada uno en dos mitades. El objetivo de esto es que mientras, por ejemplo, el módulo ADC llena una de las mitades del buffer, el bloque de procesamiento de señal (DSP) consuma los datos de la otra mitad del buffer, estos datos producidos por el ADC en instantes anteriores, evitando interferencias entre el productor de los datos (ADC) y el consumidor de los datos (DSP).

El bloque de procesamiento de señales digitales, DSP, está implementado totalmente por software, básicamente hace las operaciones computables de sumar, multiplicar por constantes, y retardos mencionadas en la sección anterior. Estas operaciones combinadas conforman el algoritmo del filtro desarrollado.

Respecto a la operación de retardo hay dos formas de implementarla:

1- Teniendo registros para cada una de las muestras de la secuencia a almacenar. Todos estos juntos conforman el buffer de entrada y salida del bloque de proceso de señales. Para poder crear el efecto de desplazamiento temporal se sobre-escribe el registro que representa el valor de la señal en un tiempo particular con el valor del registro en el instante siguiente, finalizando la operación de copia con el primer registro al que se copia el valor actual del ADC.

$$\begin{aligned} x(2) &\xleftarrow{\text{copiar}} x(1) \\ x(1) &\xleftarrow{\text{copiar}} x(0) \\ x(0) &\xleftarrow{\text{copiar}} \text{ADC} \quad \text{valor actual} \end{aligned}$$

Esta técnica se utiliza frecuentemente en los desarrollos con placas Arduino. De realizar esto con componentes discretos (caso de implementación con hardware) se utilizarían registros de desplazamientos.

2- A diferencia del modo del caso anterior descripto, va a haber una serie de registros concatenados formando dos buffers conectados en anillo (el primero interconectado con el último). En esta técnica no se van sobre-escribiendo los valores de los registros con los valores del registro anterior, sino que se trabaja con un índice que incrementa el puntero inicial y final de la cola de secuencias a procesar. Con esto se consigue mejor eficiencia en el proceso, ya que no se tienen que actualizar una cantidad determinada de registros, sino que se actualiza solo el registro que indica en qué posición del anillo está actualmente apuntando.

Ambos métodos se pueden representar con la Fig. ??.

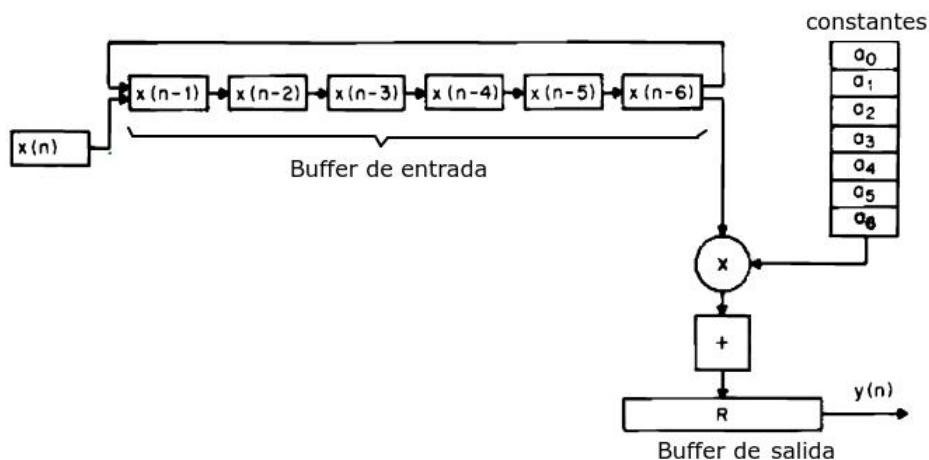


Figura 11.3: Representación del algoritmo de un filtro FIR

Si bien el módulo ADC tiene como características de soportar casi 2 MSPS, intervienen otros factores que hacen que se reduzcan la tasa de muestreo. El factor preponderante es la cantidad de ciclos de reloj que lleva el procesamiento del algoritmo del filtro, compuestos por multiplicaciones, acumulaciones, actualización de registros; este tiempo se incrementa proporcional a la cantidad de términos que tenga el filtro a procesar. Cuanto más sea el orden del filtro, mayor cantidad de términos tendrá y mayor cantidad de operaciones computacionales tendrá que ejecutar.

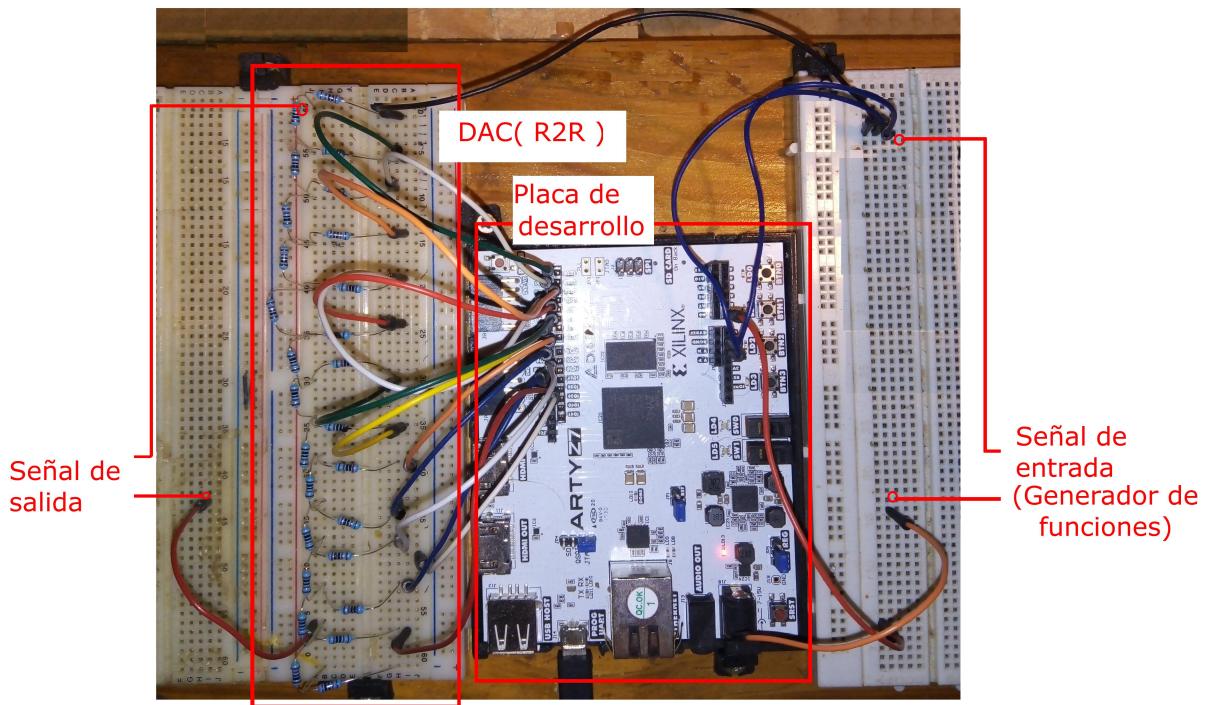


Figura 11.4: Montaje del sistema embebido con FPGA

#### Implementación realizada utilizando matriz de puertas lógicas programables, FPGA

El montaje completo consiste en la utilización de una placa de desarrollo FPGA y un circuito R2R, Fig. ???. La placa de desarrollo utilizada pertenece a la serie Arty Z7, fabricada por Digilent, contiene un chip programable de la linea Zynq-7000 de Xilinx denominado Arty Z7-20 XCZ020-1CLG400C. Este chip trae las siguientes capacidades:

1- Un ADC integrado de 12bits y de 1MSPS, tiene entrada diferenciales. La placa deja accesible 6 puertos analógicos de forma diferencial y 4 mediante una adaptación que convierte la entrada a unipolar. EL voltaje máximo de entrada en modo diferencial es de 1Vpp, y para el modo unipolar es de 0 de 3.3V.

2 - El integrado trae otros módulos, como el generador de señal de reloj, microprocesadores, etc. Entre estos el integrado contiene una FPGA compuesto de 53200 LUTs y 106400 Flip-Flops.

En cuanto a la entrada analógica utilizada, se usó las entradas unipolares, Fig. ???. Ésta tiene una frecuencia máxima de entrada de 100 MHz.

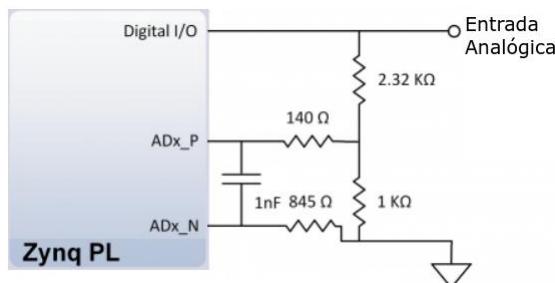


Figura 11.5: Circuito de entrada analógica unipolar incluido en la placa embebida utilizada

Esta placa de desarrollo, no trae el modulo de conversión digital a analógico, para compensar, la empresa que manufactura la placa sugiere la utilización de un R2R, Fig. ???.

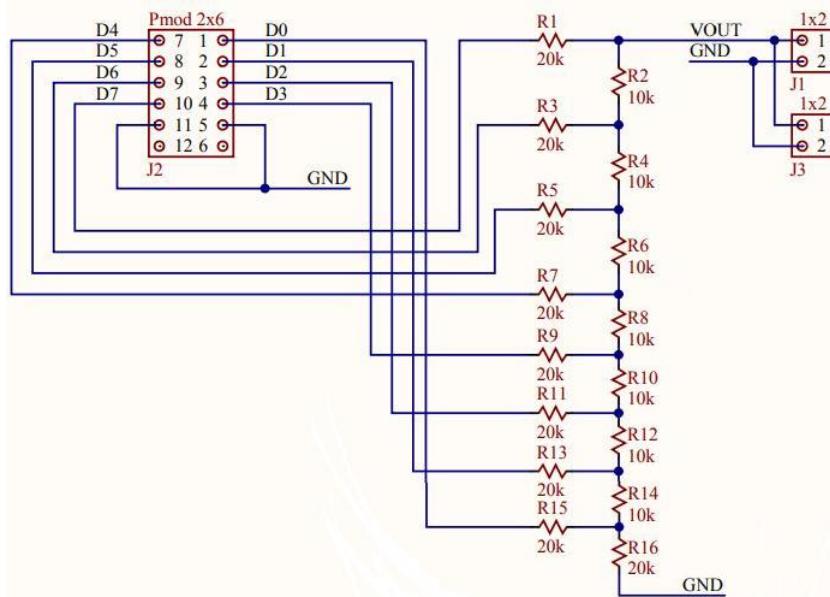


Figura 11.6: Circuito R2R sugerido por la empresa que manufactura la placa embebida

Al igual que la implementación realizada con microcontrolador, la arquitectura del filtro se puede resumir en la siguiente Fig. ???. En la Fig. ?? muestra de forma abreviada la construcción de un filtro.

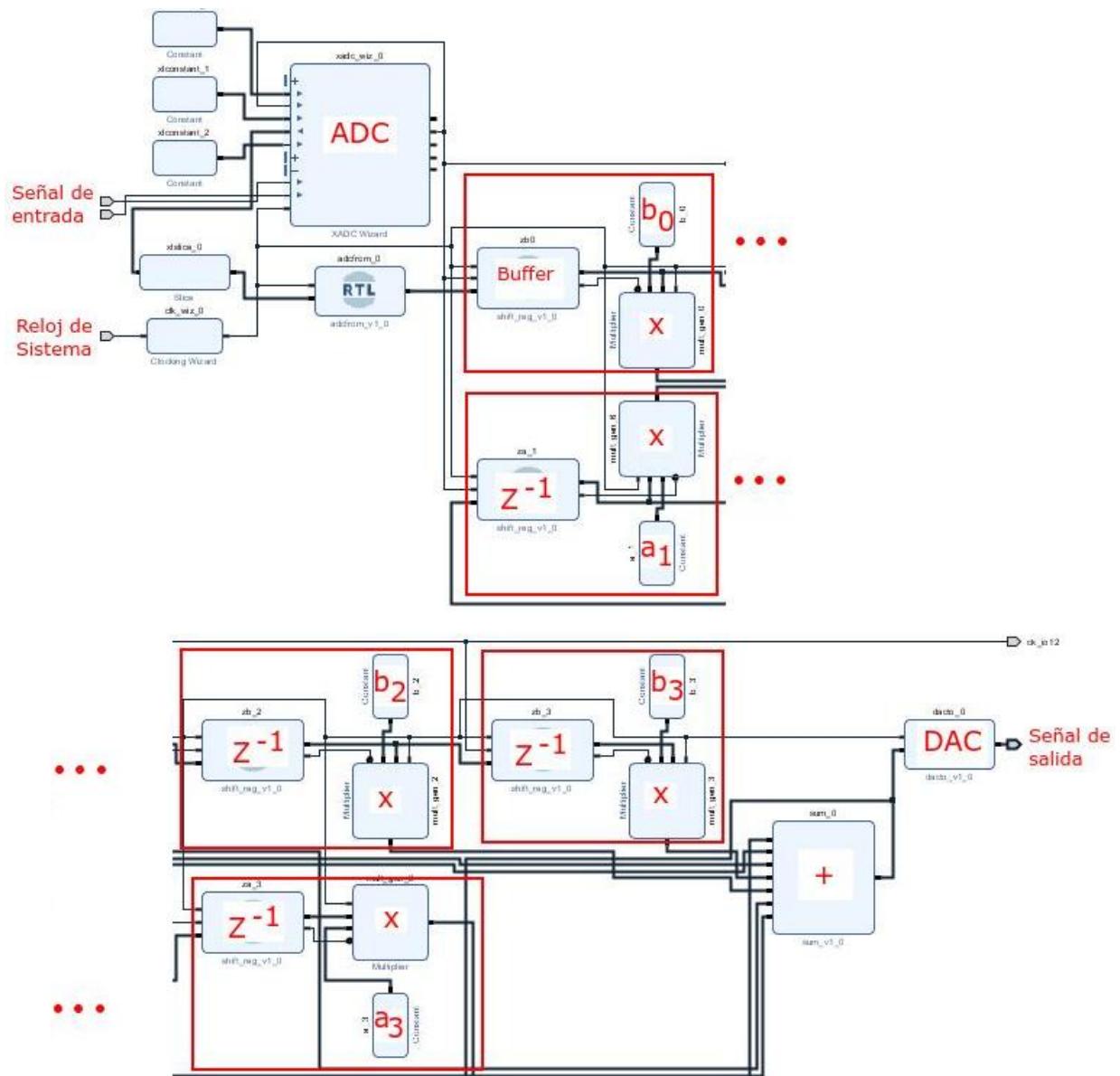


Figura 11.7: Representación de filtro realizado con FPGA mediante los bloques construidos para tal fin

## 11.2. Especificaciones de los filtros IIR

### 11.2.0.1. Filtro de Butterworth (microcontrolador)

En esta sección se muestra los resultados los dos filtros realizados, comentados en la sección anterior, y que se resumen en:

- Por medio de software, ejecutado por un microcontrolador, el cual fue representado en la Fig. ??.
- Por medio de FPGA, esquematizados en la Fig. ??.

### 11.2.1. Filtros pasa bajo

#### 11.2.1.1. Filtro de Butterworth (microcontrolador)

##### Especificaciones del filtro

Tipo Hardware	Pasa bajo Microcontrolador	Butterworth STM32F407	IIR
$f_m$	22418	Hz	Frecuencia de muestreo
$f_c$	1000	Hz	Frecuencia de corte (3dB)
$f_p$	750	Hz	Frecuencia en la banda de paso
$A_p$	0.3	dB	Atenuación en la frecuencia de banda de paso
$f_s$	1250	Hz	Frecuencia en la banda de rechazo
$A_s$	15	dB	Atenuación en la frecuencia banda de rechazo, $f_s$
$N$	8		Orden del filtro necesario

##### Función de transferencia del filtro analógico

$$H(s) = \frac{1,3354e + 42}{5,4976e + 11 s^8 + 1,7706e + 16 s^7 + 2,8514e + 20 s^6 + 2,9794e + 24 s^5 + 2,2013e + 28 s^4 + 1,1763e + 32 s^3 + 4,4445e + 35 s^2 + 1,0896e + 39 s + 1,3357e + 42}$$

##### Función de transferencia del filtro digital

$$H(z) = \frac{7,2639e - 08 z^8 + 5,8111e - 07 z^7 + 2,0339e - 06 z^6 + 4,0678e - 06 z^5 + 5,0847e - 06 z^4 + 4,0678e - 06 z^3 + 2,0339e - 06 z^2 + 5,8111e - 07 z + 7,2639e - 08}{z^8 - 6,5731 z^7 + 19,0104 z^6 - 31,5831 z^5 + 32,9540 z^4 - 22,1061 z^3 + 9,3076 z^2 - 2,2483 z + 0,2385}$$

##### Respuesta en frecuencia

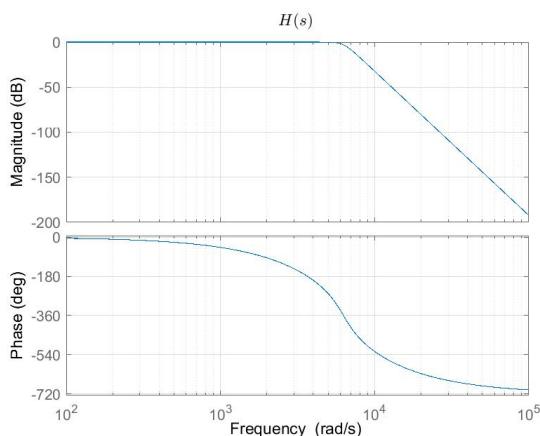


Figura 11.8: Filtro analógico

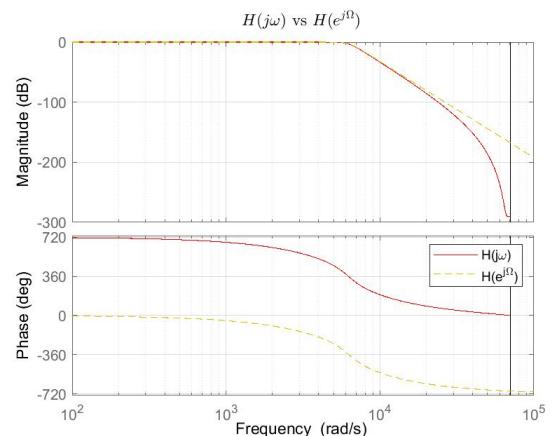


Figura 11.9: Filtro digital

Diagrama de Polos y ceros

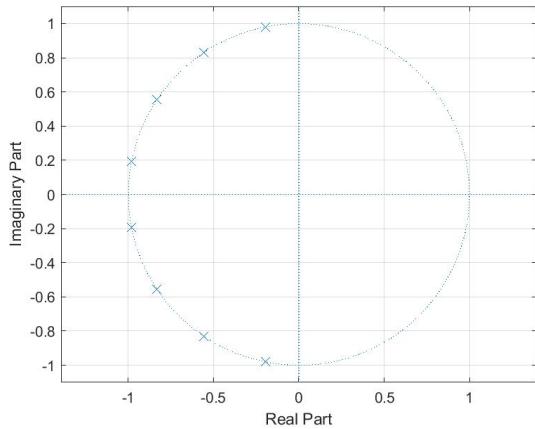


Figura 11.10: Filtro analógico

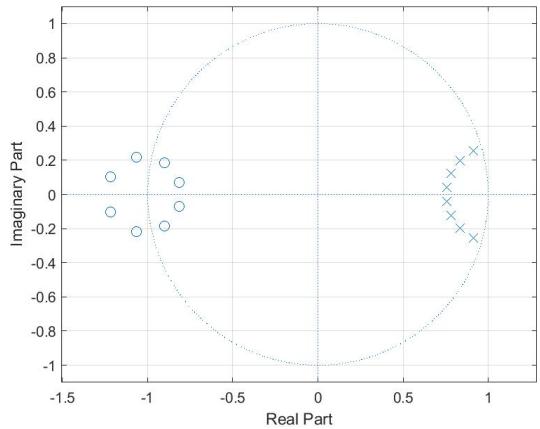


Figura 11.11: Filtro digital

Coeficientes preparados para el microcontrolador

```

1 //Filtro IIR - Butterworth - Pasa Bajo - bilineal (matlab)
2 //Orden: 8
3 //Frecuencia de Muestreo: 22418 (Hz)
4 #define ORDEN 8
5 double coef_x []= {0.00000007,0.00000058,0.00000203,0.00000407,0.00000508,
6     0.00000407,0.00000203,0.00000058,0.00000007};
7 double coef_y []= {-1.00000000,6.57310282,-19.01044364,31.58305647,-32.95401793,
8     22.10607950,-9.30763126,2.24834098,-0.23850553};
```

### 11.2.1.2. Filtro de Chebyshev (microcontrolador)

Datos del filtro

Tipo Hardware	Pasa bajo Microcontrolador	Chebyshev STM32F407	IIR
$f_m$	12760	Hz	Frecuencia de muestreo
$f_c$	1000	Hz	Frecuencia de corte (3dB)
$f_p$	950	Hz	Frecuencia en la banda de paso
$A_p$	0.1	dB	Atenuación en la banda de paso
$f_s$	1150	Hz	Frecuencia en la banda de rechazo
$A_s$	15	dB	Atenuación en la banda de rechazo
$N$	8		Orden del filtro necesario

Función de transferencia del filtro analógico

$$H(s) = \frac{5,5458e + 28}{s^8 + 9,5337e + 03 s^7 + 1,0979e + 08 s^6 + 6,5039e + 11 s^5 + 3,5376e + 15 s^4 + 1,2674e + 19 s^3 + 3,5510e + 22 s^2 + 6,1638e + 25 s + 5,5458e + 28}$$

Función de transferencia del filtro digital

$$H(z) = \frac{2,6626e - 09 z^8 + 2,1301e - 08 z^7 + 7,4552e - 08 z^6 + 1,4910e - 07 z^5 + 1,8638e - 07 z^4 + 1,4910e - 07 z^3 + 7,4552e - 08 z^2 + 2,1301e - 08 z + 2,6626e - 09}{z^8 - 7,4552 z^7 + 24,4448 z^6 - 46,0364 z^5 + 54,4599 z^4 - 41,4355 z^3 + 19,7995 z^2 - 5,4322 z + 0,6551}$$

Respuesta en frecuencia

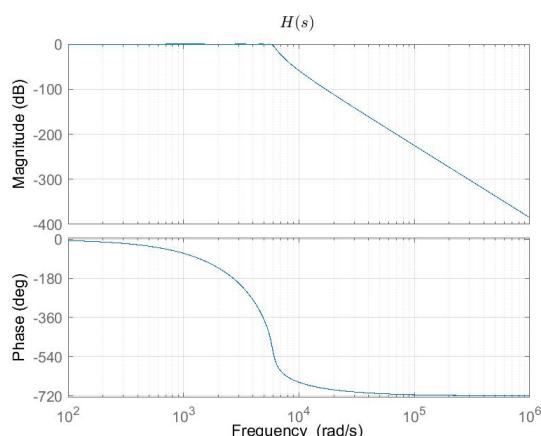


Figura 11.12: Filtro analógico

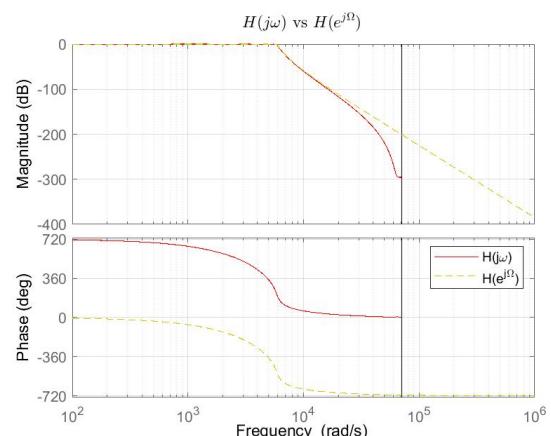


Figura 11.13: Filtro digital

Diagrama de Polos y ceros

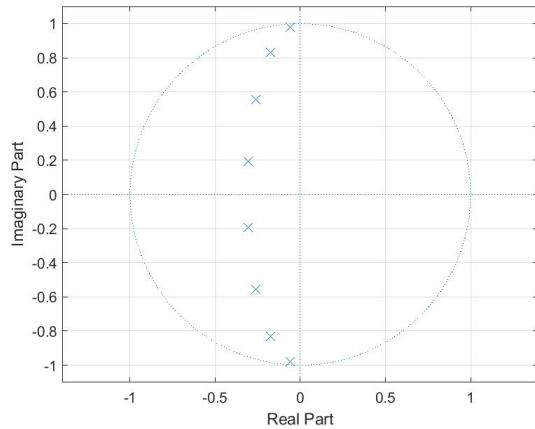


Figura 11.14: Filtro Analógico

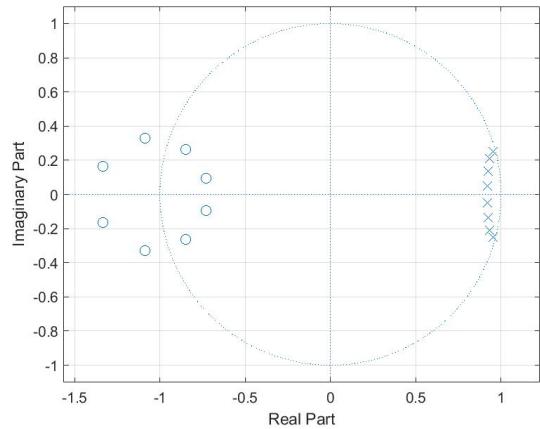


Figura 11.15: Filtro Digital

Coeficientes preparados para el microcontrolador

```

1 //Filtro IIR - Chebyshev - Pasa Bajo - bilineal (matlab)
2 //Orden: 8
3 //Frecuencia de Muestreo: 22418 (Hz)
4 #define ORDEN 8
5 double coef_x []= {0.00000000,0.00000002,0.00000007,0.00000015,0.00000019,
6     0.00000015,0.00000007,0.00000002,0.00000000};
7 double coef_y []= {-1.00000000,7.45524989,-24.44480967,46.03641080,-54.45992407,
8     41.43552867,-19.79953573,5.43221684,-0.65513740};
```

### 11.2.1.3. Filtro de Butterworth (FPGA)

Datos del filtro

Tipo Hardware	Pasa bajo FPGA	Butterworth	IIR
$f_m$	150000	Hz	Frecuencia de muestreo
$f_c$	1000	Hz	Frecuencia de corte (3dB)
$f_p$	750	Hz	Frecuencia en la banda de paso
$A_p$	0.5	dB	Atenuación en la banda de paso
$f_s$	6380	Hz	Frecuencia en la banda de rechazo
$A_s$	10	dB	Atenuación en la banda de rechazo
$N$	8	dB	Orden del filtro necesario

Función de transferencia del filtro analógico

$$H(s) = \frac{1,1332e + 27}{1,7592e + 13 s^3 + 1,4104e + 18 s^2 + 5,6538e + 22 s + 1,1332e + 27}$$

Función de transferencia del filtro digital

$$H(z) = \frac{0,0018 z^3 + 0,0055 z^2 + 0,0055 z + 0,0018}{z^3 - 2,4701 z^2 + 2,0717 z - 0,5869}$$

Respuesta en frecuencia

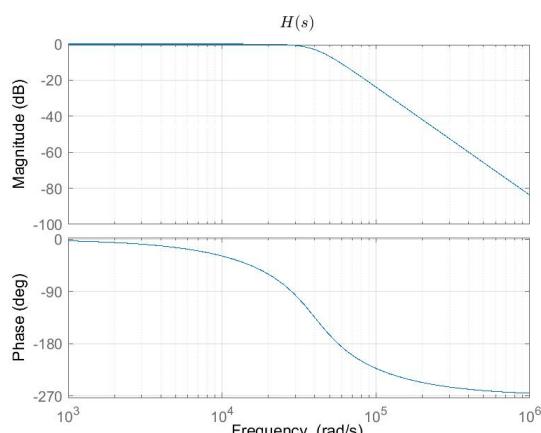


Figura 11.16: Filtro analógico

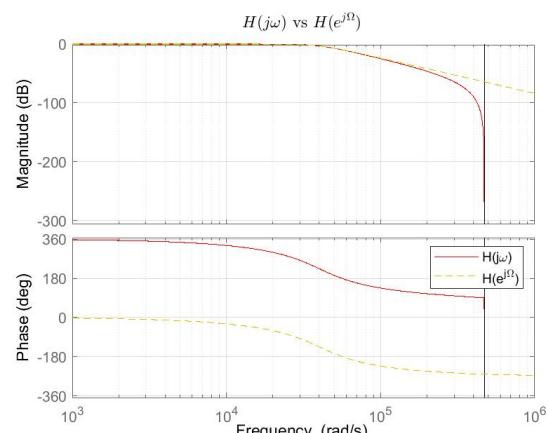


Figura 11.17: Filtro digital

### Diagrama de Polos y ceros

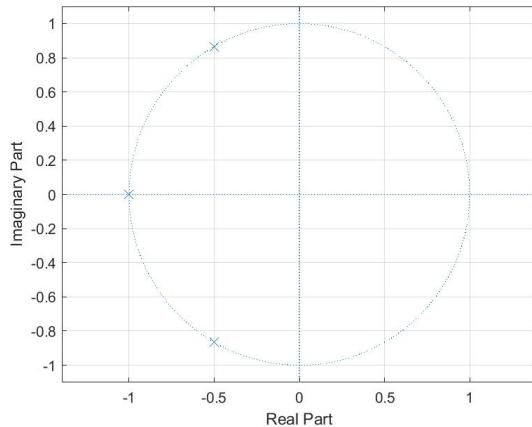


Figura 11.18: Filtro Analógico

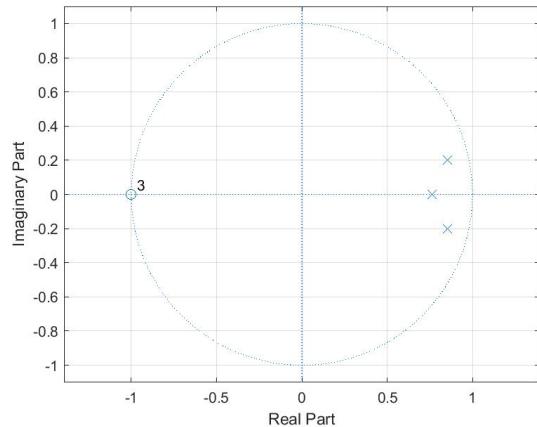


Figura 11.19: Filtro Digital

### Coeficientes preparados para la FPGA

```

1 //Filtro IIR - Pasa Bajo - bilineal(matlab)
2 //Orden: 3
3 //Frecuencia de Muestreo: 150000 (Hz)
4 assign dx0=64'0x000000000077C7B3;assign dx1=64'0x000000001675719;assign dx2=64'0x0000000001675719
  ;assign dx3=64'0x000000000077C7B3;
5
6 assign dy1=64'0x00000002785B8840;assign dy2=64'0xFFFFFFFDEDA83EAA;assign dy3=64'0x00000000963DFB80
  ;
7

```

### 11.2.2. Filtros pasa banda

#### 11.2.2.1. Filtro de Butterworth (microcontrolador)

##### Datos del filtro

Tipo Hardware	Pasa banda Microcontrolador	Butterworth STM32F407	IIR
$f_m$	22418	Hz	Frecuencia de muestreo
$f_0$	1326	Hz	Frecuencia central
$B_1$	865	Hz	Ancho de banda de paso (frecuencia de borde)
$B_p$	625	dB	Ancho de banda de paso
$A_p$	0.3	Hz	Atenuación en la banda de paso
$B_s$	2652	dB	Ancho de banda suprimida
$A_s$	10	dB	Atenuación
$N$	5		Orden del filtro necesario

##### Función de transferencia del filtro analógico

$$H(s) = \frac{1,6137e + 59 s^5}{3,4028e + 40 s^{10} + 5,9847e + 44 s^9 + 1,7073e + 49 s^8 + 1,9477e + 53 s^7 + 2,8316e + 57 s^6 + 2,1434e + 61 s^5 + 1,9655e + 65 s^4 + 9,3848e + 68 s^3 + 5,7102e + 72 s^2 + 1,3894}$$

##### Función de transferencia del filtro digital

$$H(z) = \frac{-1,5902e - 05 z^{10} + 3,5527e - 15 z^9 + 7,9512e - 05 z^8 + 7,1054e - 14 z^7 - 1,5902e - 04 z^6 + 1,1369e - 13 z^5 + 1,5902e - 04 z^4 + 2,1316e - 14 z^3 - 7,9512e - 05 z^2 + 1}{z^{10} - 8,6068 z^9 + 33,9121 z^8 - 80,4976 z^7 + 127,4262 z^6 - 140,5253 z^5 + 109,3286 z^4 - 59,2574 z^3 + 21,4202 z^2 - 4,6652 z + 0,4652}$$

##### Respuesta en frecuencia

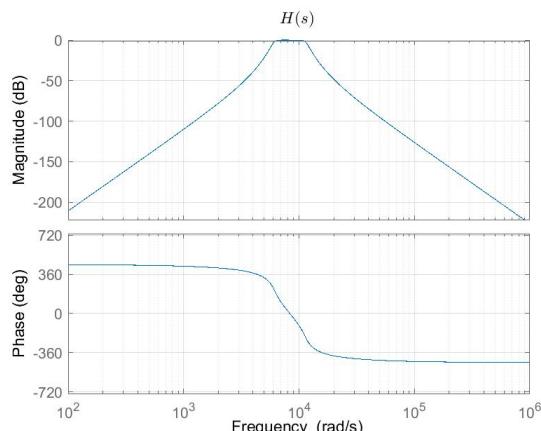


Figura 11.20: Filtro analógico

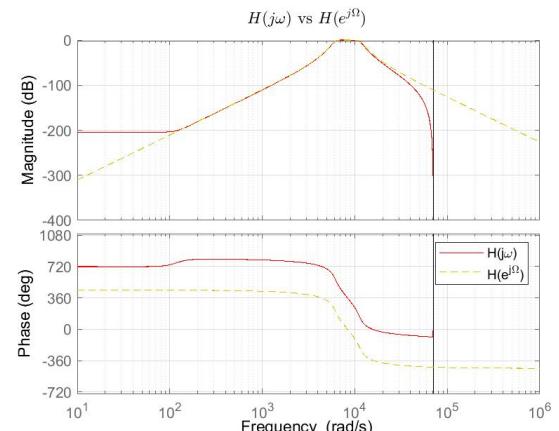


Figura 11.21: Filtro digital

Diagrama de Polos y ceros

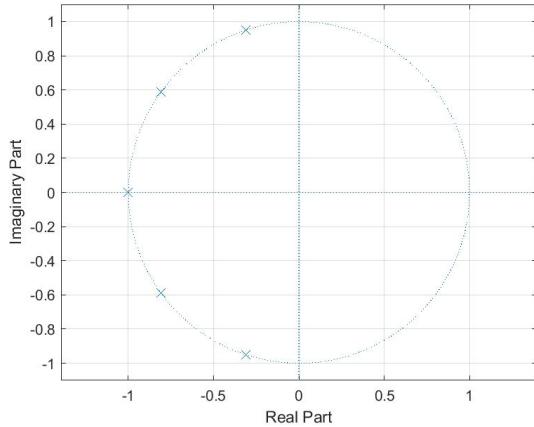


Figura 11.22: Filtro Analógico

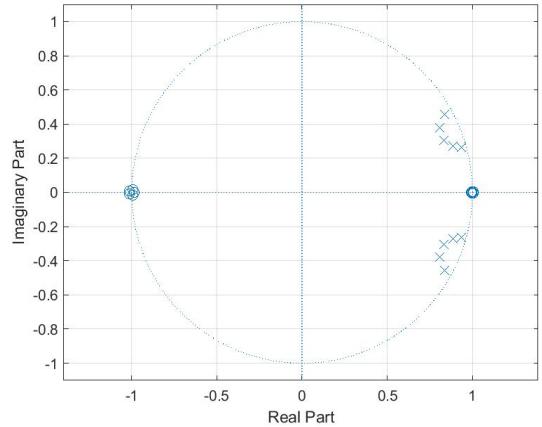


Figura 11.23: Filtro Digital

Coefficientes preparados para el microcontrolador

```

1 //Filtro IIR - Butterworth - Pasa Banda - bilineal(matlab)
2 //Orden: 5
3 //Frecuencia de Muestreo: 22418 (Hz)
4 #define ORDEN 10
5 double coef_x []= {0.00001590,-0.00000000,-0.00007951,-0.00000000,0.00015902,
6 -0.00000000,-0.00015902,-0.00000000,0.00007951,-0.00000000,
7 -0.00001590};
8 double coef_y []= {-1.00000000,8.60684135,-33.91205680,80.49757056,-127.42624846,
9 140.52526259,-109.32857281,59.25740441,-21.42019228,4.66518510,
10 -0.46522603};
```

### 11.2.3. Filtros pasa alto

#### 11.2.3.1. Filtro de Butterworth (microcontrolador)

##### Datos del filtro

Tipo Hardware	Pasa alto Microcontrolador	Butterworth STM32F407	IIR
$f_m$	22418	Hz	Frecuencia de muestreo
$f_c$	1000	Hz	Frecuencia de corte (3dB)
$f_p$	750	Hz	Frecuencia en la banda de paso
$A_p$	0.3	dB	Atenuación en la banda de paso
$f_s$	1250	Hz	Frecuencia en la banda de rechazo
$A_s$	15	dB	Atenuación en la banda de rechazo
$N$	8		Orden del filtro necesario

##### Función de transferencia del filtro analógico

$$H(s) = \frac{4,5036e + 15 s^8}{4,5046e + 15 s^8 + 1,4508e + 20 s^7 + 2,3361e + 24 s^6 + 2,4408e + 28 s^5 + 1,8033e + 32 s^4 + 9,6355e + 35 s^3 + 3,6405e + 39 s^2 + 8,9248e + 42 s + 1,0940e + 46}$$

##### Función de transferencia del filtro digital

$$H(z) = \frac{0,4883 z^8 - 3,9061 z^7 + 13,6714 z^6 - 27,3428 z^5 + 34,1785 z^4 - 27,3428 z^3 + 13,6714 z^2 - 3,9061 z + 0,4883}{z^8 - 6,5732 z^7 + 19,0108 z^6 - 31,5838 z^5 + 32,9547 z^4 - 22,1065 z^3 + 9,3077 z^2 - 2,2483 z + 0,2385}$$

##### Respuesta en frecuencia

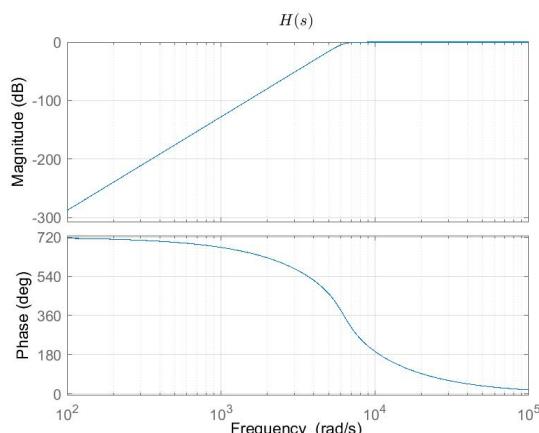


Figura 11.24: Filtro analógico

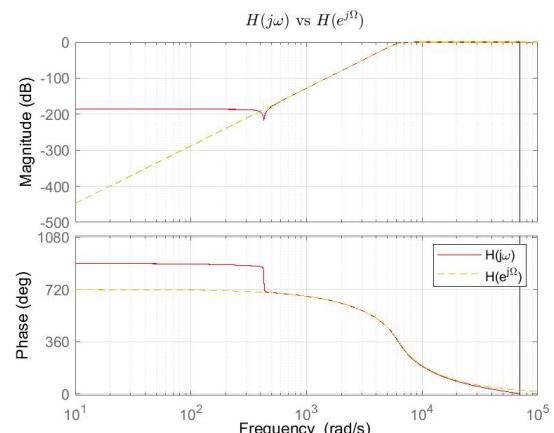


Figura 11.25: Filtro digital

Diagrama de Polos y ceros

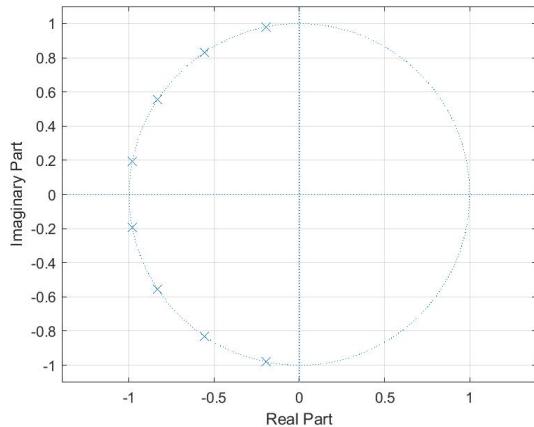


Figura 11.26: Filtro Analógico

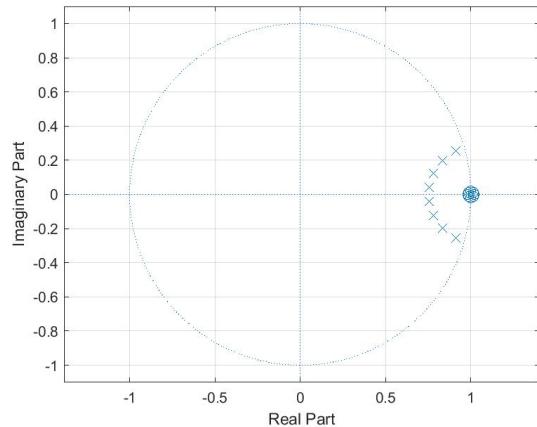


Figura 11.27: Filtro Digital

Coeficientes preparados para el microcontrolador

```

1 //Filtro IIR - Butterworth - Pasa Alto - bilineal (matlab)
2 //Orden: 8
3 //Frecuencia de Muestreo: 22418 (Hz)
4 #define ORDEN 8
5 double coef_x []= {0.48826428,-3.90611423,13.67139981,-27.34279962,34.17849952,
6 -27.34279962,13.67139981,-3.90611423,0.48826428};
7 double coef_y []= {-1.00000000,6.57317134,-19.01078560,31.58375452,-32.95474459,
8 22.10645539,-9.30768750,2.24831447,-0.23849661};
```

## 11.3. Especificaciones de los filtros FIR

### 11.3.1. Filtro pasa bajo

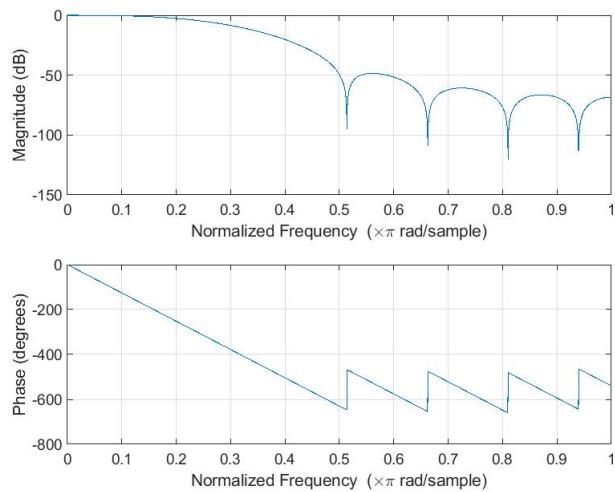
Datos del filtro:

Tipo Hardware	Pasa bajo Microcontrolador	FIR STM32F407	
$f_m$	12760	Hz	Frecuencia de muestreo
$f_c$	1500	Hz	Frecuencia de corte (3dB)
$N$	8	dB	Orden del filtro necesario
Ventana	Hamming		

Función de transferencia del filtro digital

$$H(z) = -0,0014 z^{14} - 0,0063 z^{13} - 0,0141 z^{12} - 0,0077 z^{11} + 0,0396 z^{10} + 0,1308 z^9 + 0,2264 z^8 + 0,2676 z^7 + 0,2264 z^6 + 0,1308 z^5 + 0,0396 z^4 - 0,0077 z^3 - 0,0141 z^2 - 0,0063 z - 0,0014$$

Respuesta en frecuencia



Coeficientes preparados para el microcontrolador

```

1 //Filtro FIR -pasa_bajo -
2 //Orden: 15
3 //Frecuencia de Muestreo: 22418 (Hz)
4 #define ORDEN 14
5 double coef_x []= {-0.00140802,-0.00629594,-0.01408120,-0.00765779,0.03955279,
6     0.13078253,0.22639634,0.26764207,0.22639634,0.13078253,
7     0.03955279,-0.00765779,-0.01408120,-0.00629594,-0.00140802
8     };
9 double coef_y []= {-1.00000000,0.00000000,0.00000000,0.00000000,0.00000000,
10    0.00000000,0.00000000,0.00000000,0.00000000,0.00000000,
11    0.00000000,0.00000000,0.00000000,0.00000000,0.00000000
12    };

```

### 11.3.2. Filtro pasa alto

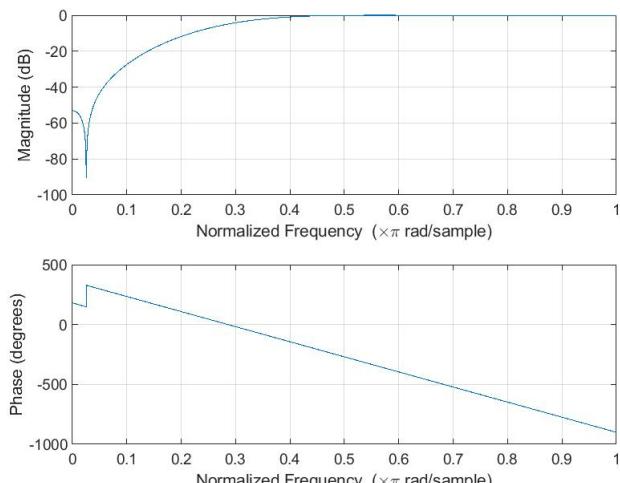
Datos del filtro

Tipo Hardware	Pasa alto Microcontrolador	FIR STM32F407	
fm	12760	Hz	Frecuencia de muestreo
fc	1500	Hz	Frecuencia de muestreo
N	8	dB	Orden del filtro necesario
Ventana	Hamming		

Función de transferencia del filtro digital

$$H(z) = 0,0014 z^{14} + 0,0063 z^{13} + 0,0141 z^{12} + 0,0077 z^{11} - 0,0396 z^{10} - 0,1308 z^9 - 0,2264 z^8 + 0,7324 z^7 - 0,2264 z^6 - 0,1308 z^5 - 0,0396 z^4 + 0,0077 z^3 + 0,0141 z^2 + 0,0063 z + 0,0014$$

Respuesta en frecuencia



Coeficientes preparados para el microcontrolador

```

1 //Filtro FIR -pasa_alto -
2 //Orden: 15
3 //Frecuencia de Muestreo: 22418 (Hz)
4 #define ORDEN 14
5 double coef_x []= {0.00140802,0.00629594,0.01408120,0.00765779,-0.03955279,
6     -0.13078253,-0.22639634,0.73235793,-0.22639634,-0.13078253,
7     -0.03955279,0.00765779,0.01408120,0.00629594,0.00140802
8     };
9 double coef_y []= {-1.00000000,0.00000000,0.00000000,0.00000000,0.00000000,
10     0.00000000,0.00000000,0.00000000,0.00000000,0.00000000,
11     0.00000000,0.00000000,0.00000000,0.00000000,0.00000000
12     };

```

# **12 Filtro Activo**

## **12.1. Comentarios sobre la bibliografía**

Paul Bildstein. Filtros Activos

S.A. Pactitis. Active Filters: Theory and Design

Wai-Kai Chen. Passive, Active, and Digital Filters