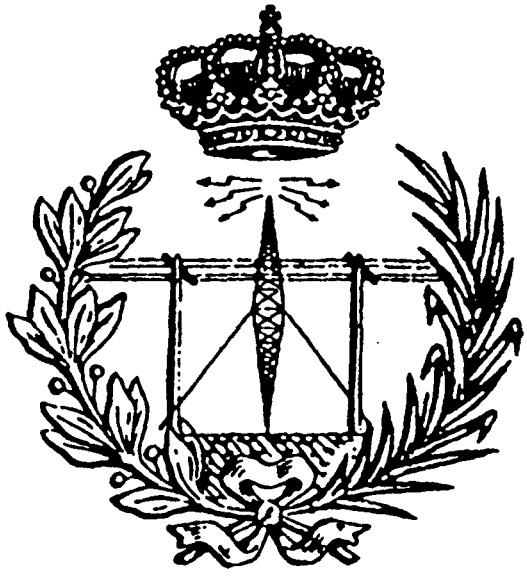
Universidad de las Palmas de Gran Canaria

E.U.I.T.TELECOMUNICACIÓN



TRABAJO FIN DE CARRERA

TITULO: **SISTEMA DE DESARROLLO BASADO EN MATLAB PARA EL DISEÑO DE FILTROS DIGITALES EN EL DSP TMDS3200051.**

ESPECIALIDAD: **TELEFONÍA Y TRANSMISIÓN DE DATOS.**

AUTOR: **RAFAEL SOCAS GUTIÉRREZ.**

TUTOR: **JUAN RUIZ ALZOLA.**

FECHA: **DICIEMBRE 1995.**

TUTOR: AUTOR:

PRESIDENTE: SECRETARIO: VOCAL: CALIFICACIÓN:

ÍNDICE:

página

CAPITULO 1: OBJETIVOS.

Los objetivos planteados al principio del presente Trabajo fin de carrera fueron:

1.-Diseñar un software que:

a) Ayude a estudiar y a diseñar filtros digitales.

b) Auxilie en la implementación de dichos filtros en el DSP TMDS3200051.

c) Simule el comportamiento de los filtros diseñados , permitiendo comparar su funcionamiento ideal con el que realmente van a tener en el DSP.

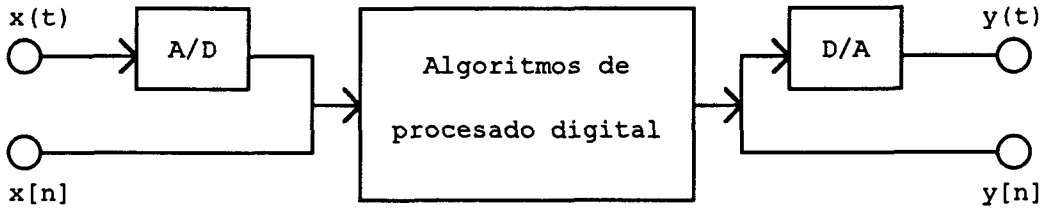
2.-Implementar varios filtros en el DSP TMDS3200051 sirviéndose del software desarrollado.

3.-Implementar un sistema de procesado en tiempo real de señal de electrocardiograma basado en el DSP TMDS3200051 . En concreto , elaborar un "detector de complejo QRS que permitiese evaluar el ritmo cardiaco en tiempo real"

CAPITULO 2: ANTECEDENTES.

2.1: INTRODUCCIÓN AL FILTRADO DIGITAL.

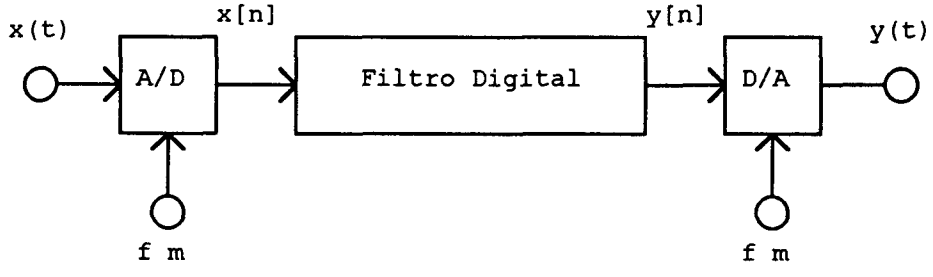
Con la aparición de los sistemas digitales aparece un nuevo concepto en el tratamiento de señales que es el procesado en tiempo discreto o procesado digital el cual se ajusta al siguiente diagrama:



Como se observa el diagrama anterior el procesado digital permite tratar tanto señales de naturaleza digital (x[n]) como de analógica (x(t)) que han sido previamente tratadas con un sistema conversor analógico-digital.

En el presente Trabajo fin de carrera nos vamos a preocupar de procesar señales de tiempo continuo (analógicas) y por tanto vamos a describir las correspondencias que existen entre un procesado digital y la correspondiente respuesta analógica.

Por tanto para nuestro estudio nos vamos a basar en un sistema como el siguiente:



Las operaciones que vamos a realizar son de filtrado , por tanto nos vamos a preocupar de estudiar un subconjunto de los sistemas que procesan señales digitales que son los FILTROS DIGITALES.

El sistema global que buscamos es un filtro analógico (Sistema que deja pasar una frecuencias y elimina el resto), que sea LTI , es decir, lineal e invariante en el tiempo . Para que el sistema analógico sea LTI los conversores A/D y D/A deben cumplir el criterio de Nyquist y el filtro digital debe ser LTI.

Los criterios de linealidad e invarianza temporal para el filtro digital son los siguientes:

Linealidad:
Si el sistema responde de la siguiente manera:

entrada		salida
x1[n]	----------	y1[n]
x2[n]	----------	y2[n]

Si es lineal debe responder:

entrada		salida
ax1[n]+bx2[n]	--------	ay1[n]+by2[n]

Invarianza temporal:
Si el sistema tiene una respuesta:

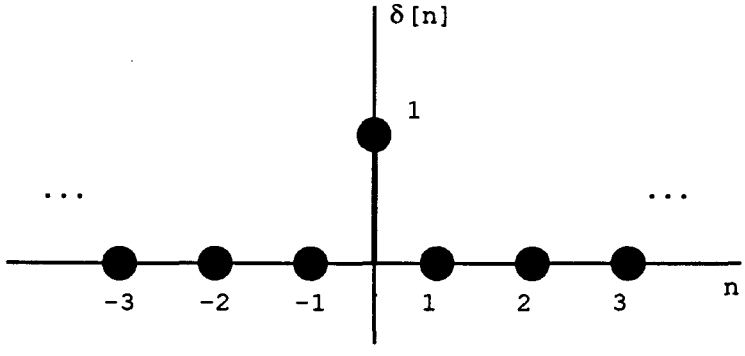
entrada		salida
x[n]	---------	y[n]

Si es invariante temporal nos dará:

entrada		salida
x[n-no]	----------	y[n-no]

La característica antes mencionada de que el sistema sea LTI nos permite caracterizarlo totalmente conociendo solamente la respuesta al impulso δ[n].Veamos el porqué:

El impulso δ[n]tiene la siguiente forma:



Cualquier secuencia x[n] se puede poner en función del impulso δ[n] de la siguiente forma:

$$x[n]=\sum_{k=-\infty}^{+\infty} x[k]\delta[n-k]$$

La salida de un sistema viene dada por la aplicación T{}a la entrada de tal manera que:

$$y[n]=T\{x[n]\}$$

y en términos de $\delta[n]$ nos queda como:

$$y[n] = T\left\{\sum_{k=-\infty}^{+\infty} x[k]\delta[n-k]\right\}$$

pero como el sistema es lineal se puede poner como:

$$y[n] = \sum_{k=-\infty}^{+\infty} x[k]T\{\delta[n-k]\}$$

además es invariante temporal , con lo cual , $T\{\delta[n-k]\}=h[n-k]$ partiendo de la hipótesis de que $T\{\delta[n]\}=h[n]$. La expresión anterior se nos reduce a:

$$\boxed{y[n]=\sum_{k=-\infty}^{+\infty} x[k]h[n-k]=x[n]*h[n]}$$

La operación obtenida se denomina convolución y podemos calcular la salida para cualquier entrada si conocemos la respuesta la impulso h[n].

Si a un sistema LTI lo atacamos con una exponencial compleja de la forma $x[n]=\exp(j\Omega n)$ que vaya desde $-\infty<n<+\infty$ la salida será:

$$y[n] = \sum_{k=-\infty}^{+\infty} h[k]e^{j\Omega(n-k)} = e^{j\Omega n}\left(\sum_{k=-\infty}^{+\infty} h[k]e^{-j\Omega k}\right)$$

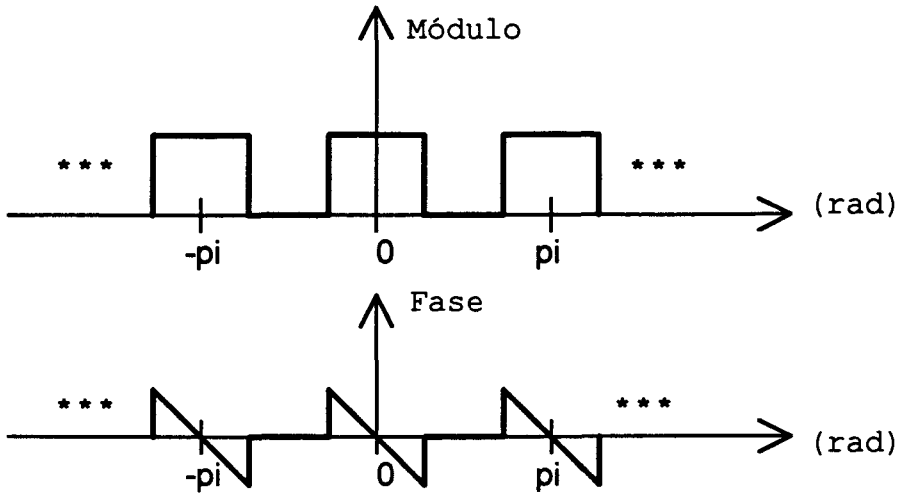
Como observamos en la expresión anterior la salida es la misma exponencial (autofunción) multiplicada por una constante (autovalor).

Este autovalor nos dará como se va comportando el sistema para las diferentes exponenciales complejas , es decir , por cuanto se multiplica cada exponencial compleja en función de la frecuencia de esta. Este autovalor a su vez , en general , es un número complejo que se denomina **función de transferencia del sistema** y que coincide con la transformada de Fourier de h[n].

Podemos escribir entonces:

$$\boxed{H(\Omega)=\sum_{k=-\infty}^{+\infty} h[k]e^{-j\Omega k}}$$

Esta $H(\Omega)$ entre otras propiedades tiene la de ser periódica de periodo 2π y suele representarse su módulo y su fase en función de la Ω y nos puede dar algo de la forma siguiente:



Como caso general de la transformada de Fourier esta la Transformada Z que nos da información adicional en cuanto a estabilidad , causalidad , efectos al trabajar con precisión finita por la movilidad de los polos y ceros , etc.

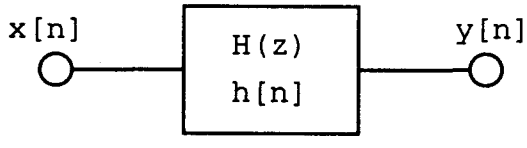
La transformada Z de la respuesta al impulso de un filtro es:

$$H(z) = \sum_{n=-\infty}^{+\infty} h[n] z^{-n}$$

Según la forma de esta función podemos caracterizar el filtro digital en cuestión , a partir de ella podemos ver la respuesta en frecuencia la TF , si el sistema es estable, etc.

Tras esta breve introducción a la teoría de sistemas discretos LTI (filtros digitales), vamos a ver de forma somera como es la forma de los filtros digitales que se van a tratar en el presente Trabajo fin de carrera.

Los filtros LTI son un subconjunto de los sistemas discretos y nos vamos a ocupar de un conjunto aún más reducido que son los LTI descritos por ecuaciones en diferencias. Este tipo de sistemas tienen una ecuación característica como la siguiente:



$$\sum_{k=0}^{N} a_k y[n-k] = \sum_{k=0}^{M} b_k x[n-k]$$

De este tipo de sistema podemos conocer fácilmente la transformada H(z) y por supuesto la transformada H(Ω) y conocer como se comporta el filtro en función de la frecuencia.

La H(z) de este sistema se calcula aplicando transformada Z en ambos miembros obteniendo:

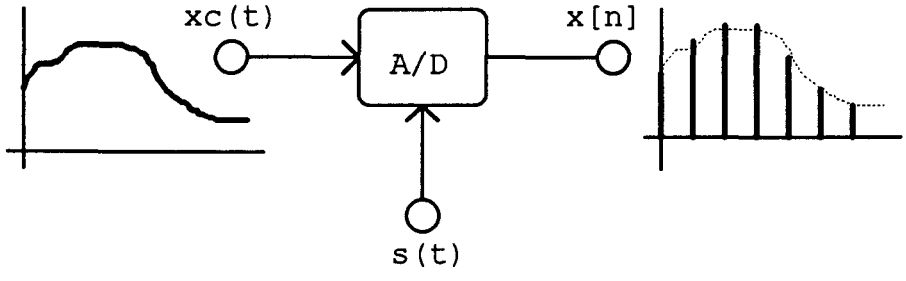
$$H(z)=\frac{Y(z)}{X(z)}=\frac{\sum_{k=0}^{M} b_k z^{-k}}{\sum_{k=0}^{N} a_k z^{-k}}$$

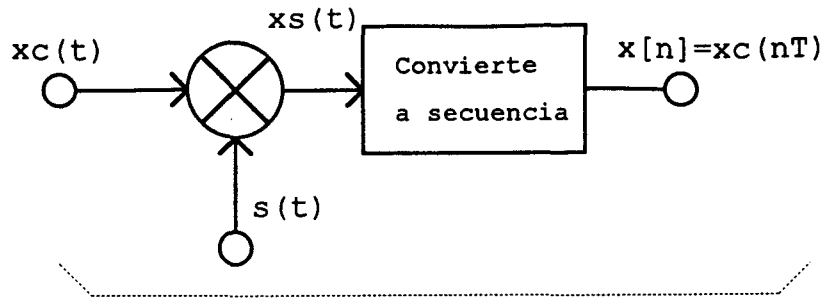
Para obtener la transformada de Fourier , basta con particularizar la transformada Z en el círculo unidad ($z=\exp(j\Omega n)$).

Esto filtros tienen la ventaja de que pueden implementarse tanto sistemas FIR (N=0) como sistemas IIR ($N \neq 0$) en un sistema digital con poco tiempo de proceso y sin necesidad de gran almacenamiento de muestras de señal.

Ya una vez visto someramente lo que son los filtros digitales vamos a explicar qué relaciones hay entre el espectro analógico de la señal de entrada y el espectro digital del filtro implementado como un algoritmo en un sistema de procesado digital de señales.

El primer paso a seguir es conseguir de la señal de tiempo continuo una señal de tiempo discreto. Esto se consigue mediante un muestreo periódico según muestra la figura:





Conversor A/D

Si tomamos un muestreo teórico tendremos lo siguiente:

$$s(t)=\sum_{n=-\infty}^{+\infty}\delta(t-nT)$$

también $xs(t)=xc(t)s(t)$

$$xs(t)=xc(t)\sum_{n=-\infty}^{+\infty}\delta(t-nT)$$

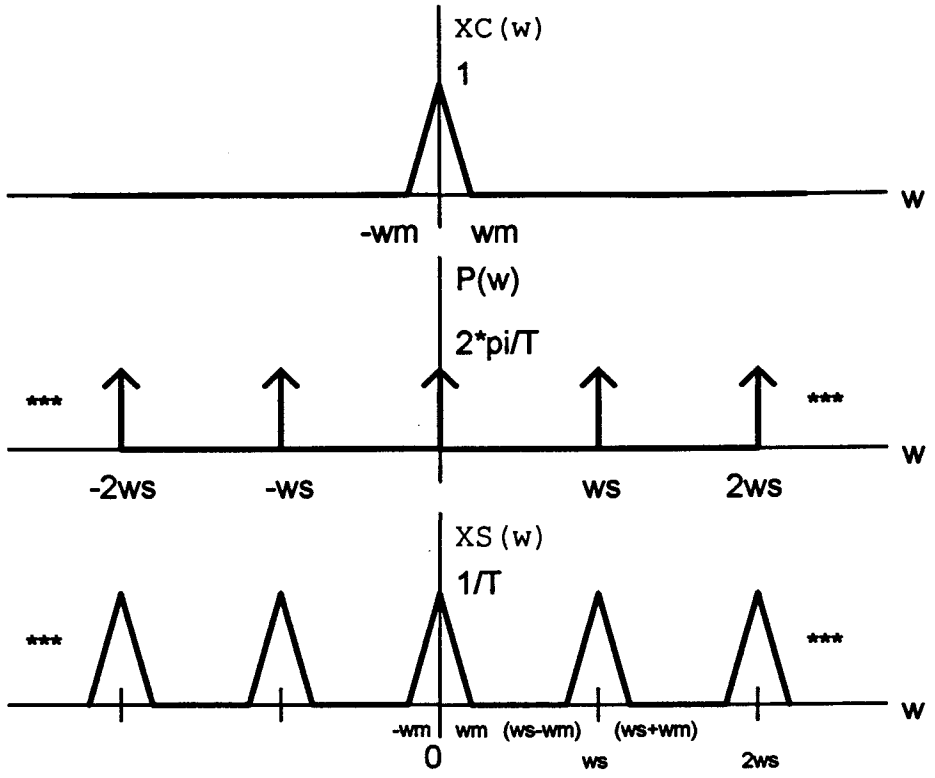
y como la transformada de Fourier de s(t) vale:

$$S(w)=\frac{2\pi}{T}\sum_{k=-\infty}^{+\infty}\delta(w-kws);\text{donde } ws=\frac{2\pi}{T}$$

Con lo que nos sale que:

$$\boxed{XS(w)=\frac{1}{T}\sum_{k=-\infty}^{+\infty}XC(w-kws)}$$

De manera gráfica resulta ser:



Como se observa en la gráfica para que no haya solapamiento se debe cumplir el Teorema de Nyquist , es decir, que ws>2wm.

Como vemos la señal de tiempo continuo muestreada tiene una transformada de Fourier periódica , con lo cual una vez esta señal pase por el convertidor a secuencia ya tendremos la secuencia digital y podremos relacionar el sistema analógico global con el sistema digital.

Volvamos nuevamente al siguiente esquema:

Del diagrama anterior podemos escribir:

$$x[n]=xc(nT)$$

y tomando transformada de Fourier

$$X(\Omega) = \frac{1}{T}\sum_{k=-\infty}^{+\infty} Xc\left(\frac{\Omega}{T} - \frac{2\pi k}{T}\right)$$

Tras la salida del conversor D/A (posee un filtro paso bajo)

Tras la salida del conversor D/A (posee un filtro paso bajo)

$$yr(t) = \sum_{n=-\infty}^{+\infty} y[n] \frac{sen[\pi(t-nT)/T]}{\pi(t-nT)/T}$$

y en el dominio transformado nos queda como:

$$Yr(w) = Hr(w)Y(WT) = \begin{cases} TY(WT) ; |W| < p/T \\ 0 ; \text{resto} \end{cases}$$

También podemos escribir:

$$Y(\Omega) = H(\Omega)X(\Omega)$$

que de forma global:

$$Yr(w) = Hr(w)H(wT)X(wT)$$

y teniendo en cuenta la relación $wT=\Omega$:

$$Yr(w) = Hr(w)H(wT)\frac{1}{T}\sum_{k=-\infty}^{+\infty} Xc(w - \frac{2\pi k}{T})$$

$$Yr(w) = \begin{cases} H(wT)Xc(w) ; |w| < p/T \\ 0 ; |w| \geq p/T \end{cases}$$

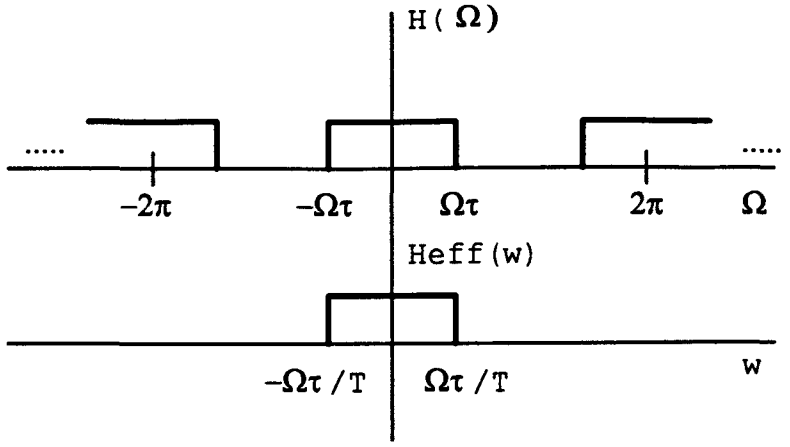
Y podemos encontrar la relación entre el filtro analógico global y el filtro digital:

$$Yr(w) = Heff(w)Xc(w)$$

donde:

$$\boxed{Heff(w) = \begin{cases} H(wT) ; |w| < p/T \\ 0 ; |w| \geq p/T \end{cases}}$$

La relación entre el filtro digital y el filtro analógico global de manera gráfica quedaría como:



Como observamos en el análisis anterior el filtro analógico global del sistema compuesto por unos conversores A/D,D/A y un DSP es el filtro digital particularizado para $w=\Omega/T$ y truncando entre $-\pi$ y $+\pi$. De esta manera tenemos una correspondencia directa entre el filtro analógico que queremos y el filtro digital que se debe programar en el DSP.

Ahora haremos un pequeño repaso de los métodos utilizados para diseñar tanto filtros FIR como IIR . En el presente TFC hemos utilizado el MÉTODO DE LAS VENTANAS para filtros FIR y el MÉTODO DE LA TRANSFORMACIÓN BILINEAL para los IIR.

MÉTODO DE LAS VENTANAS:

Los filtros FIR tienen la ventaja de que se pueden diseñar fácilmente para que tengan fase lineal y por tanto retardo de grupo constante. Esto se consigue si la respuesta h[n] del filtro es simétrica respecto a su centro, analíticamente sería:

Si h[n] es de longitud M debe cumplir:

$$h[n]=\begin{cases} h[M-n]; 0\le n\le M \\ 0; \text{resto} \end{cases}$$

Mediante el método de las ventanas vamos a conseguir que h[n] cumpla la propiedad de simetría y por tanto que tenga fase lineal.

El método de las ventanas está basado en los siguientes puntos:

Especificamos el módulo del filtro ideal que queremos diseñar.

Luego calculamos el hi[n] ideal que le corresponde al filtro ideal especificado anteriormente.

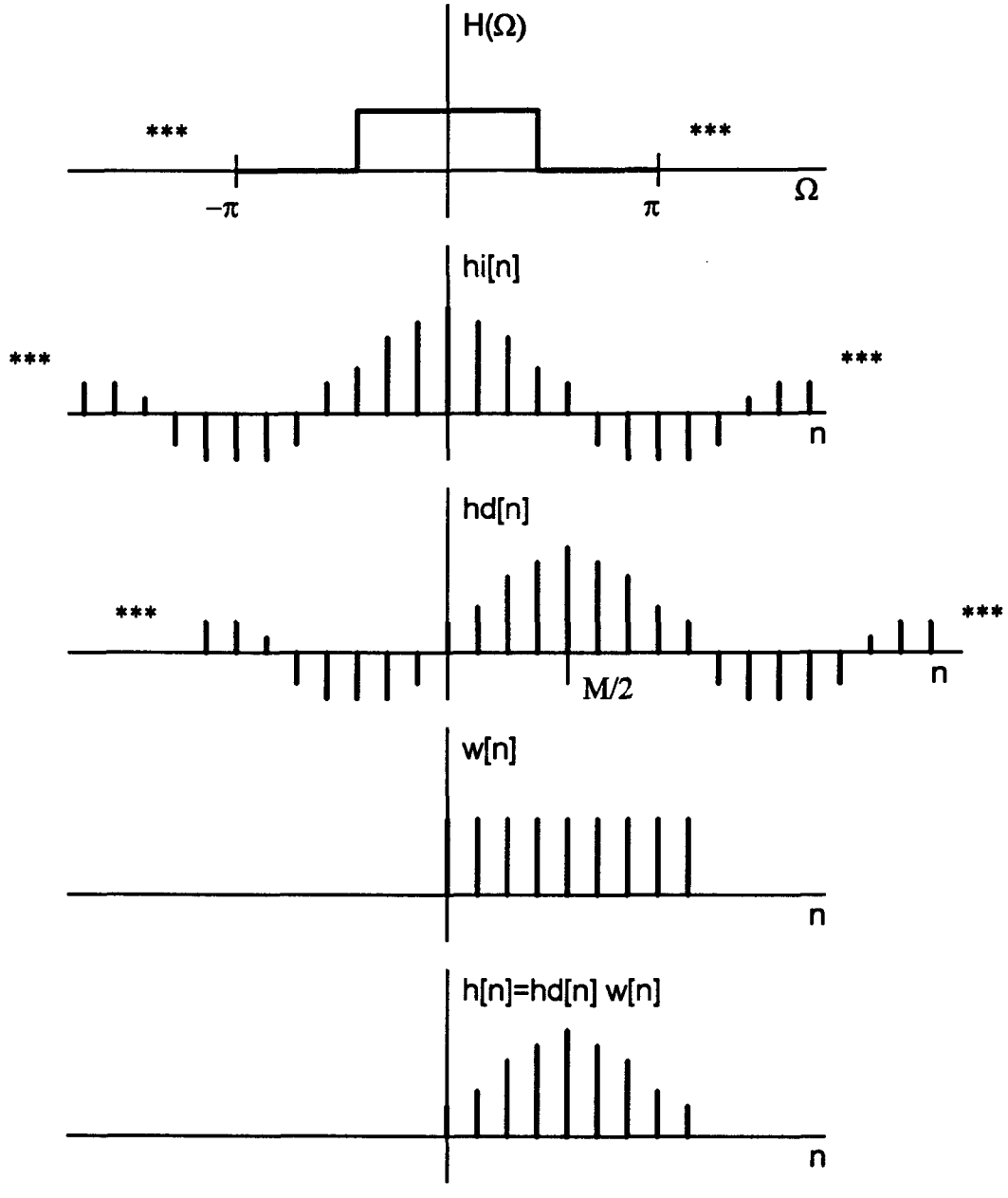
El filtro real tendrá una longitud M por lo que creamos hd[n] que esté centrada en M/2 ,es decir, hd[n]=hi[n-M/2].

Más tarde enventanamos esta hd[n] con una ventana w[n] de longitud M que tiene que cumplir la propiedad de simetría respecto a M/2 con lo cual habremos obtenido el filtro

$$h[n]=hd[n]\ w[n]$$

Puesto que hd[n] es simétrica y w[n] se escoge de tal manera que lo sea respecto a M/2 el h[n] será simétrico y por tanto con fase lineal.

De manera gráfica sería:



Como observamos en las gráficas anteriores al enventanar estamos eliminando términos de la respuesta del filtro ideal (la respuesta del filtro ideal tiene infinitos términos) y por tanto la respuesta en frecuencia se

ajustará mas a la especificación inicial cuanto mayor sea la longitud de la ventana.

Analíticamente la respuesta en frecuencia del filtro que se está diseñando tiene la forma:

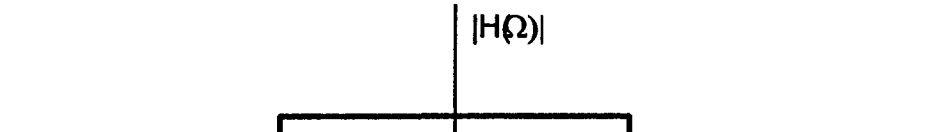
$$h[n] = hd[n]w[n]$$

$$H(W) = \frac{1}{2\pi}\int_{-\pi}^{\pi} Hd(\Theta)W(W-\Theta)d\Theta$$

Como se observa en la ecuación anterior $H(\Omega)$ es la convolución periódica en el dominio frecuencial de $Hd(\Omega)$ con $W(\Omega)$, con lo cual para que $H(\Omega)$ se parezca lo mas posible a $Hd(\Omega)$, $W(\Omega)$ de ser lo más parecido a una $\delta(\Omega)$.

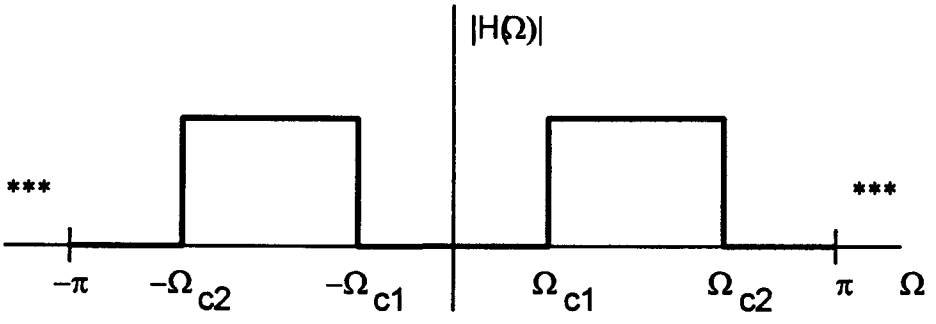
Los tipos de carátulas de especificación que vamos a utilizar para este tipo de filtros son las siguientes:

PASO-BAJO:

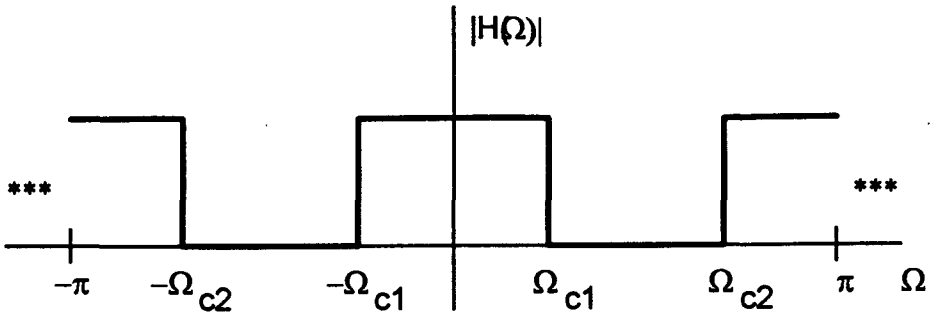


PASO-ALTO:

PASO-BANDA:



BANDA-ELIMINADA:

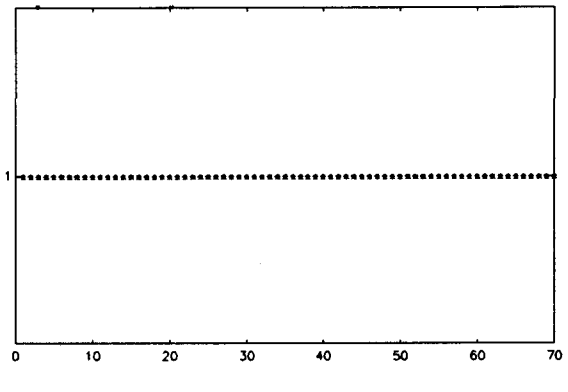


A continuación vamos a mostrar las ventanas más utilizadas para este tipo de diseño de filtros FIR que son las siguientes:

Rectangular

$$w[n]=\begin{cases}1;0\le n\le M\\0;\text{Resto}\end{cases}$$

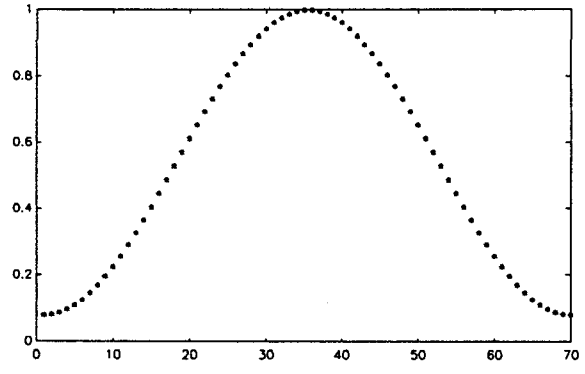
A continuación se representa una w[n] rectangular de M=70.



#Hamming

$$w[n]=\begin{cases}0.54-0.46\cos\left(\dfrac{2\pi n}{M}\right);0\le n\le M\\0;\text{Resto}\end{cases}$$

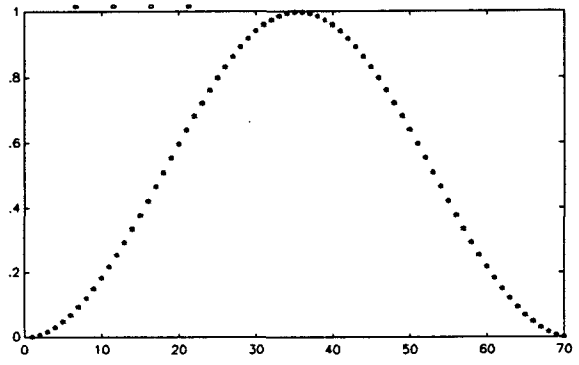
A continuación se representa una w[n] de Hamming donde M=70.



Hanning

$$w[n]=\begin{cases}0.5-0.5\cos\left(\frac{2\pi n}{M}\right);0\le n\le M\\ 0;\text{Resto}\end{cases}$$

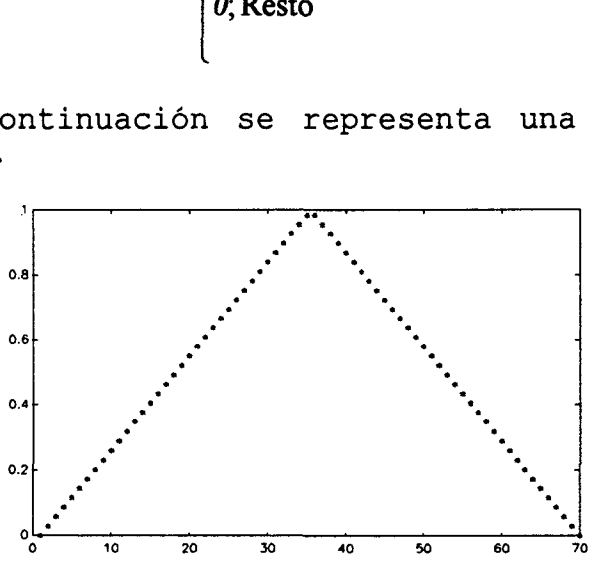
A continuación se representa una w[n] de Hanning donde M=70.



#Bartlett

$$w[n]=\begin{cases}\frac{2n}{M};0\le n\le M/2\\ 2-\frac{2n}{M};M/2<n\le M\\ 0;\text{Resto}\end{cases}$$

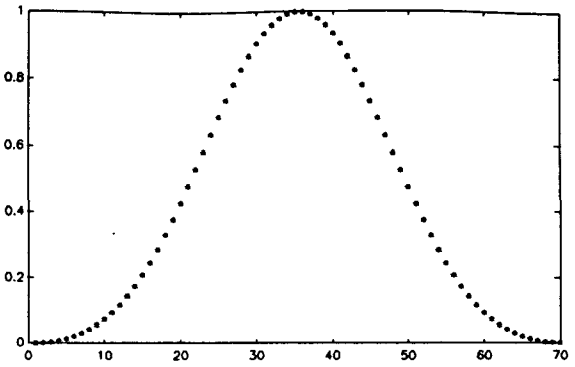
A continuación se representa una w[n] de Bartlett donde M=70.



#Blackman

$$w[n]=\begin{cases}0.42-0.5\cos\left(\frac{2\pi n}{M}\right)+0.08\cos\left(\frac{4\pi n}{M}\right);0\le n\le M\\ 0;\text{Resto}\end{cases}$$

A continuación se representa una w[n] de Blackman donde M=70.



Las ventanas anteriores tienen una transformada de Fourier la cual básicamente es un lóbulo principal con varios lóbulos secundarios, la ventana que nos interesa debe tener el lóbulo principal lo más estrecho posible y lóbulos secundarios que se atenúen rápidamente (nos interesa que sea lo más parecida a una delta de Dirac) aunque según la precisión del diseño podemos coger un tipo de ventana u otro.

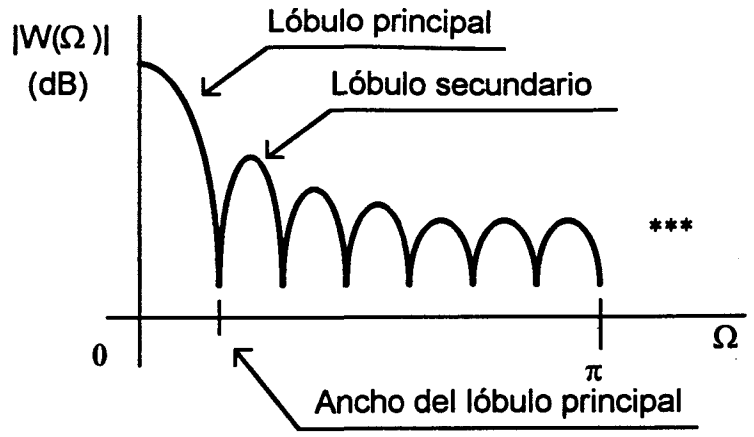
En cuanto a la longitud de la ventana , independientemente de la ventana escogida , cuanto más larga sea más preciso será el filtro resultante pero con la implicación que los algoritmos de procesado serán más largos.

A continuación se muestra una tabla donde aparecen los valores del ancho del lóbulo principal y la amplitud relativa de los lóbulos secundarios de la TF de las ventanas anteriormente estudiadas:

Tipo de VENTANA	Amplitud relativa del lóbulo secundario en (dB)	Ancho del lóbulo principal
RECTANGULAR	-13	$4\pi/(M+1)$
HAMMING	-41	$8\pi/M$
HANNING	-31	$8\pi/M$
BARTLETT	-25	$8\pi/M$
BLACKMAN	-57	$12\pi/M$

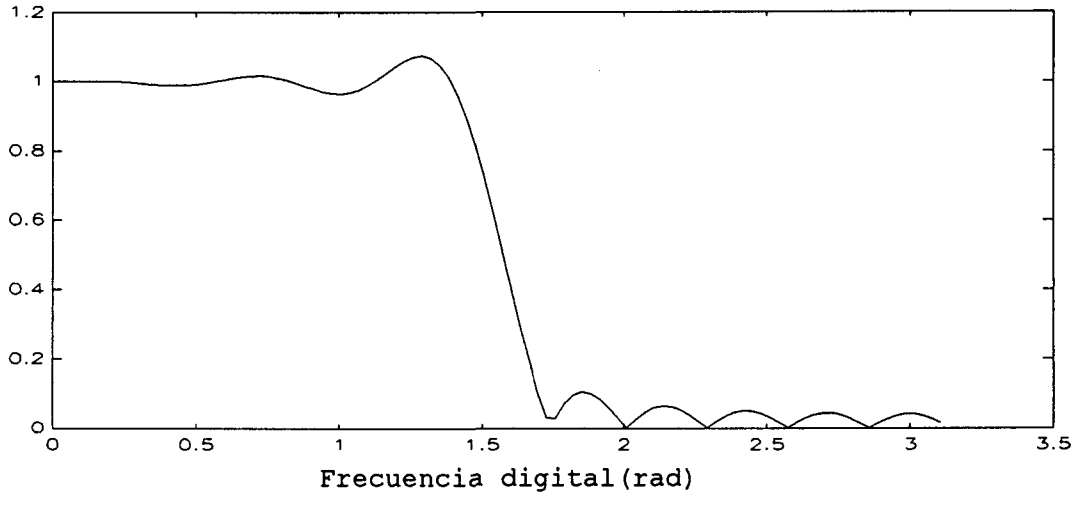
Para que la TF de w[n] se aproxime lo más posible a una $\delta(\Omega)$ tiene que tener un lóbulo principal lo más estrecho posible y que los lóbulos secundarios estén bastante atenuados , ya que los lóbulos secundarios introducen distorsión por efecto GIBBS (rizado en las zonas de discontinuidad).

La forma característica del módulo de la TF de las ventanas anteriormente descritas es como muestra la gráfica siguiente:

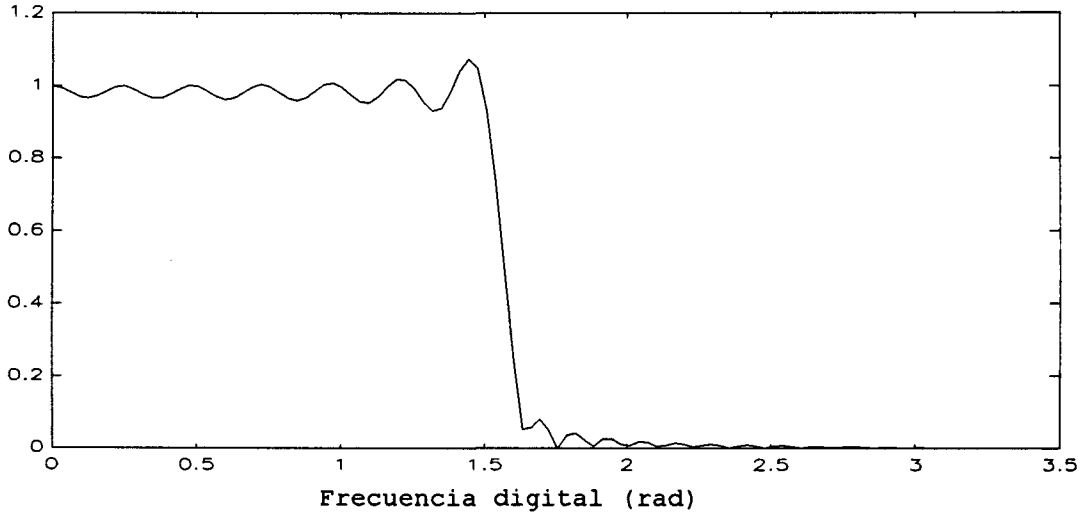


Como caso práctico observemos como al diseñar un filtro paso bajo con frecuencia de corte $\pi/2$ por el método de las ventanas , usando una ventana rectangular , el módulo de dicho filtro es más parecido al ideal a medida que aumentamos la longitud de la ventana.

1° Ventana rectangular de longitud M=22



2° Ventana rectangular de longitud M=52



Como el diseño lo hemos hecho con una ventana rectangular la cual tiene los lóbulos secundarios menos atenuados observamos en la figura como aparece distorsión por efecto GIBBS .

Como mencionamos al principio de este apartado este tipo de filtros iba a tener un retardo de grupo constante que será lo que calculemos a continuación:

hi[n] tiene el módulo ideal y fase cero.

mas tarde hemos obtenido

$$hd[n]=hi[n-M/2]$$

con lo cual hd[n] tendrá una fase $\Omega M/2$, es decir :

$$hi[n] \quad <===> \quad Hi(\Omega)$$
$$hd[n] \quad <===> \quad Hi(\Omega)\exp(-j\Omega M/2)$$

Nuestro filtro será h[n]=w[n] hd[n]

por consiguiente h[n] tiende a tener la fase $-\Omega M/2$ y un retardo de grupo de valor:

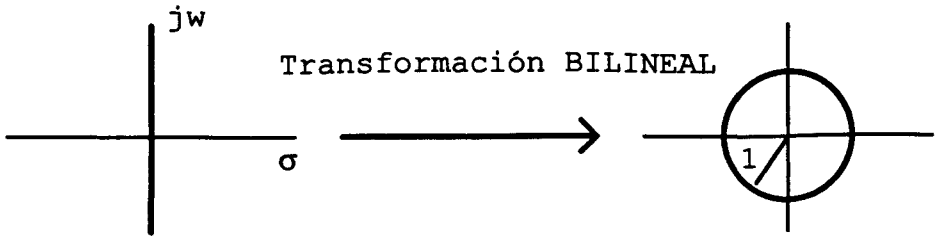
$$\tau(\Omega) = -\frac{d\theta(\Omega)}{d\Omega} = -\frac{d(-\Omega M/2)}{d\Omega} = M/2 = \text{Constante}$$

Como conclusión , la respuesta en frecuencia del filtro obtenido se adaptan más al ideal a medida que aumentamos la longitud de la ventana , aumentando también el retardo de grupo y por tanto el orden de dicho filtro (el orden del filtro es igual a la longitud de la ventana menos 1 n=M-1).

MÉTODO DE LA TRANSFORMACIÓN BILINEAL:

En el apartado anterior hemos mostrado las bases para el diseño de filtros FIR mediante el método de las ventanas , ahora lo haremos para diseñar filtros IIR basándonos en el diseño de filtros analógicos.

Antes de exponer dichas bases vamos a mostrar un tipo de transformación matemática encargada de pasar del dominio complejo del plano S (plano utilizado en el análisis de sistemas y señales de tiempo continuo) al dominio del plano Z (plano utilizado en el análisis de sistemas y señales de tiempo discreto). Esta transformación es la **transformación BILINEAL.**



Plano S Plano Z

Esta transformación nos debe dar en la frecuencia real un cambio de dominio de la forma:

$$-inf \leq w \leq +inf \quad <=========> \quad -\pi \leq \Omega \leq \pi$$

Este cambio de dominio no puede ser lineal ya que hay que condensar ejes infinitos en ejes finitos.

La transformación Bilineal es la siguiente:

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

La conclusión sacada de todo esto es que si Ha(s) es la función de transferencia de un filtro analógico el correspondiente filtro digital tendrá una función de transferencia (y por tanto características de filtro similares pero en el dominio de frecuencias digitales) de la forma:

$$\boxed{H(z) = Ha\left[\frac{2}{T}\left(\frac{1-z^{-1}}{1+z^{-1}}\right)\right]}$$

Veamos qué implicaciones tiene esta transformación al pasar del dominio S al dominio Z , para lo cual despejando de la transformación Bilineal nos queda:

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

si tomásemos en cuenta que $s=\sigma+jw$ tenemos:

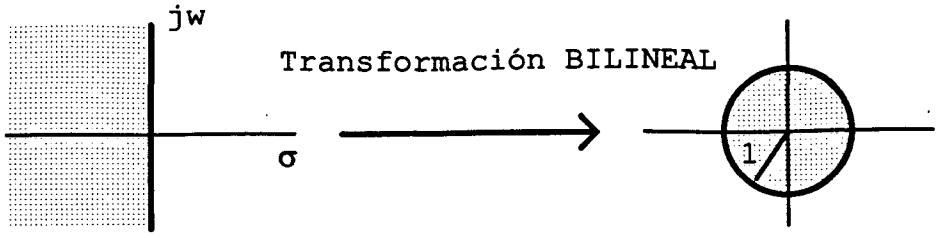
$$z = \frac{1+\frac{\sigma T}{2}+j\frac{wT}{2}}{1-\frac{\sigma T}{2}-j\frac{wT}{2}}$$

entonces obtenemos lo siguiente:

$$\sigma < 0 \Rightarrow |z| < 1; \forall w$$

$$\sigma > 0 \Rightarrow |z| > 1; \forall w$$

Esto gráficamente representa lo siguiente:



Por lo tanto ,que los polos de Ha(s) que estén en el semiplano izquierdo del plano S caerán dentro de la circunferencia unidad del plano Z y los que estén a la derecha fuera de dicha circunferencia. Por tanto si Ha(s) es la función de transferencia de un filtro estable y causal , el filtro H(z) obtenido mediante transformación bilineal también será estable y causal.

Particularicemos ahora la transformación bilineal en eje imaginario del plano S (eje jw , que corresponde a la frecuencia real analógica), esto nos queda como:

$$s = jw \Rightarrow z = \frac{1+j\frac{wT}{2}}{1-j\frac{wT}{2}} \Rightarrow |z| = 1 \ \forall w$$

Lo que es equivalente a:

$$e^{j\Omega} = \frac{1+j\frac{wT}{2}}{1-j\frac{wT}{2}}$$

y también se cumple que:

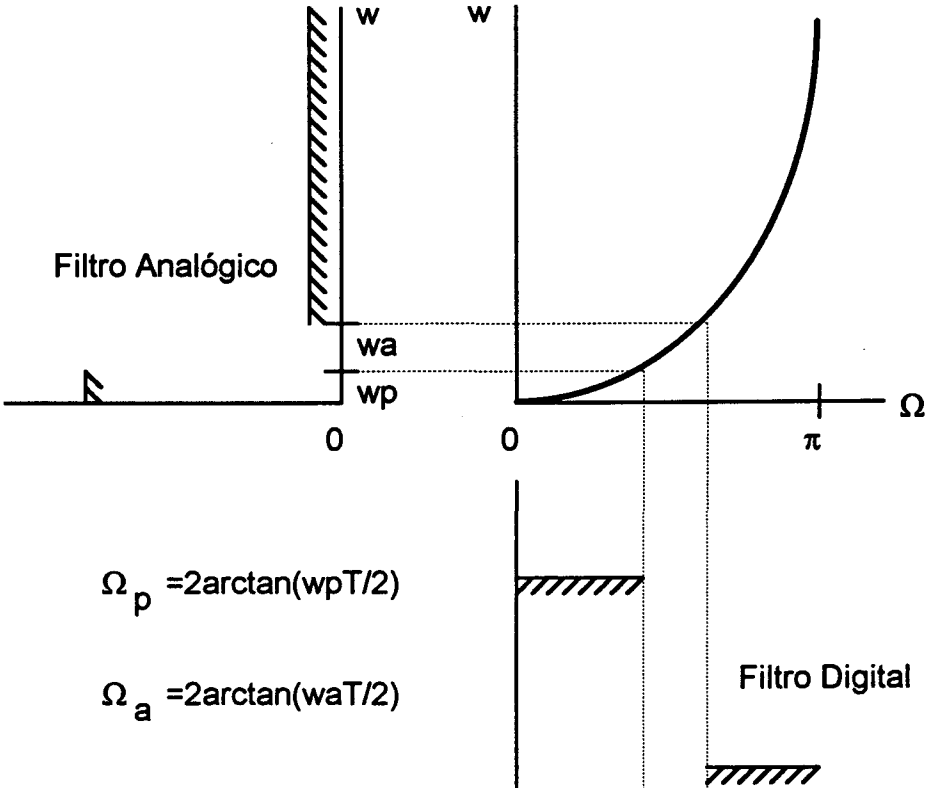
$$s = \frac{2}{T}\left(\frac{1-z^{-1}}{1+z^{-1}}\right) \Rightarrow s = \frac{2}{T}\left(\frac{1-e^{-j\Omega}}{1+e^{-j\Omega}}\right)$$

$$s = \sigma + jw = \frac{2}{T}\left[\frac{2e^{-j\Omega/2}(j\,\mathrm{sen}\,\Omega/2)}{2e^{-j\Omega/2}(\cos\Omega/2)}\right] = \frac{2j}{T}\tan(\Omega/2)$$

por tanto la relación de frecuencias reales del filtro digital y el analógico es:

$$\boxed{\begin{aligned} w &= \frac{2}{T}\tan(\Omega/2) \\ \Omega &= 2\arctan(wT/2) \end{aligned}}$$

Gráficamente sería como se muestra a continuación:



Ya hemos visto en que se basa la Transformación Bilineal y por lo tanto vamos a pasar a describir cómo se diseñan los filtros IIR basados en prototipos analógicos.

Los filtros analógicos en los que basaremos nuestro diseño son:

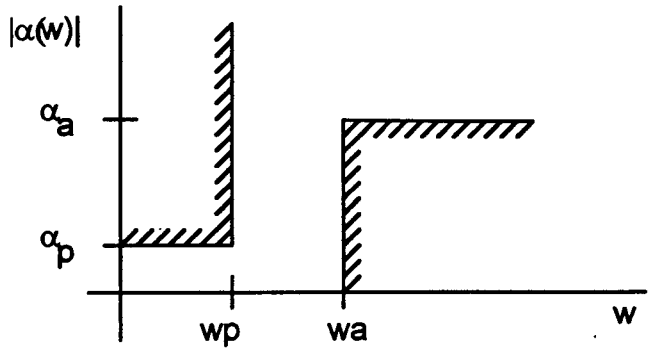
Filtro Butterworth (Máximamente plano).

Filtro Chebyshev Directo (Rizado en la banda de paso).

Filtro Chebyshev Inverso (Rizado en la banda atenuada).

Filtro Elíptico o de Cauer (Rizado en ambas bandas).

Estos filtros son diseñados teniendo en cuenta las características del módulo de la TF de h(t) y por tanto la fase de dichos filtros vendrá impuesta y por tanto no podemos conseguir que sea lineal , ya que si intentamos que un filtro tenga un módulo lo más ideal posible la fase no lo será y viceversa. En estos filtros se diseña un prototipo paso bajo y luego se hacen las correspondientes transformaciones para obtener el filtro deseado (paso-alto,paso-banda o rechazo-banda).Las características de especificación que se utilizan para este filtro es $|\alpha(w)|$ que se trata de la atenuación del filtro , es decir , la inversa de $|H(w)|$ .Gráficamente sería:



Los parámetros de diseño de este filtro son:

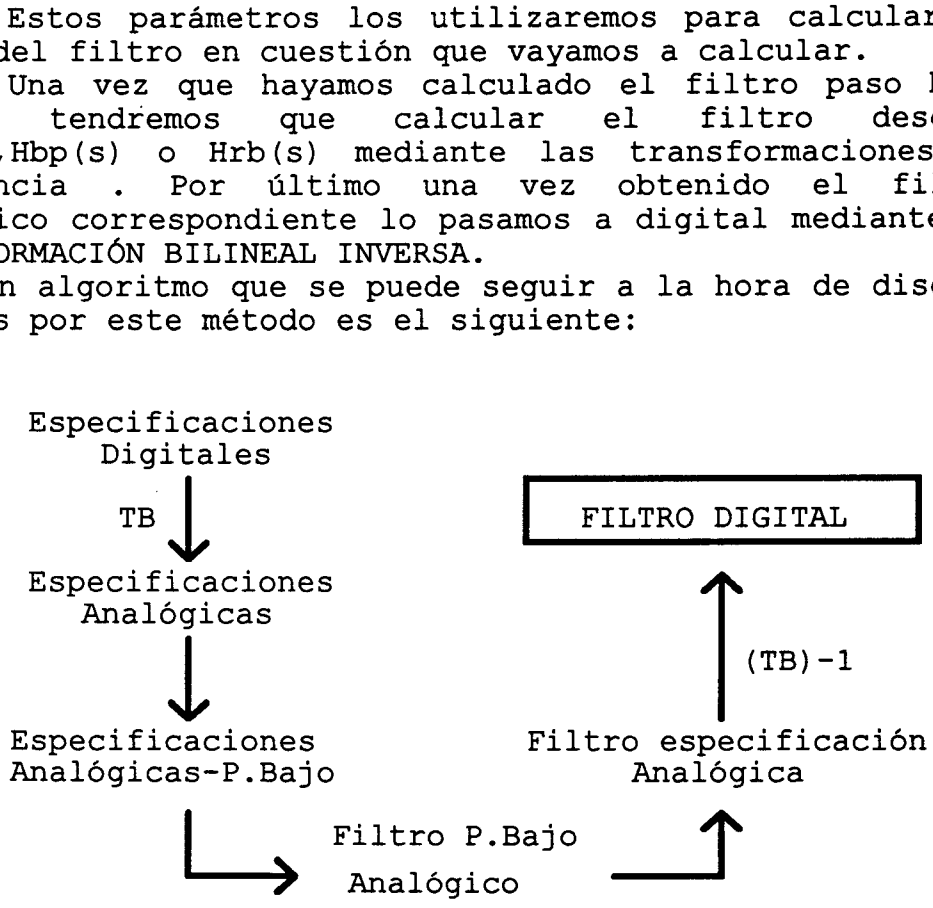
Parámetro de tolerancia: $ks = \frac{wp}{wa}$

Parámetro de discriminación: $kd = \sqrt{\frac{10^{\frac{\alpha p}{10}} - 1}{10^{\frac{\alpha a}{10}} - 1}}$

Estos parámetros los utilizaremos para calcular el orden del filtro en cuestión que vayamos a calcular.

Una vez que hayamos calculado el filtro paso bajo Hlp(s) tendremos que calcular el filtro deseado Hhp(s),Hbp(s) o Hrb(s) mediante las transformaciones en frecuencia . Por último una vez obtenido el filtro analógico correspondiente lo pasamos a digital mediante la TRANSFORMACIÓN BILINEAL INVERSA.

Un algoritmo que se puede seguir a la hora de diseñar filtros por este método es el siguiente:



2.2:PROBLEMAS DE OPERAR CON PRECISIÓN FINITA.

Los filtros descritos por ecuaciones en diferencias están caracterizados por los coeficientes que van a ser la huella de identidad de dicho sistema . Estos van a marcar

la respuesta en frecuencia e implícito a ello la posición de los polos y ceros de la función de transferencia H(z). Si al intentar implementar un filtro digital en un sistema real no podemos darle la precisión suficiente a los coeficientes esto nos va a suponer que hemos variado la H(z) y por lo tanto la respuesta en frecuencia del sistema e implícitamente hemos movido los polos y los ceros de sus posiciones originales.

Hemos visto anteriormente que según sea la posición que ocupen los polos en el plano Z el filtro puede volverse inestable caso que afectará a los filtros IIR en cambio a los FIR solo les cambiará la respuesta en frecuencia pero no los volverá inestables debido a que tienen los polos situados en el origen.

Veamos de manera analítica como analizar esta cuantificación en los coeficientes de un filtro.

Empecemos analizando las posible estructuras que puede tener tanto los sistema FIR como IIR

ESTRUCTURAS FIR:Las posibilidades que tenemos para implementar estas estructuras son:

Estructura DIRECTA:

$$H(z)=\sum_{k=0}^{M} b_k z^{-k}$$

Estructura en CASCADA:

$$H(z)=\prod_{k=1}^{Ms} (b_{0k}+b_{1k}z^{-1}+b_{2k}z^{-2})$$

ESTRUCTURAS IIR:Las posibilidades que tenemos para implementar estas estructuras son:

Estructura DIRECTA:

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

Estructura en CASCADA:

$$H(z)=\prod_{k=1}^{Ns} \frac{b_{0k}+b_{1k}z^{-1}+b_{2k}z^{-2}}{1-a_{1k}z^{-1}-a_{2k}z^{-2}}$$

Estructura en PARALELO:

$$H(z)=\sum_{k=0}^{Np} C_k z^{-k}+\sum_{k=1}^{Ns} \frac{e_{0k}+e_{1k}z^{-1}}{1-a_{1k}z^{-1}-a_{2k}z^{-2}}$$

El hecho de utilizar estructuras alternativas a la realización directa es que al cuantificar los coeficientes pueden tener menor sensibilidad a dicha cuantificación como se verá a continuación.

Cuantificación de filtros IIR:

Estudiemos la cuantificación de los coeficientes en forma directa :

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

Al cuantificar los coeficientes la función nos queda como:

$$\hat{H}(z)=\frac{\sum_{k=0}^{M} \hat{b}_k z^{-k}}{1-\sum_{k=1}^{N} \hat{a}_k z^{-k}}$$

donde:

$$\hat{a}_k = a_k + \Delta a_k \; ; \quad \hat{b}_k = b_k + \Delta b_k$$

denotando el error de cuantificación con:

$$\Delta a_k \, , \, \Delta b_k$$

Analizando el denominador y denotando por zj la posición de los polos nos queda como:

$$A(z)=1-\sum_{k=1}^{N} a_k z^{-k}=\prod_{j=1}^{N}(1-z_j z^{-1})$$

Ahora denotaremos los polos de la función cuantificada como zi+Δzi y el error de localización lo podemos expresar en función del error del coeficiente como:

$$\Delta z_i=\sum_{k=1}^{N}\frac{\partial z_i}{\partial a_k}\Delta a_k ; i=1,2,....,N$$

Que podemos escribir como:

$$\left(\frac{\partial A(z)}{\partial z_i}\right)_{z=z_i}\frac{\partial z_i}{\partial a_k}=\left(\frac{\partial A(z)}{\partial a_k}\right)_{z=z_i}$$

Con lo que obtenemos la función de sensibilidad de los polos en función de la cuantificación de los coeficientes:

$$\frac{\partial z_i}{\partial a_k} = \frac{z_i^{N-k}}{\prod_{j=1, j\neq i}^{N} (z_i - z_j)}$$

La expresión general se nos queda como:

$$\Delta z_i = \sum_{k=1}^{N} \frac{z_i^{N-k}}{\prod_{j=1, j\neq i}^{N} (z_i - z_j)} \Delta a_k$$

La función anterior nos indica que si los polos están muy juntos la variación es elevada y si están separados la sensibilidad es menor, para una cuantificación de los coeficientes dada.

Poca Sensibilidad

Mucha Sensibilidad

Una conclusión importante que podemos sacar de aquí es que cuando un sistema lo representamos como un conjunto de sistemas de segundo orden habrán situaciones en las que habremos alejados las posiciones relativas de los polos y por tanto el sistema tendrá menor sensibilidad a la cunatificación y es por eso por lo que a veces es mejor utilizar estructuras alternativas a la directa y por supuesto habrá veces que no.

El análisis analítico realizado aquí se puede extrapolar a los ceros teniendo en cuanta las mismas suposiciones.

Como conclusión , podemos sacar que las realizaciones paralelo y cascada que están formadas por estructuras directas de segundo orden (polos y ceros complejos conjugados) , aíslan estos pares (ceros o polos) del resto con lo cual los errores de cuantificación se pueden disminuir bastante.

Cuantificación de filtros FIR:

Ahora vamos a estudiar la cuantificación en forma directa de un filtro FIR:

$$H(z) = \sum_{n=0}^{M} h[n] z^{-n}$$

Si ahora cuantificamos los coeficientes y denotamos:

$$\hat{h}[n] = h[n] + \Delta h[n]$$

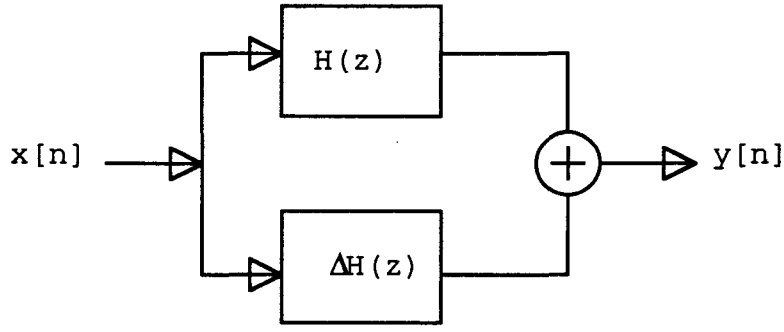
nos quedará una función de transferencia como:

$$\hat{H}(z) = \sum_{n=0}^{M} \hat{h}[n] z^{-n} = H(z) + \Delta H(z)$$

donde:

$$\Delta H(z) = \sum_{n=0}^{M} \Delta h[n] z^{-n}$$

Que se puede modelar con un sistema como el siguiente:



Un razonamiento análogo al realizado para los filtros IIR se puede hacer aquí obteniendo la siguiente ecuación:

$$\Delta c_i = \sum_{k=1}^{M} \frac{c_i^{M-k}}{\prod_{j=1, j\neq i}^{M} (c_i - c_j)} \Delta h[k]$$

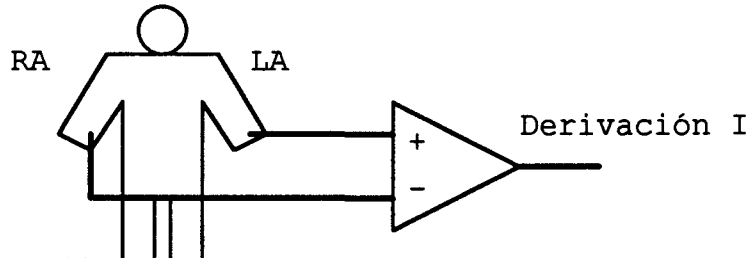
Igual que antes podemos sacar la conclusión que si los ceros están muy juntos los efectos de la cuantificación serán más notables y que habrá situaciones en las que las estructuras alternativas (en este caso la estructura en cascada) tendrá menor sensibilidad a la cuantificación de los coeficientes . Por último indicar que para los filtros FIR es fácil garantizar la fase lineal aún cuando se cuantifiquen los coeficientes.

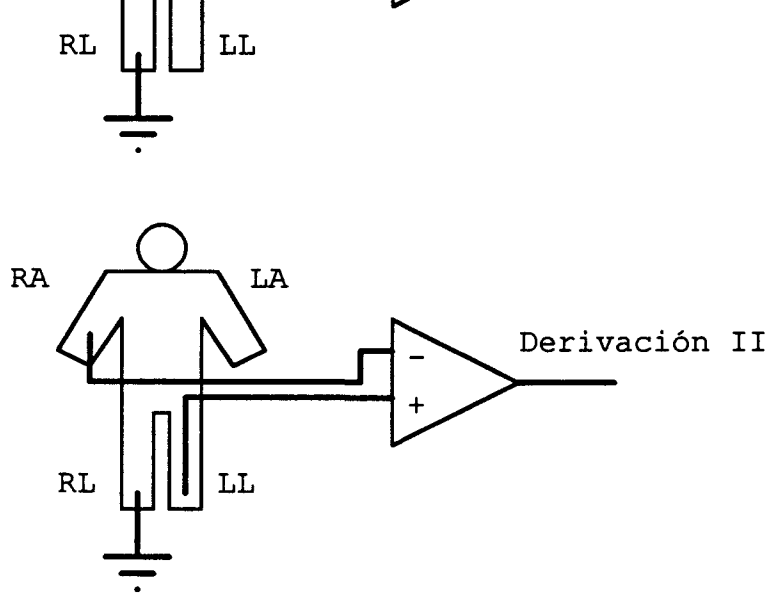
2.3:SEÑAL DE ECG.

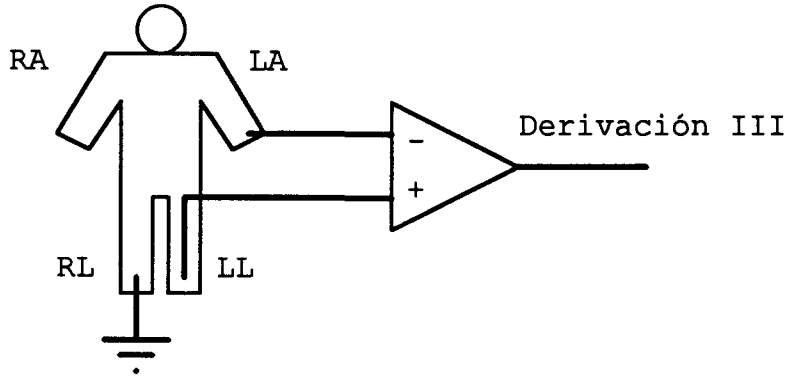
Una de las principales técnicas en el diagnóstico de enfermedades cardiacas esta basada en el electrocardiograma.El electrocardiógrafo permite conocer defectos del corazón midiendo la señal de ECG . La señal de ECG son los potenciales que aparecen en la superficie del cuerpo debidos al movimiento cardiaco. Por tanto con la medición de la señal de ECG , se pueden obtener el ritmo del corazón y otros parámetros cardiacos.

La manera de obtener la señal de ECG es conectar al paciente con unos electrodos en unas partes concretas del cuerpo y mediante una amplificación diferencial obtener la señal a estudio.

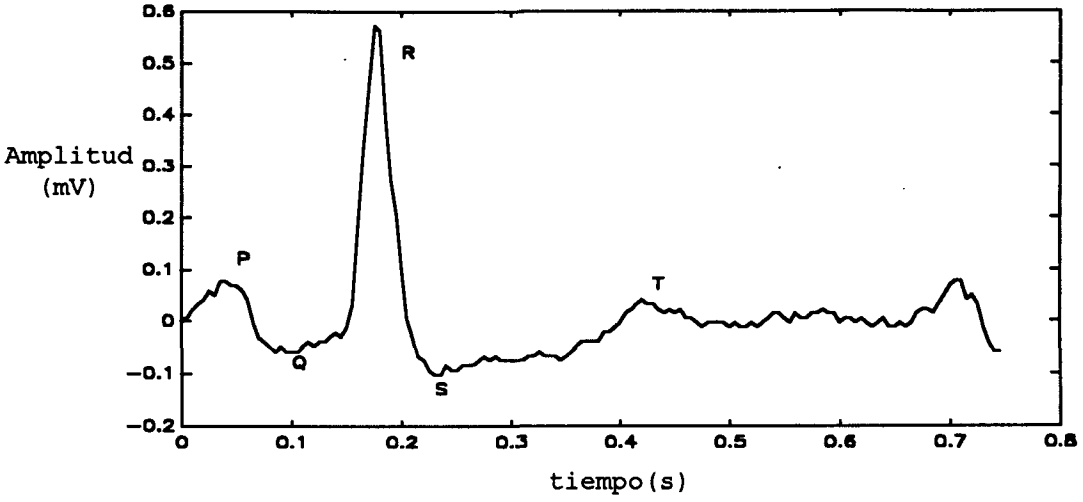
Las maneras más usuales de conexión son:







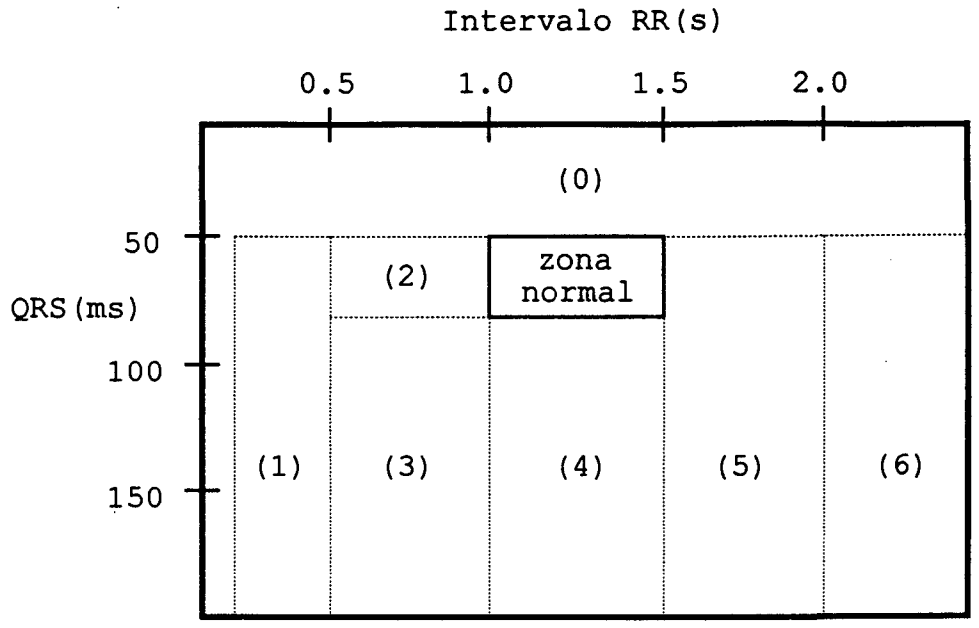
Supongamos en adelante que estaremos tomando la señal de ECG siempre con la derivación I y en estas circunstancias la forma de la señal obtenida en el dominio temporal tiene la siguiente forma:



La señal tiende a ser periódica aunque de frecuencia variable , dependiendo del ritmo cardiaco del individuo.

De esta señal la parte fundamental es el complejo QRS ya que es la parte más aguda y que más se diferencia del resto de la señal , por lo que podemos obtener información precisa de cuando se ha producido un impulso cardiaco.

El cardiólogo para obtener el diagnóstico de una enfermedad cardiaca analiza todos los intervalos de señal , es decir , la onda P , Q , R , S y T su amplitud y las distancias temporales entre ellas. También hay muchas enfermedades que se detectan mediante un análisis de arritmias de la señal de ECG y que se basa en medir la duración del complejos QRS y la distancia entre las señales R.A partir de los valores obtenidos podemos conocer unas patologías concretas.A continuación se muestra una tabla donde se pueden conocer las posibles patologías tras un análisis de arritmias:



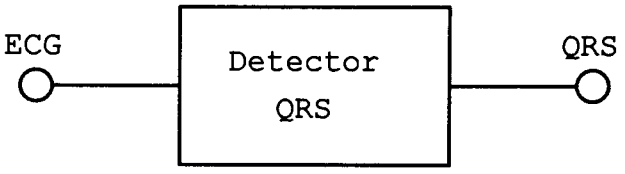
Las posibles arritmias se obtienen según donde se sitúen las pulsaciones dentro del cuadro anterior.

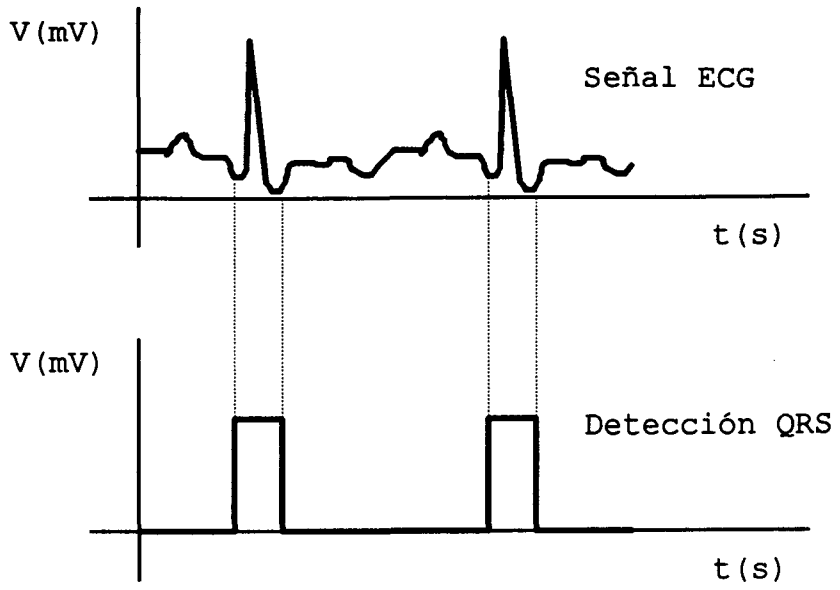
Por ejemplo una arritmia como una Taquicardia ocasionaría que todas las pulsaciones estuvieran en la zona (1) o una Bradicardia en la zona (6). Otras arritmias se obtendrían estudiando secuencias de pulsaciones como por ejemplo una rápida contracción ventricular con una amplia pausa está caracterizado por un corto intervalo RR acompañado con una larga duración del complejo QRS , seguido por un largo intervalo RR acompañado de una duración normal del complejo QRS . Esto se manifiesta situando dos puntos en el diagrama anterior , el primero en la zona (3) y el segundo en la zona (5).

2.4:DETECTOR DEL COMPLEJO QRS.

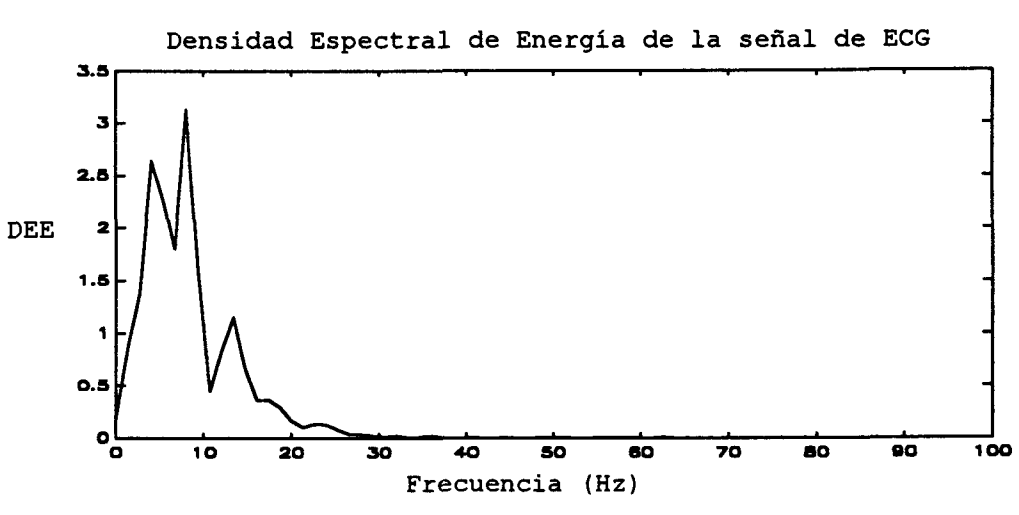
En el apartado anterior estudiamos someramente las características de la señal de ECG y se vio la importancia que tenía la detección y posterior análisis del complejo QRS para detectar patologías en enfermos cardiacos . En este apartado vamos a desarrollar un método basado en procesado digital para obtener el complejo QRS de la señal de ECG.

El sistema que implementaremos se adapta al siguiente esquema:





Como los algoritmos de detección van a ser filtros digitales , pasemos a estudiar el espectro de la señal de ECG.



Como observamos en la figura la señal de ECG está por debajo de 20Hz y su pico más energético
el complejo QRS se moverá alrededor de los 10Hz dependiendo del paciente que se trate.

La frecuencia de muestreo a utilizar tiene que ser mayor de 80Hz y nosotros tomaremos un margen prudencial y trabajaremos a 200Hz.

El sistema podrá estar compuesto por los siguientes bloques:

#Filtro paso bajo que elimine el posible ruido que pueda añadir la red eléctrica y otras fuentes de interferencia.

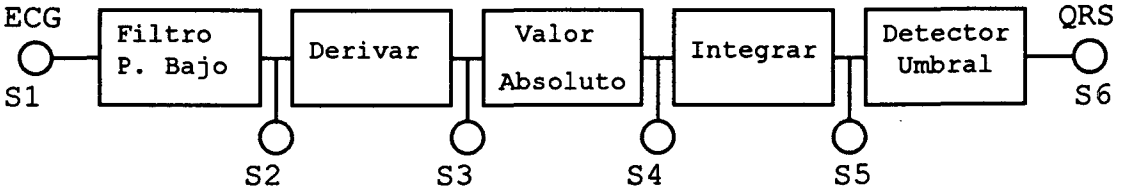
#Un derivador para acentuar más el pico del complejo QRS con respecto al resto de la señal.

#Una vez el pico está acentuado calculamos el valor absoluto para coger las partes negativas que pertenecen al complejo QRS derivado.

#Luego estos picos positivos los integramos para obtener una señal parecida a un pulso.

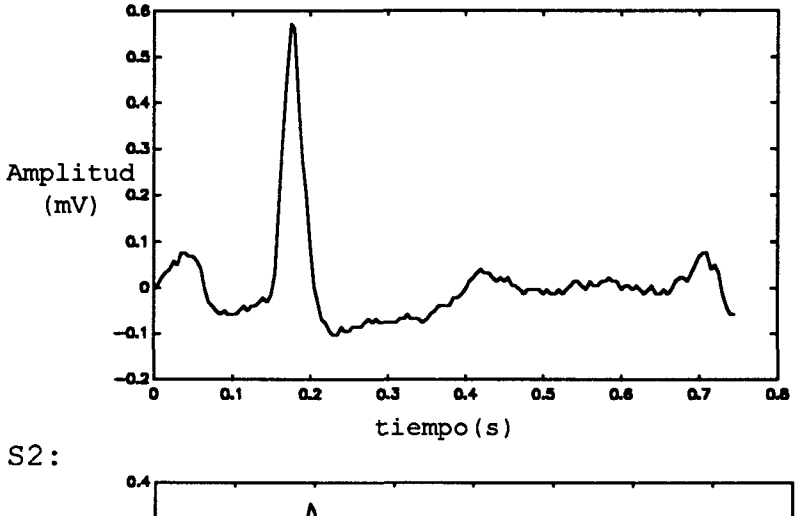
#Por último colocamos un detector de umbral para obtener un pulso en el lugar que había un complejo QRS.

El diagrama de bloques tendrá la siguiente forma:

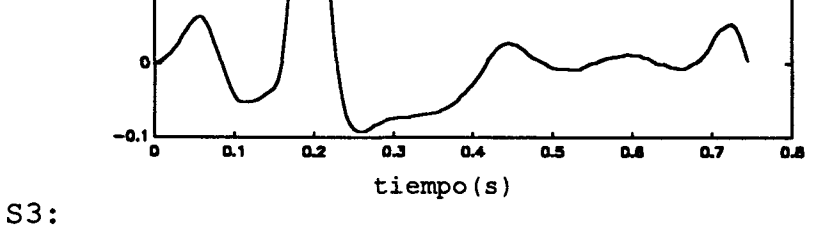


Las señales se van procesando según muestran los siguientes diagramas:

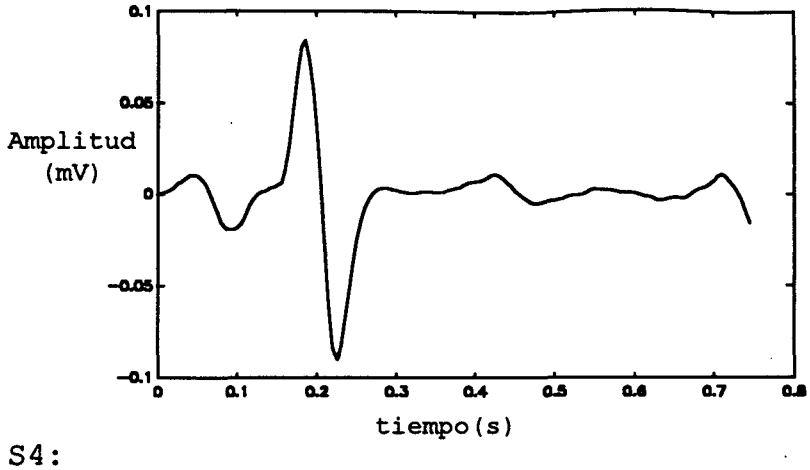
#SEÑAL S1:



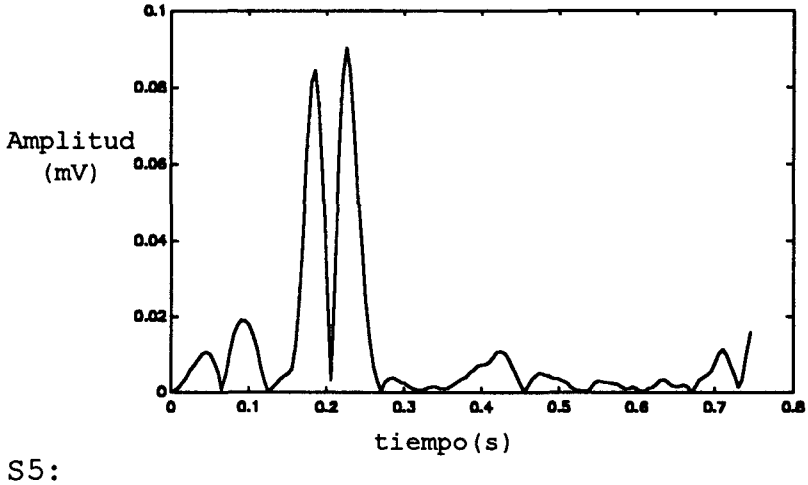
#SEÑAL S2:



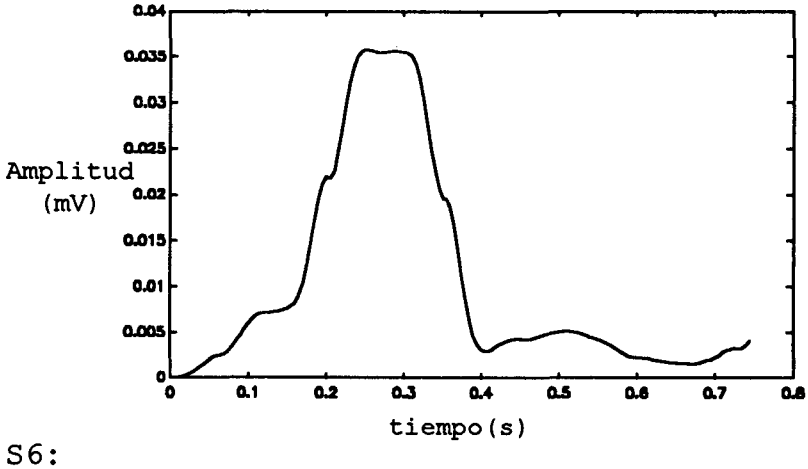
#SEÑAL S3:



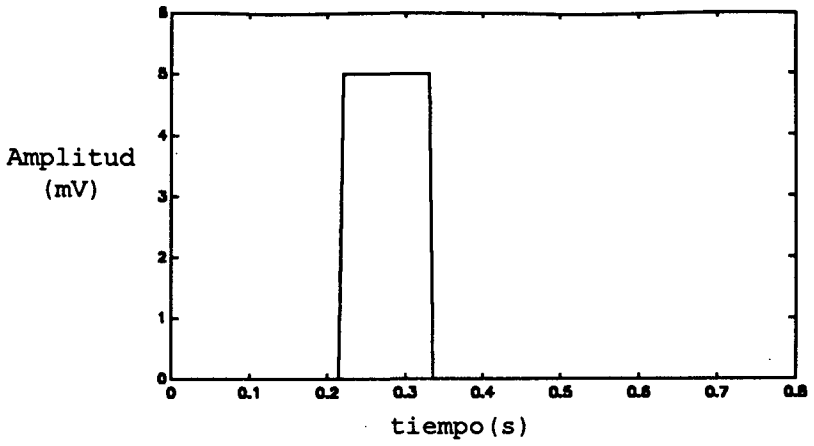
#SEÑAL S4:



#SEÑAL S5:



#SEÑAL S6:



En el capítulo 5 se desarrolla la forma de como se realiza el detector de QRS de manera práctica.

CAPITULO 3:SIMULADOR "DISFILT" EN MATLAB.

En este capítulo vamos a describir las funciones que realiza el programa DISFILT que ha sido desarrollado en **MATLAB para WINDOWS** y cuyo objetivo es diseñar y analizar filtros digitales de propósito general tanto con la precisión que ofrece el MATLAB como con la que tendrían dichos filtros trabajando en un sistema real como es el DSP TMDS3200051.

3.1:DISEÑO Y ANÁLISIS DE FILTROS DIGITALES CON PRECISIÓN MATLAB.

El diseño y análisis de filtros digitales con precisión Matlab se realizan en este programa con tres aplicaciones concretas que son:

- **Diseño de Filtros Digitales.**
- **Análisis de Filtros Digitales.**
- **Estudiar filtros analógico global.**

Para ello vamos a desglosar qué podemos hacer en cada una de ellas:

Diseño de Filtros Digitales:

Esta aplicación está basada en el diseño de filtros digitales pasándole especificaciones en módulo por tanto la fase vendrá impuesta.Al entrar a ejecutar esta aplicación nos preguntará que tipo de filtro queremos diseñar y tenemos dos posibilidades FIR(método de las ventanas) o IIR(método de la transformación bilineal).

Dentro de los filtros FIR podemos diseñar los siguientes:

- FIR con ventana RECTANGULAR.
- FIR con ventana HAMMING.
- FIR con ventana HANNING.
- FIR con ventana BARTLETT.
- FIR con ventana BLACKMAN.

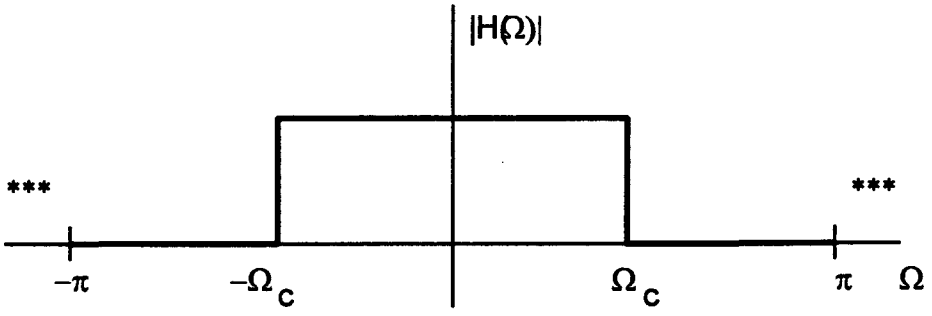
En cuanto a los IIR podemos diseñar los filtros haciendo una de las siguientes aproximaciones:

- Aproximación BUTTERWORTH (máximamente plano).
- Aproximación CHEBYSHEV DIRECTO (rizado en la banda de paso).
- Aproximación CHEBYSHEV INVERSO (rizado en la banda atenuada).
- Aproximación ELIPTICO (rizado en ambas bandas).

Para diseñar filtros FIR escogemos el tipo de filtro FIR que queramos , es decir , el tipo de ventana a utilizar y una vez hecho esto podemos elegir si lo queremos PASO-BAJO , PASO-ALTO , PASO-BANDA o RECHAZO-BANDA . Más tarde nos pedirá el orden del filtro que queremos diseñar y las

frecuencias de corte . Una vez hecho esto el programa se encargará de diseñar el filtro , es decir , calcular los coeficientes y guardarlos en el disco.

La carátula de especificaciones en las cuales está basada el diseño es la siguiente(por ejemplo para un paso bajo):

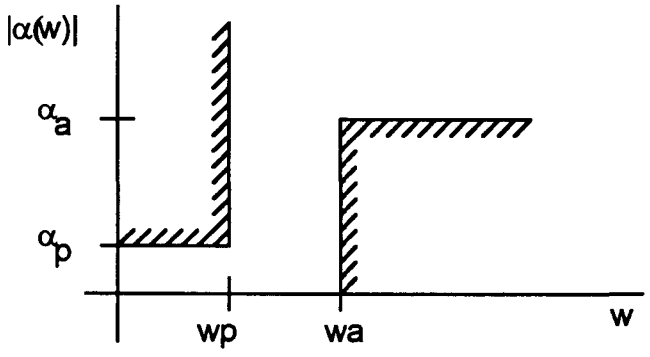


Las carátulas para paso-alto , paso-banda o rechazo-banda son fácilmente extrapolables a partir de esta.

Al darle las frecuencias de corte la respuesta del filtro diseñado se aproximará más a la respuesta ideal cuanto mayor sea el orden de dicho . Para más información sobre el método de las ventanas consultar apartado 2.1.

Para diseñar filtros IIR una vez elegido el tipo de aproximación le tendremos que pasar si lo queremos PASO-BAJO , PASO-ALTO , PASO-BANDA o RECHAZO BANDA y una vez hecho esto le tendremos que pasar las frecuencia para la banda de paso y para las bandas atenuadas , también tendremos que pasarle la atenuación máxima en la banda de paso (>0dB) y la atenuación mínima en la banda atenuada (>0dB) . Una vez pasadas las especificaciones del filtro que queremos diseñar el programa calculará los coeficientes y los guardará en el disco.

La carátula de especificación que se usará para los filtros IIR es una como la siguiente donde α es la atenuación en dB(por ejemplo para un paso-bajo):



Las carátulas para paso-alto , paso-banda o rechazo-banda son fácilmente extrapolables a partir de esta.

Al pasarle los parámetros en frecuencia (tanto para FIR como parar IIR)tendremos dos posibilidades:

Frecuencia DIGITALES (0 hasta π).

Frecuencia ANALÓGICA (0 hasta 0.5*frecuencia muestreo).

Al pasarle la especificación ANALÓGICA es la respuesta que va a tener el filtro una vez que esté trabajando en un sistema como un DSP.

Análisis de Filtros Digitales:

En esta aplicación podemos analizar los filtros digitales anteriormente diseñados o podemos especificarle uno concreto que ya tengamos diseñado nosotros . Los filtros diseñados que podemos analizar son los FIR (Rectangular , Hamming , Hanning , Bartlett o Blackman) y los IIR (Butterworth , Chebyshev directo , Chebyshev inverso o Eliptico).

En cuanto al filtro que podemos especificarle tenemos tres posibilidades:

Ecuación en diferencias.
Función Racional H(z).
Diagrama de Polos y Ceros.

En la ecuación en diferencias le pasamos los coeficientes de la siguiente ecuación:

$$\sum_{k=0}^{N} a_k y[n-k] = \sum_{k=0}^{M} b_k x[n-k]$$

En la función racional H(z) le pasamos los coeficientes de la siguiente ecuación:

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

Y cuando le pasemos un diagrama de polos y ceros le pasaremos las coordenadas de estos (polos y ceros) tanto en coordenadas rectangulares o coordenadas polares . El programa le colocará al filtro una ganancia máxima igual a la unidad.

Una vez elijamos el filtro diseñado o le especifiquemos uno mediante las tres posibilidades anteriores el programa hará el siguiente análisis:

Comprueba primeramente si el filtro es ESTABLE o INESTABLE.
Más tarde mostrará los coeficientes del filtro en cuestión indicándonos el orden y el tipo(paso-bajo , paso-alto , paso-banda o rechazo-banda).
Seguidamente mostrará el módulo del filtro en dB o de forma lineal.
Continuará mostrando la fase y el retardo de grupo.

Por último mostrará el diagrama de POLOS y CEROS.

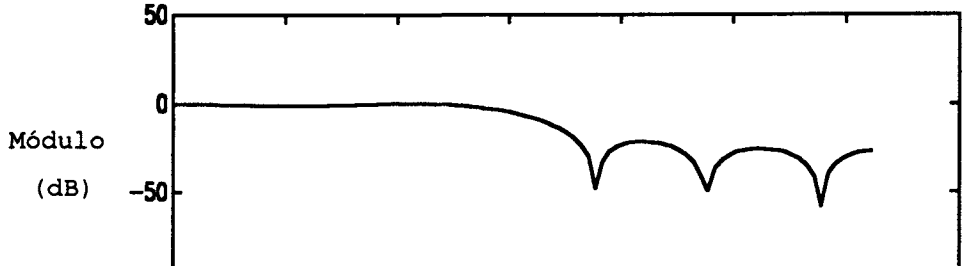
Si por ejemplo el filtro diseñado es un FIR Rectangular de orden 10 , paso bajo y cuya frecuencia de corte es pi/2 el resultado del análisis del programa sería:

Una ecuación en diferencia con los siguientes coeficientes:

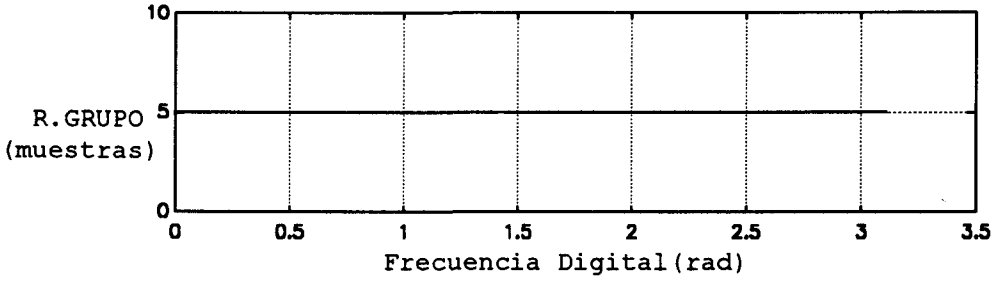
x_n:0.0605;0;-0.1009;0;0.3027;0.4754;0.3027;0;-0.1009;0;0.065

y_n:1

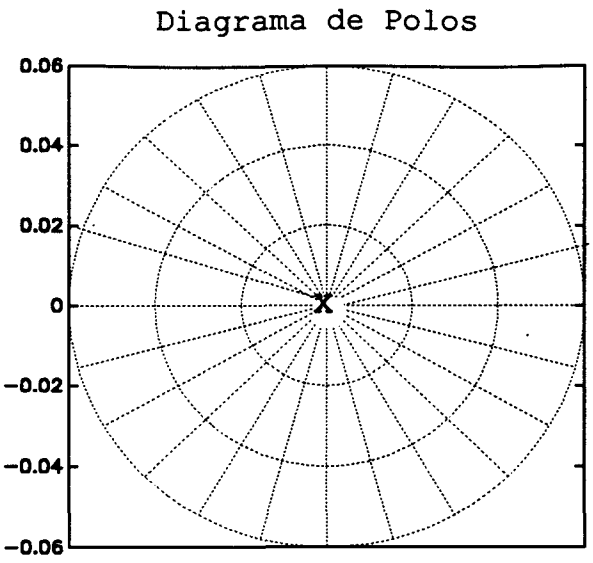
Un módulo como el siguiente:



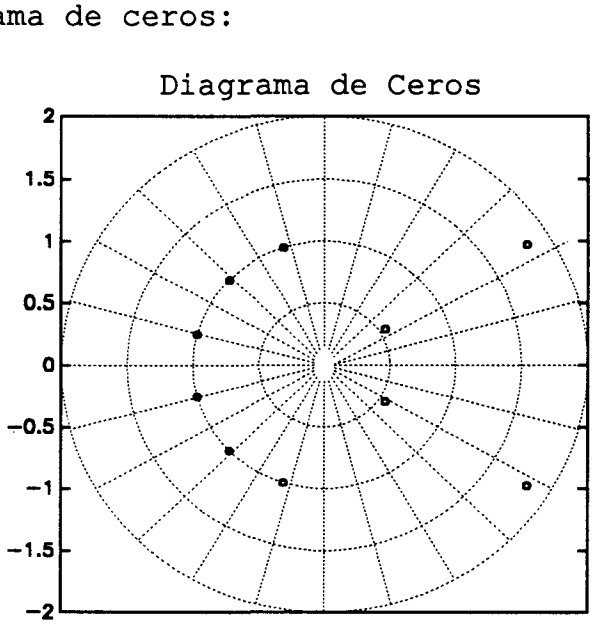
Una fase y un retardo de grupo:



Un diagrama de polos:

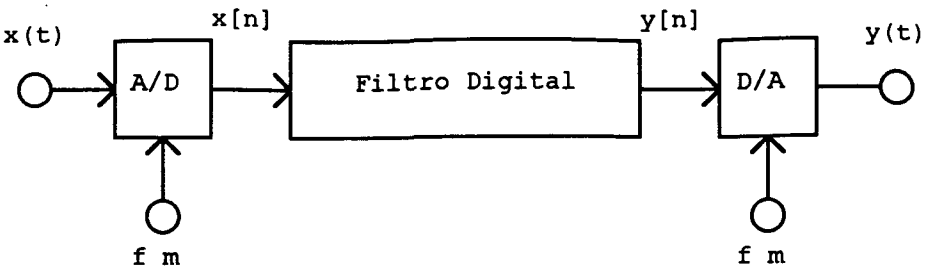


Un diagrama de ceros:



Estudiar Filtro analógico global:

En esta aplicación se mostrará la respuesta que tendrá el filtro digital trabajando con un conversor A/D a la entrada y uno D/A a la salida (por ejemplo un DSP) y que responde al siguiente esquema:



En este análisis el programa preguntará que filtro se quiere analizar:

- FIR/IIR diseñados anteriormente.
-filtro que se especificó en el apartado de análisis de filtros digitales .

Una vez se ha elegido el filtro a analizar se preguntará por la frecuencia de muestreo de los conversores A/D y D/A y consiguientemente el programa calculará la respuesta que tendrá el filtro analógico global mostrando por pantalla lo siguiente:

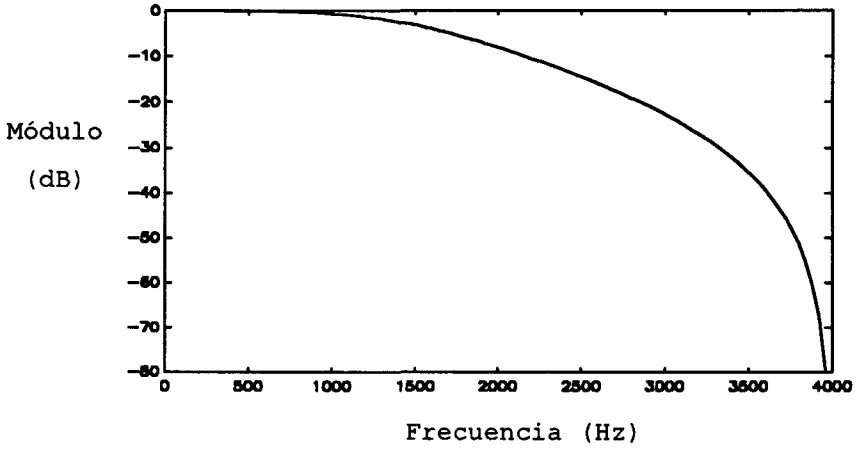
Módulo analógico en dB.
Módulo analógico en escala lineal.
Retardo de grupo

Si como ejemplo diseñamos un filtro de Butterworth con las siguientes características:

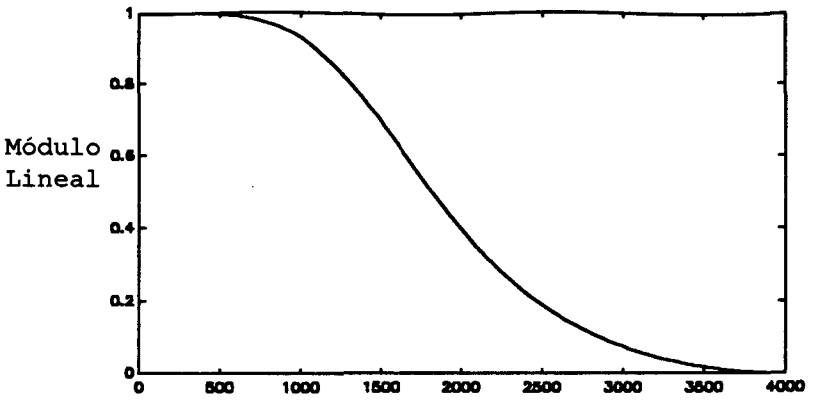
Paso bajo.
Frecuencia de paso 0.2π y atenuación máxima de 1 dB.
Frecuencia de atenuación 0.5π y atenuación mínima 8dB.
Para el equivalente analógico una frecuencia de muestreo de 8000Hz.

Obtenemos los siguientes resultados:

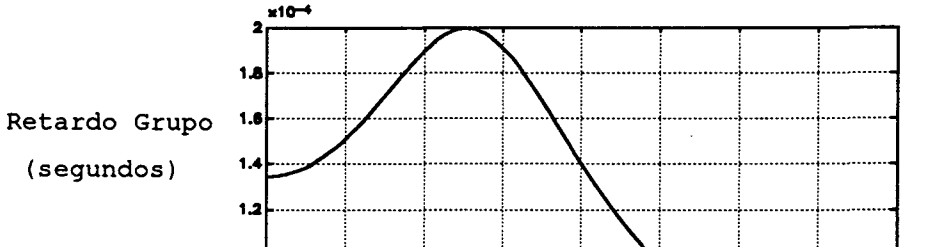
Un módulo en dB:



Un módulo lineal:



Y un retardo de grupo:



3.2:APLICACIÓN DE LOS FILTROS CON PRECISIÓN MATLAB AL TRATAMIENTO DE SEÑALES DE TIEMPO CONTINUO.

Esto se lleva a cabo en el programa mediante la aplicación:

 # Simular con Señales Digitalizadas.

Al entrar en esta aplicación lo primero que se nos pide es el tipo de señal que queremos pasar a través del filtro y tenemos las siguientes posibilidades:

Señal de ECG.
Señal de VOZ.
Otra señal guardada en disco.
Ultima señal tratada.

La señal de ECG y de VOZ son señales de librería que están en el programa a título de ilustración , la señal de ECG está muestreada a 200Hz y la de VOZ a 8000Hz . Está señal de ECG y VOZ se encuentran en los ficheros **ecg.mat** y **voz.mat** y que el usuario puede variar su contenido cuando lo precise . La señal correspondiente a "otra señal

guardada en disco" se encuentra en el fichero **otra_s.mat** y en este el usuario puede guardar la señal que quiera analizar con el programa y la manera de hacerlo es la siguiente:

Mediante un sistema de adquisición (por ejemplo **GLOBAL LAB** o similar) de datos digitaliza la señal a analizar.
Más tarde se convierte a formato MATLAB mediante el mismo sistema de adquisición de datos.
El fichero debe contener un vector que se debe llamar **otra_s** que será donde esté la señal a analizar.

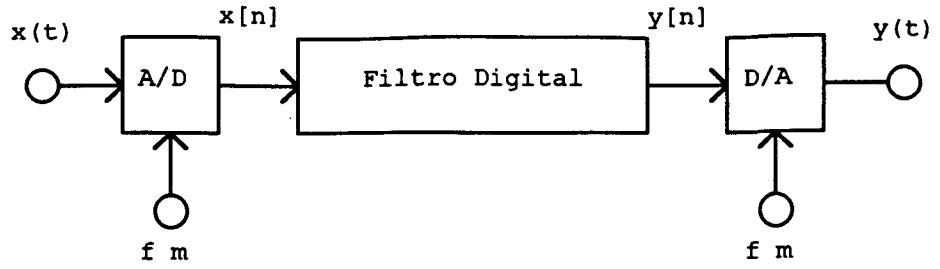
Por último la señal correspondiente a "Ultima señal tratada" se encuentra en el fichero **ult_sen.mat** y es el resultado del último procesado realizado en esta aplicación ,la cual puede servir para:

Poder ir realizando algoritmos de procesado en cascada , ya que podemos ir procesando estos resultados e irlos introduciendo en filtros diferentes para ir viendo como va evolucionando la señal.
Una vez hayamos hecho un procesado coger esta señal que se encuentra en **ult_sen.mat** e introducirla en un sistema de adquisición como GLOBAL LAB y pasarla a una secuencia analógica para analizarla en el dominio de tiempo continuo.

Una vez le hayamos indicado el tipo de señal que vamos a pasar por un filtro concreto el programa nos pedirá por qué filtro queremos hacerlo y tenemos las siguientes posibilidades:

Filtro FIR diseñado anteriormente.
Filtro IIR diseñado anteriormente.
Filtro especificado anteriormente.

Como último dato de entrada nos pedirá a que frecuencia está muestreada la señal en el disco y que tendrá que coincidir con la que nosotros supusimos en el diseño del filtro digital.
Una vez hecho esto el programa pasará a mostrarnos los resultados de una manera gráfica y considerando un análisis analógico ,es decir , que nos mostrará la señal a la entrada y a la salida junto con la respuesta global del siguiente sistema:



El primer resultado gráfico que muestra es la respuesta del filtro elegido mostrando:

Módulo lineal.
Módulo en dB.
Retardo de grupo.

Más tarde muestra simultáneamente la señal a la entrada del filtro y a la salida , ambas en el dominio temporal y con la posibilidad de hacer un ZOOM para ver una zona concreta de la señal con la posibilidad de ver nuevamente la señal original.

Por último nos muestra la DEE de la señal de entrada simultáneamente con la de la salida para poder observar que componentes de frecuencia a eliminado el filtro . También aquí existe la posibilidad de ZOOM que se detalló en el párrafo anterior.

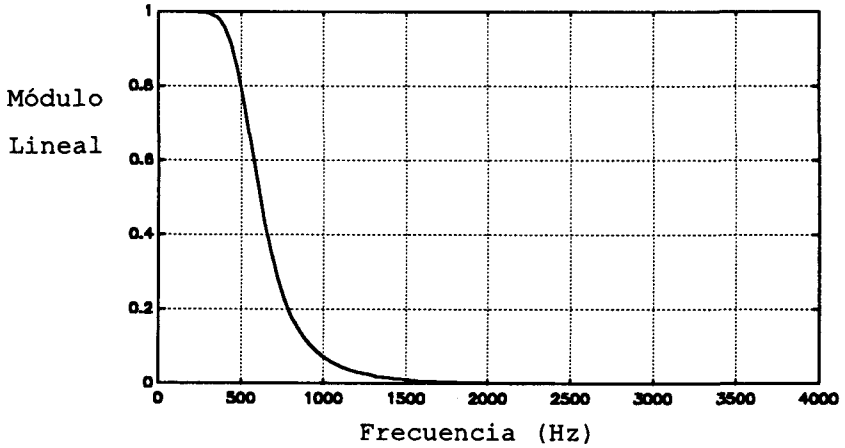
Si por ejemplo diseñamos un filtro de Butterworth paso bajo para una frecuencia de muestreo de 8000Hz cuyas características son:

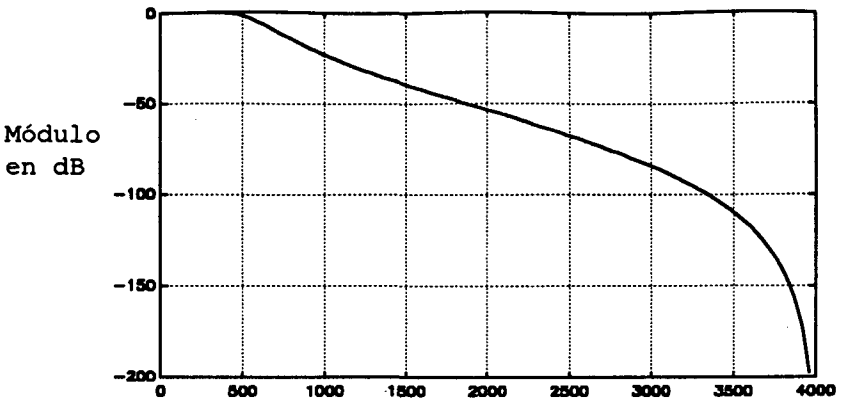
Frecuencia de paso 400 Hz con atenuación máxima de 1 dB.

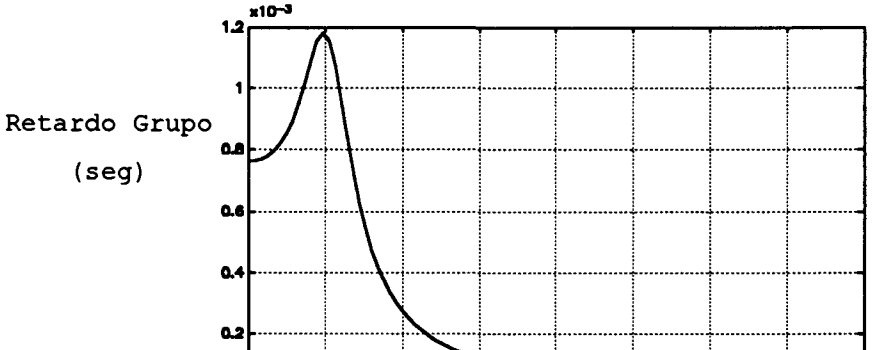
Frecuencia de atenuación 700 Hz con una atenuación mínima 10 dB.

Y le hacemos pasar una señal de voz que se encuentra en el disco muestreada a 8000 Hz los resultados del programa son los siguientes:

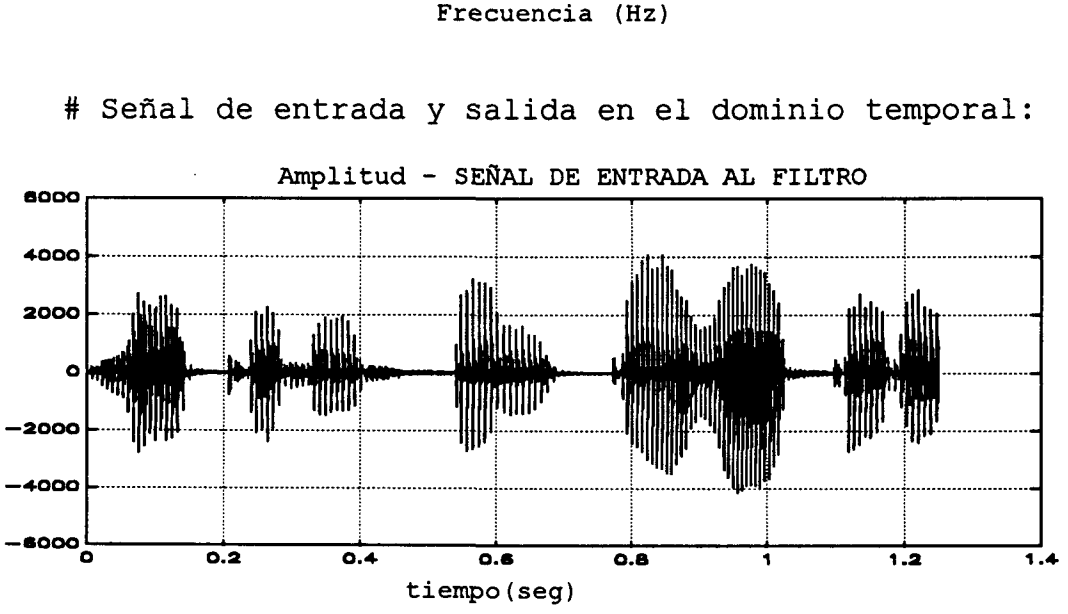
Características del filtro:

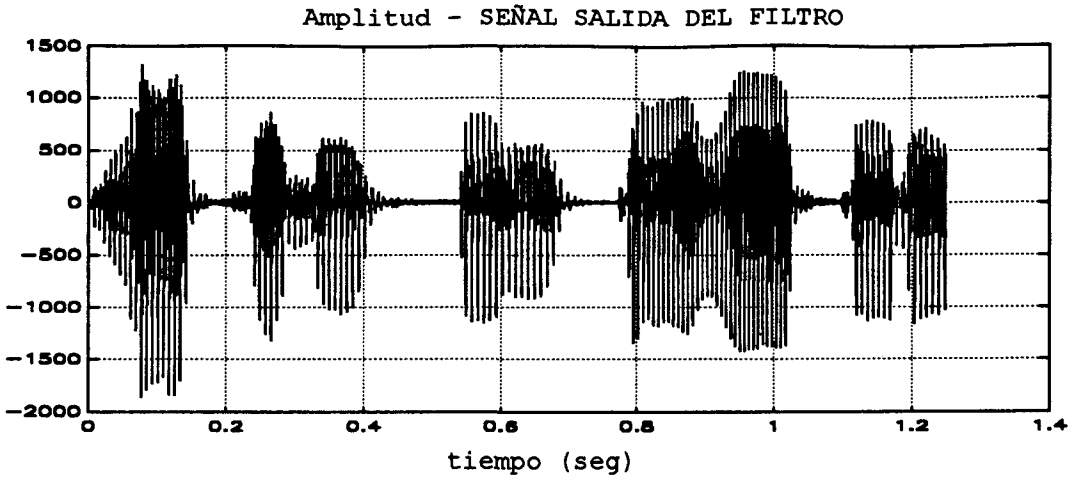




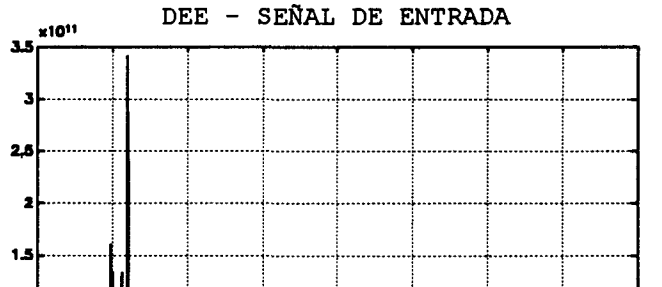


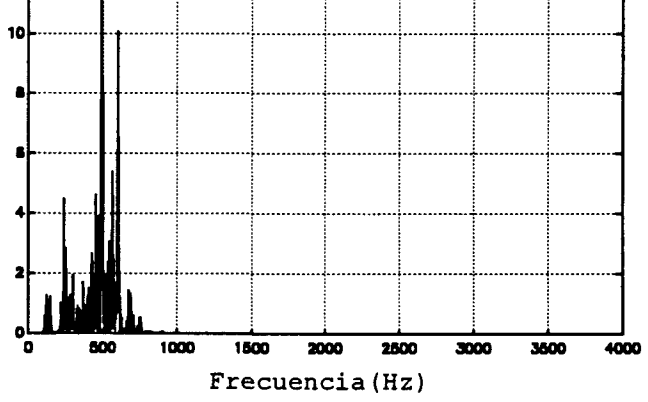
Señal de entrada y salida en el dominio temporal:





DEE de la señal de entrada y salida:





3.3:ESTUDIO DE LOS FILTROS DIGITALES EN EL DSP TMDS3200051.

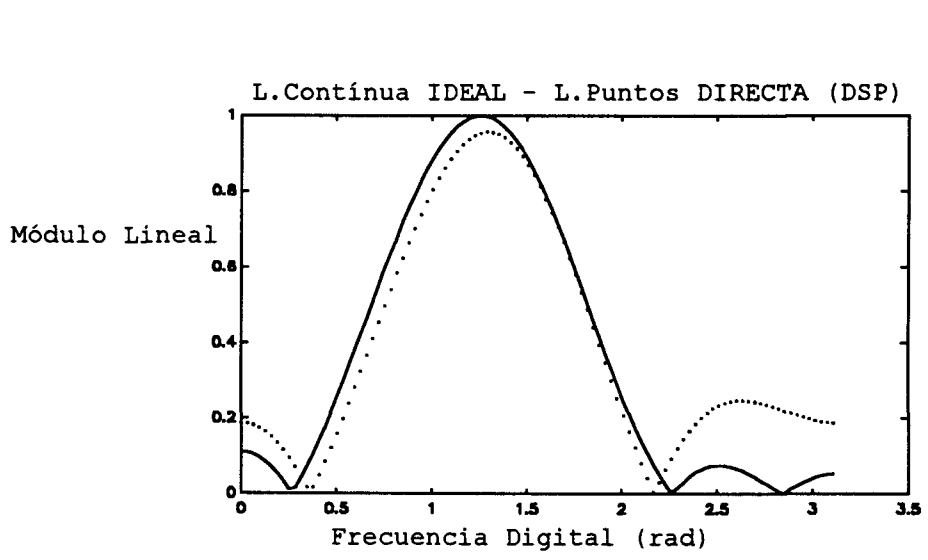
En el programa este estudio se puede llevar a cabo mediante las dos aplicaciones siguientes:

 # Simular los filtros en el DSP TMDS3200051.
 # Filtro analógico global del DSP TMDS3200051.

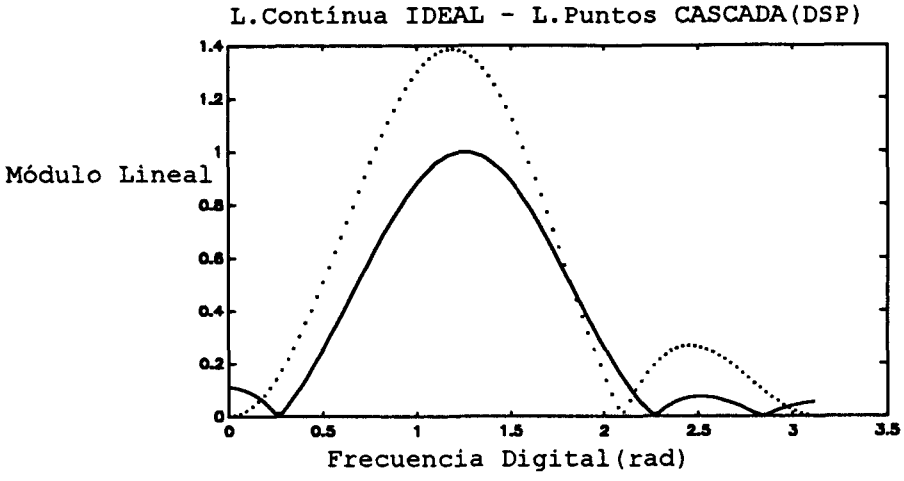
En la aplicación **"Simular los filtros en el DSP TMDS3200051"** lo que se lleva a cabo es una comparación gráfica de las características entre los filtros que se han diseñado con precisión Matlab y los filtros que van a funcionar en el DSP (los filtros del DSP van a sufrir una cuantificación de los coeficientes).El programa muestra las características ideales junto con las realizaciones directa cascada y paralelo según corresponda para filtro FIR/IIR . El hecho de mostrar simultáneamente las diferentes realizaciones con la característica real es para que el usuario elija la realización que más se ajusta a la ideal para más tarde trabajar en el DSP.

Si por ejemplo diseñamos un filtro FIR paso banda con ventana rectangular de orden 10 y con frecuencias de corte de 0.2*pi y 0.6*pi el programa nos dará unas gráficas de salida como las siguientes (aquí mostraremos sólo el módulo lineal , el programa también representa el módulo en dB y la fase).

Comparación filtro ideal con filtro en el DSP para realización directa:



Comparación filtro ideal con filtro en el DSP para realización cascada:



Una vez que el programa muestra las gráficas del filtro ideal con las de las realizaciones en DIRECTA , CASCADA y PARALELO según corresponda a filtros FIR o IIR el usuario tendrá la posibilidad de decidir que filtro es el más que le conviene.

En la aplicación **"Filtro Analógico global del DSP TMDS3200051"** podemos analizar como se va comportar el DSP TMDS3200051 desde la entrada analógica hasta la salida analógica , es decir , como se va comportar el sistema global . Lo primero que nos pedirá el programa es qué filtro queremos simular de los diseñados o especificados en los apartados anteriores , seguidamente nos preguntará por las frecuencias de muestreo para el DSP que van de unos valores típicos de 15 KHz hasta 100Hz y por último según sea el filtro FIR o IIR nos preguntará que realización queremos simular DIRECTA , CASCADA o PARALELO según corresponda . El análisis nos va a arrojar un resultado gráfico mostrándonos el módulo del filtro lineal , el módulo en dB y la fase todo ello referido a la frecuencia analógica.

3.4:ANÁLISIS DEL FILTRADO DE SEÑALES DE TIEMPO CONTINUO CON EL DSP TMDS3200051.

Este tipo de análisis lo podemos hacer con la aplicación del programa **Simular con señales digitalizadas en el DSP TMDS3200051** y en ésta podemos realizar los siguientes estudios:

Al principio el programa nos da a escoger la señal que queremos poner en la entrada del filtro:

Señal de ECG.

Señal de VOZ.
Otra señal guardada en disco.
Ultima señal tratada.

Estas señales se pueden utilizar de manera análoga a como se utilizaban en el **apartado 3.2** con la salvedad de que ahora las señales digitalizadas deben estar codificadas con 14 bits (13 bits de módulo +1 bit de signo) ya que es así como las trata el DSP TMDS3200051 . Igual que explicamos en el apartado 3.2 las señales pueden ser cambiadas por otras nuevas para un análisis concreto o para sacar los resultados de un determinado procesado para ello sólo basta con utilizar los ficheros con formato Matlab siguientes:

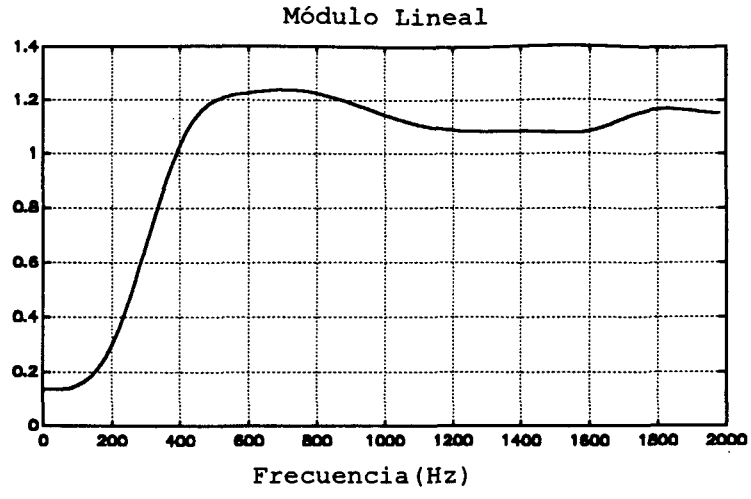
Señal de ECG ; fichero **ecgt.mat**
Señal de VOZ ; fichero **vozt.mat**
Otra señal en disco ; fichero **otra_st.mat**
Ultima señal tratada ; fichero **ult_st.mat**

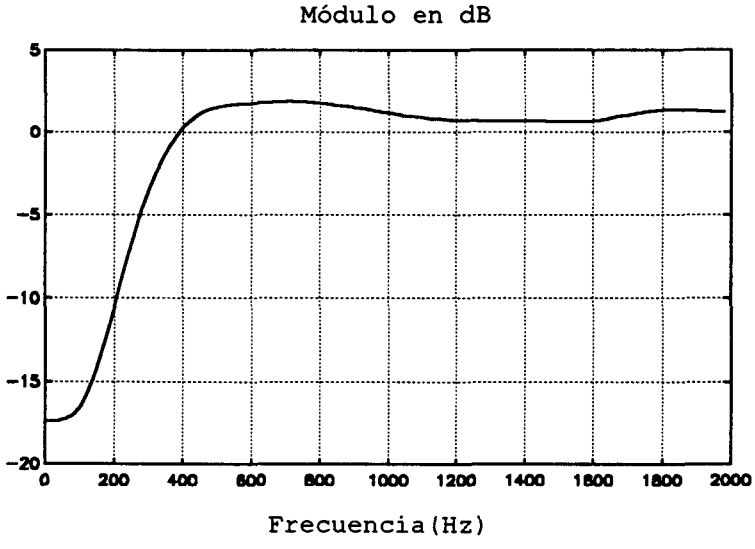
Una vez que le indiquemos al programa que señal vamos a procesar este nos pedirá con que filtro queremos tratarla , luego nos pedirá la frecuencia de muestreo que va típicamente de 15KHz hasta 100Hz y según el tipo de filtro elegido nos pedirá el tipo de realización a utilizar . Como resultado del análisis el programa arroja los siguientes resultados gráficos:

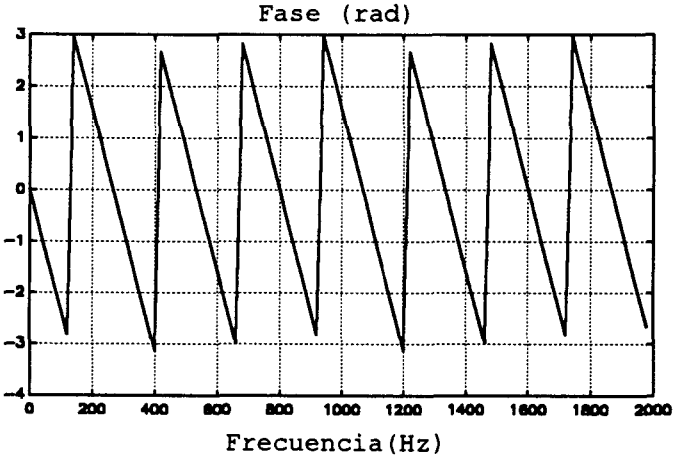
Primeramente muestra las características del filtro que se está utilizando mostrando módulo lineal , módulo en dB y fase todo esto en función de la frecuencia analógica . A continuación muestra las señales de entrada y salida en el dominio temporal pudiendo hacer un ZOOM para observar mejor la señal en estudio . Por último muestra la DEE de la señal de entrada frente a la DEE de la señal de salida con la posibilidad de hacer un ZOOM como en el apartado anterior.

Si por ejemplo hemos diseñado un filtro FIR HAMMING paso alto de orden 30 con una frecuencia de muestreo de 4000 Hz y una frecuencia de corte de 300 Hz y lo atacamos con una señal arbitraria(En este caso son dos tonos) y simulamos la realización directa el DSP tendrá una respuesta como la siguiente:

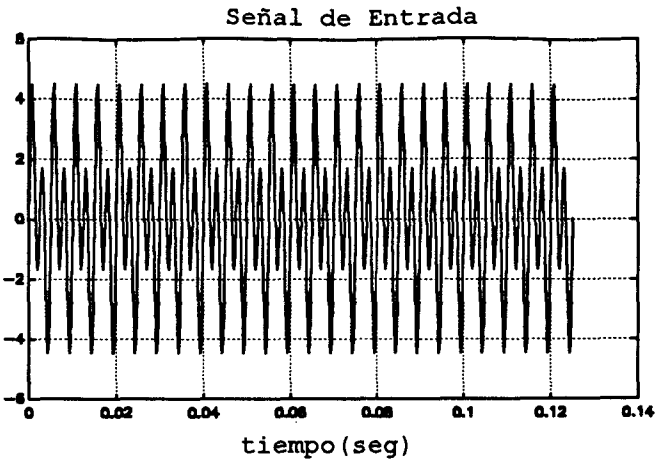
La respuesta global del DSP será:

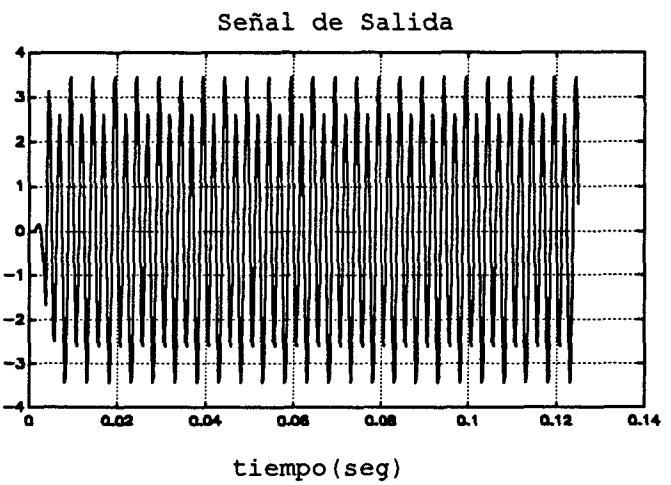




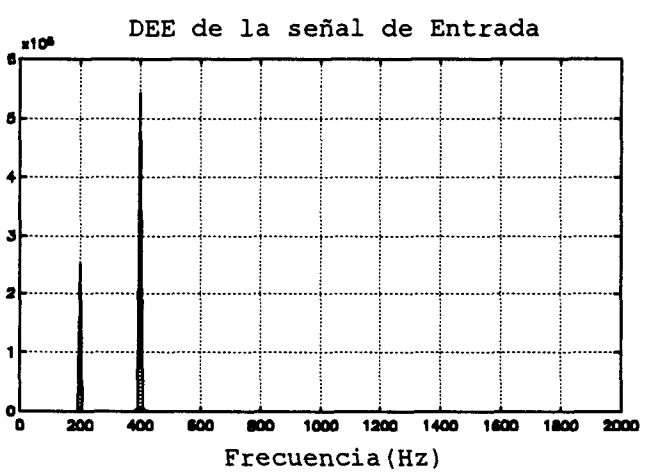


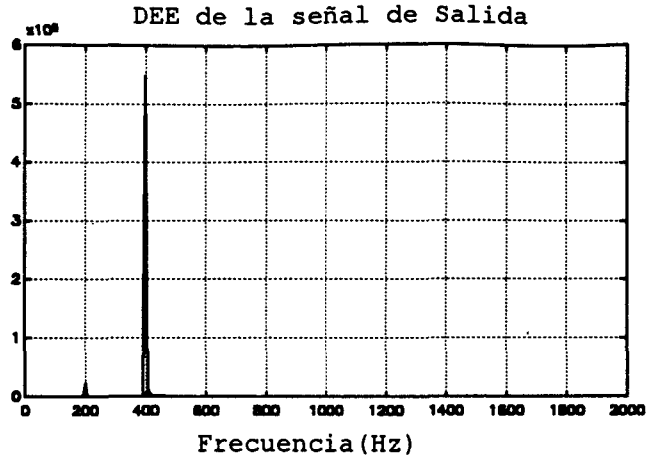
La señal de entrada y salida en el dominio temporal son:





La DEE de la señal de entrada y salida son:





Como se observa en esta aplicación podemos saber cómo se va comportar el DSP ante una señal de entrada concreta sin tener que trabajar con éste.

Un procedimiento bastante importante que podemos realizar tanto en esta aplicación como la descrita en el apartado 3.2 es que si no tenemos señales digitalizadas con las que queremos trabajar podemos sintetizarlas con un lenguaje como el MATLAB.

3.5:AYUDA A LA PROGRAMACIÓN DEL DSP TMDS3200051 Y AL PROGRAMA "DISFILT".

En este apartado se van a describir las siguientes aplicaciones del programa "DISFILT":

Programación del DSP TMDS3200051.
Ayuda para utilizar el programa.

La aplicación **Programación del DSP TMDS3200051** nos dará la posibilidad de programar los conversores A/D y D/A del DSP así como los filtros que hayamos diseñado con el programa DISFILT . La manera de hacerlo es la siguiente:

En el presente Trabajo Fin de Carrera se han desarrollado unos programas en ensamblador para el DSP TMDS3200051 los cuales ya tienen rutinas para programar los conversores A/D y D/A y otras para programar filtros FIR como IIR . Para que estos algoritmos funcionen sólo les faltan unos valores que se deben poner en memoria para indicarle las frecuencias de muestreo de los conversores y los coeficientes de los filtros en cuestión que queramos implementar . Por consiguiente esta aplicación nos dará los valores que se deben poner en memoria para una aplicación concreta y así ya tendremos los algoritmos programados en el DSP.

Al entrar en esta aplicación nos pedirá qué queremos programar del DSP:

Conversores A/D y D/A.
Filtro FIR diseñado.

Filtro IIR diseñado.
Filtro Especificado.

Si elegimos conversores A/D y D/A nos pedirá a qué frecuencia de muestreo queremos que trabajen teniendo un margen que va desde 100 Hz hasta los 15 KHz , una vez elegida esta nos mostrará que valores deben tener los registros **R_TCR** , **P_PRD** , **TA** , **RA** , **TB** y **RB** que serán los valores que tengamos que poner en la memoria del DSP . También nos dirá las frecuencias mínima y máxima de entrada que podemos procesar con esta frecuencia de muestreo.

Cuando elijamos programar tanto filtro FIR , IIR o Especificado , el programa nos dará los valores de los coeficientes de dicho filtro , es decir , los coeficientes de la ecuación en diferencias que lo caracteriza y que tiene la siguiente forma:

$$\sum_{k=0}^{N} a_k y[n-k] = \sum_{k=0}^{M} b_k x[n-k]$$

Si por ejemplo diseñamos un filtro con las siguientes especificaciones:

Aproximación: **Butterworth paso bajo.**
Frecuencia muestreo: **8 KHz.**
Frecuencia de paso: **2 KHz.**
Frecuencia de atenuación: **3 KHz.**
Atenuación máxima banda de paso: **1 dB.**
Atenuación mínima banda atenuada: **6 dB.**

Si una vez realizado un análisis con DISFILT nos interesa programarlo por ejemplo con la realización directa el programa nos arrojará los siguientes resultados:

Conversores A/D y D/A:

Registro R_TCR: **20h.**
Registro P_PRD: **01h.**
Registros TA y RA: **16.**
Registros TB y RB: **41.**
Frecuencia mínima de entrada: **230 Hz.**
Frecuencia máxima de entrada: **4 KHz.**

Filtro IIR diseñado:

Los coeficientes de x[n-k]:

b0:	2	16380	1
b1:	4	0	0
b2:	2	16380	1

Los coeficientes de y[n-k]:

a0:	0	1	0
a1:	2	16380	1
a2:	2	8188	2

En el capítulo 4 se explica más detalladamente los procedimientos para programar el DSP TMDS3200051.

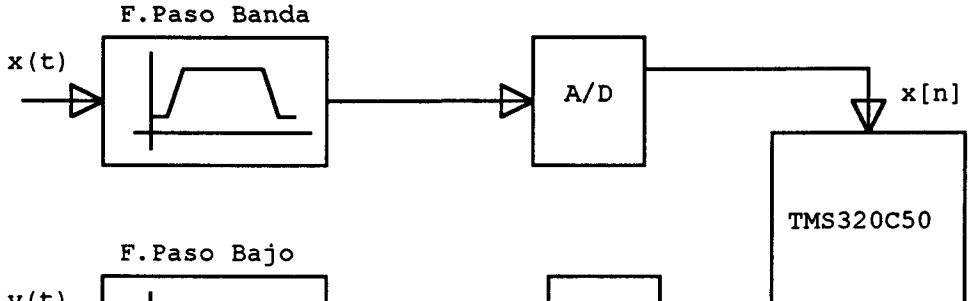
En la aplicación **ayuda para utilizar el programa** aparecen unos menús de ayuda para cada una de las aplicaciones de DISFILT donde se indica de manera somera los procedimientos que se pueden realizar . Para mayor información consultar el anexo **manual de usuario.**

CAPITULO 4: PROGRAMACIÓN DEL DSP TMDS3200051.

En este capítulo se va a describir como programar el DSP TMDS3200051 para trabajar procesando señales de tiempo continuo y en tiempo real:

4.1: INICIALIZACIÓN DEL DSP TMDS3200051 PARA TRABAJAR COMO FILTRO EN TIEMPO REAL.

El DSP TMDS3200051 tiene una estructura como la siguiente:



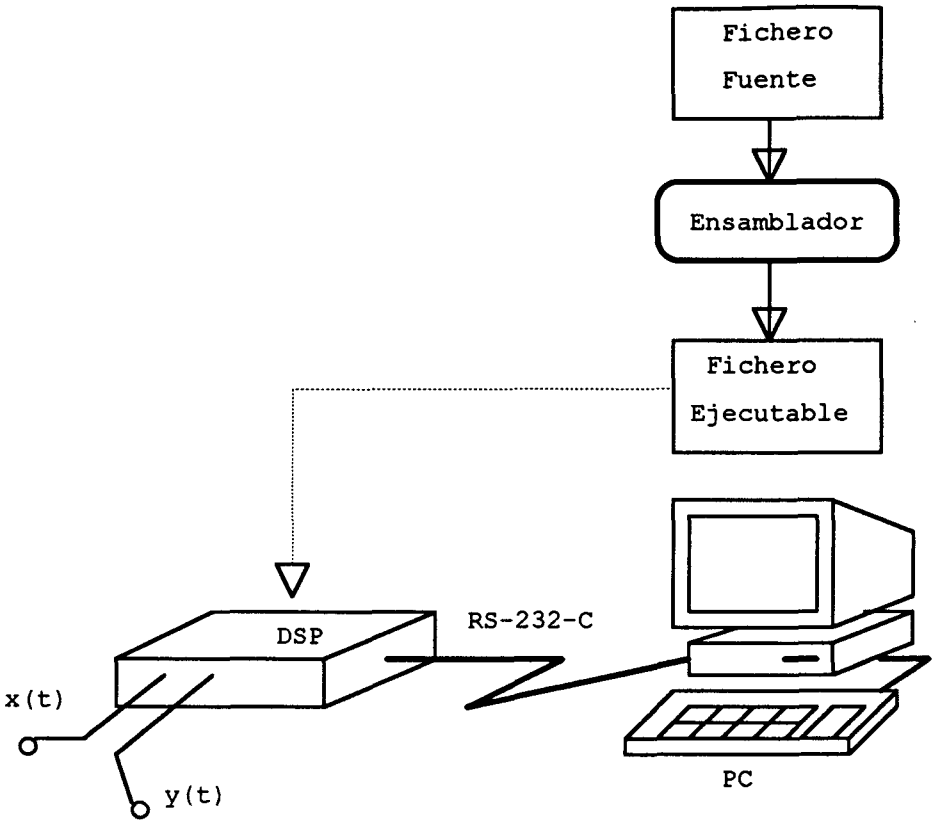
DSP TMDS3200051

Los conversores A/D y D/A se programan conjuntamente con los filtros paso banda de entrada y el paso bajo de salida.

La secuencia para poder implementar un algoritmo de procesado en el DSP es la siguiente:

El programa fuente se elabora y se compila en un PC y una vez existe el fichero ejecutable se carga a través del puerto serie RS-232C en el DSP TMDS3200051 y una vez hecho esto el DSP es independiente y puede trabajar autónomamente.

El diagrama de programación sigue la siguiente estructura:



La manera de hacer la secuencia en el PC es la siguiente:

Generamos el fichero fuente en ensamblador de TMS320C50 nom_fichero.**ASM** con el editor del DOS.

Más tarde lo ensamblamos con el ensamblador del TMS320C50 del la siguiente manera:

DSK5A nom_fichero.**ASM**

generando el fichero ejecutable nom_fichero.**DSK**

Por último sólo nos queda cargarlo en el DSP mediante el comando:

DSK5L nom_fichero.**DSK**

Una vez hecho esto el DSP se puede desconectar del PC y estará ejecutando el programa cargado hasta que le falte la alimentación.

En el apartado 4.3 se describirá con detalle cómo compilar un fichero que contenga ya un filtro cargado en memoria para trabajar en tiempo real.

En el presente TFC se ha desarrollado un fichero fuente en ensamblador que contiene tres rutinas fundamentales:

Rutina de inicialización del DSP para trabajar en tiempo real.
Rutina de filtro FIR.
Rutina de filtro IIR.

En este apartado describiremos la primera rutina y en el apartado 4.2 se describirán las dos siguientes.

Antes de comenzar a describir dicha rutina veamos la estructura del programa fuente en ensamblador.

%%% DECLARACIÓN DE VARIABLES EN MEMORIA %%%

.
.
.

%%% DECLARACIÓN DE VECTORES DE INTERRUPCIÓN %%%

.
.
.

%%% INICIALIZACIÓN DEL TMS320C50 %%%

.
.
.

%%% RUTINA DE INICIACIÓN DEL AIC (A/D y D/A) %%%

.
.
.

%%% RUTINA DE DIVISIÓN CON DESPLAZAMIENTOS A LA DERECHA %%%

.
.
.

%%% RUTINA DE FILTRO FIR %%%

.
.
.

%%% RUTINA FILTRO IIR ESTRUCTURA TIPO II %%%

.
.
.

%%% RUTINA DE SERVICIO PARA INT. DE RECEPCIÓN DEL PUERTO SERIE %%%

Cuando vayamos a implementar un filtro sólo vamos a tener que escribir en la **declaración de variables en memoria** y en la **rutina de servicio para interrupción de recepción del puerto serie** . El usuario debe escribir en estas rutinas ya que según sea el filtro o filtros a implementar debemos guardar unos coeficientes u otros y también según sea la frecuencia de muestreo a utilizar. El resto de rutinas serán siempre iguales independiente del filtro a utilizar.

Para hacer trabajar el DSP como filtro en tiempo real tenemos que colocar en memoria los valores de los registros **TA , RA , TB , RB , R_TCR** y **R_PRD** con los valores que nos haya indicado DISFILT según la frecuencia de muestreo con que vayamos a trabajar.

La rutina declaración de variables tiene la siguiente forma:

```
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%%%%%%%%%%%%%%%%%% DECLARACIÓN DE VARIABLES EN MEMORIA %%%%%%%%%%%%%%%%%%%%
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
          .mmregs
          .ds     0f00h ; comienzo memoria datos (0f00h--2bc0h)
TA        .word
RA        .word
TB        .word           ;** Valores para frec. muestreo y filtros BP y LP **
RB        .word              ;( Ver tablas de diseño de filtros )
R_TCR     .word
R_PRD     .word
AIC_CTR   .word   29h


                      ; ** Variables temporales de los filtros  **
                      ; ** Poner los filtros en páginas contiguas **
                        ;( Ver tablas de diseño de filtros )
```

Al lado de la instrucción **.word** ponemos los valores que nos haya dado DISFILT.

4.2:PROGRAMACIÓN DE LOS FILTROS FIR/IIR.

Para programar los filtros en el DSP tenemos que poner en memoria los coeficientes y algunos valores auxiliares según se muestra a continuación.

Si por ejemplo queremos diseñar un filtro FIR con los siguientes parámetros:

- **# FIR ventana HAMMING paso banda.**
- **# Orden 8.**
- **# Frecuencia de corte inferior digital 0.4*π.**
- **# Frecuencia de corte superior digital 0.6*π.**
- **# Realización en cascada.**
- **# Frecuencia de muestreo de 6 KHz.**

Nos queda en la **declaración de variables** una cosa como la siguiente:

```
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%%%%%%%%%%%%%%%%%% DECLARACION DE VARIABLES EN MEMORIA %%%%%%%%%%%%%%%%%%%%
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
          .mmregs
          .ds     0f00h ; comienzo memoria datos (0f00h--2bc0h)
TA        .word   21
RA        .word   21
TB        .word   41        ;** Valores para frec. muestreo y filtros BP y LP **
RB        .word   41           ;( Ver tablas de diseño de filtros )
R_TCR     .word   20h
R_PRD     .word   01h
AIC_CTR   .word   29h
```

```
          .ds 1000h
F1        .word    0
SA1       .word    0,0
ORD1      .word    2,2
DIC1      .word    100ah,100ch
A01       .word    0,1,0
S01       .word    0,0
B01       .word    1,2044,4
B11       .word    2,1022,5

          .ds 1100h
F2        .word    0
SA2       .word    0,0
ORD2      .word    2,2
DIC2      .word    110ah,110ch
A02       .word    0,1,0
S02       .word    0,0
B02       .word    1,16380,1
B12       .word    4,0,0

          .ds  1200h
F3        .word    0
SA3       .word    0,0
ORD3      .word    2,2
DIC3      .word    120ah,120ch
A03       .word    0,1,0
S03       .word    0,0
B03       .word    1,16380,1
B13       .word    4,0,0

          .ds  1300h
F4        .word    0
SA4       .word    0,0
ORD4      .word    2,2
DIC4      .word    130ah,130ch
A04       .word    0,1,0
S04       .word    0,0
B04       .word    2,16380,1
B14       .word    4,0,0

          .ds  1400h
F5        .word    0
SA5       .word    0,0
ORD5      .word    2,2
DIC5      .word    140ah,140ch
A05       .word    0,1,0
S05       .word    0,0
B05       .word    2,16380,1
B15       .word    4,0,0

          .ds 1500h
F6        .word    0
SA6       .word    0,0
ORD6      .word    2,2
DIC6      .word    150ah,150ch
A06       .word    1,2,0
S06       .word    0,0
B06       .word    3,0,0
B16       .word    2,16380,1h

          .ds 1600h
F7        .word    0
SA7       .word    0,0
ORD7      .word    2,2
DIC7      .word    160ah,160ch
A07       .word    1,2,0
S07       .word    0,0
B07       .word    4,0,0
B17       .word    2,16380,1h

          .ds 1700h
F8        .word    0
SA8       .word    0,0
ORD8      .word    2,2
DIC8      .word    170ah,170ch
A08       .word    1,2,0
S08       .word    0,0
B08       .word    4,0,0
```

```
B18         .word    2,16380,1h
```

y en la **rutina de servicio para la interrupción de recepción del puerto serie** una como la siguiente:

```
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%%%%%% RUTINA DE SERVICIO PARA LA INT. DE RECEPCION DEL PUERTO SERIE %%%%%%
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
RECEPCION:
        LAMM    DRR  ; Muestra llegada del AIC al Acumulador


          ;*** Algoritmos para diseño de filtros en tiempo real***
          ; ( Ver tablas de coeficientes y de algoritmos de filtros)

        LDP  #F1
        CALL FILTRO_FIR
        LDP  #F2
        CALL FILTRO_FIR
        LDP  #F3
        CALL FILTRO_FIR
        LDP  #F4
        CALL FILTRO_FIR
        LDP  #F5
        CALL FILTRO_FIR
        LDP  #F6
        CALL FILTRO_FIR
        LDP  #F7
        CALL FILTRO_FIR
        LDP  #F8
        CALL FILTRO_FIR

        SAMM    DXR  ; Muestra del Acumulador al AIC
        RETE ; volvemos al bucle de espera hasta la próxima muestra

        .end
```

Veamos lo que significa cada cosa de las líneas de código anteriores:

En la declaración de variables aparecen una primera parte que es la explicada anteriormente para los conversores A/D y D/A y luego ocho bloques que corresponden a los ocho filtros de la realización en cascada que son un resultado arrojado por DISFILT.Veamos lo que significa cada término de ese bloque:

Por ejemplo el primer bloque tiene la siguiente forma:

```
        .ds 1000h
F1      .word    0
SA1     .word    0,0
ORD1    .word    2,2
DIC1    .word    100ah,100ch
A01     .word    0,1,0
S01     .word    0,0
B01     .word    1,2044,4
B11     .word    2,1022,5
```

La primera línea **.ds 1000h** corresponde a la página de memoria donde están los coeficientes del filtro , se deben coger páginas consecutivas , es decir , 1000h , 1100h , 1200h, etc. siempre empezando por la 1000h . El resto significan lo siguiente:

SA1: debe tener dos valores a cero.(señales internas del filtro).

ORD1: debe tener dos valores e iguales al orden del filtro.

A01: valor del coeficiente a0 del filtro obtenido de DISFILT.
B01-B11: resto de coeficientes bn del filtro obtenido de DISFILT.
S01: tantos valores a cero como el orden del filtro (señales internas del filtro).
DIC1: dirección absoluta del coeficiente S01 y B01 y para este caso se ve que valen 100ah y 100ch

En la rutina de servicio para la interrupción de recepción del puerto serie aparecen las ocho llamadas a los ocho filtros en cascada , para un solo filtro habría que direccionar sus coeficientes con **LDP #nom_filtro** y luego llamar al filtro **CALL FILTRO_FIR** .

```
LDP  #F1
CALL FILTRO_FIR
```

Para llevar a cabo la realización en cascada basta con ir llamando los filtros uno detrás de otro ya que la señal de salida de cada uno se guarda en el acumulador del DSP y es la que sirve de entrada para el siguiente según como muestra el ejemplo anterior cuyo código es el siguiente:

```
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%%%%%% RUTINA DE SERVICIO PARA LA INT. DE RECEPCION DEL PUERTO SERIE %%%%%%
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
RECEPCION:
        LAMM    DRR  ; Muestra llegada del AIC al Acumulador


          ;*** Algoritmos para diseño de filtros en tiempo real***
          ; ( Ver tablas de coeficientes y de algoritmos de filtros)

        LDP  #F1
        CALL FILTRO_FIR
        LDP  #F2
        CALL FILTRO_FIR
        LDP  #F3
        CALL FILTRO_FIR
        LDP  #F4
        CALL FILTRO_FIR
        LDP  #F5
        CALL FILTRO_FIR
        LDP  #F6
        CALL FILTRO_FIR
        LDP  #F7
        CALL FILTRO_FIR
        LDP  #F8
        CALL FILTRO_FIR

        SAMM    DXR  ; Muestra del Acumulador al AIC
        RETE ; volvemos al bucle de espera hasta la próxima muestra

       .end
```

Para llevar a cabo la realizacíon directa es sencilla ya que solo hay llamar a un solo filtro y para la realización en paralelo lo veremos a continuación a la vez que explicamos como implentar un filtro IIR . Para ello vamos a tomar el siguiente filtro:

 # Filtro BUTTERWORTH Paso Bajo.
 # Frecuencia de paso digital π/4.

Frecuencia de atenuación digital π/2.
Atenuación máxima banda de paso 1 dB.
Atenuación mínima atenuada 9 dB.
Frecuencia de muestreo 6 KHz.

Nos queda en la **declaración de variables** una cosa como la siguiente:

```
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%%%%%%%%%%%%%%%%%% DECLARACION DE VARIABLES EN MEMORIA %%%%%%%%%%%%%%%%%%%%
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
          .mmregs
          .ds     0f00h ; comienzo memoria datos (0f00h--2bc0h)
TA        .word   21
RA        .word   21
TB        .word   41          ;** Valores para frec. muestreo y filtros BP y LP **
RB        .word   41             ;( Ver tablas de diseño de filtros )
R_TCR     .word   20h
R_PRD     .word   01h
AIC_CTR   .word   29h

          .ds 1000h
F1        .word    0
SA        .word    0,0,0
ORDA      .word    3,3
ORDB      .word    2,2
ORD       .word    3
DIREC     .word    100ch,100fh,1018h
S0        .word    0,0,0
A0        .word    0,1,0
A1        .word    1,16380,1
A2        .word    2,8188,2
B0        .word    1,16380,1
B1        .word    2,8188,1

VAL       .word    0
```

y en la **rutina de servicio para la interrupción de recepción del puerto serie** una como la siguiente:

```
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%%%%%% RUTINA DE SERVICIO PARA LA INT. DE RECEPCION DEL PUERTO SERIE %%%%%%
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
RECEPCION:
         LAMM    DRR  ; Muestra llegada del AIC al Acumulador


           ;*** Algoritmos para diseño de filtros en tiempo real***
           ; ( Ver tablas de coeficientes y de algoritmos de filtros)

         LDP #F1
         SACL VAL
         CALL FILTRO_IIR
         ADD  VAL
         SAMM    DXR  ; Muestra del Acumulador al AIC
         RETE ; volvemos al bucle de espera hasta la próxima muestra

         .end
```

En este caso la realización en paralelo consiste en un filtro IIR de orden 3 y una constante de valor 1 . Veamos lo que significan las líneas de código en la declaración de variables:

Aparece una primera parte que es lo de los conversores A/D y D/A que ya se ha comentado anteriormente luego están los coeficientes del filtro:

```
        .ds 1000h
F1      .word   0
SA      .word   0,0,0
ORDA    .word   3,3
ORDB    .word   2,2
ORD     .word   3
DIREC   .word   100ch,100fh,1018h
S0      .word   0,0,0
A0      .word   0,1,0
A1      .word   1,16380,1
A2      .word   2,8188,2
B0      .word   1,16380,1
B1      .word   2,8188,1

VAL     .word   0
```

nuevamente el filtro está en la página indicada por **.ds 1000h** , el resto tiene el siguiente significado:

F1: nombre del filtro.
SA: tres valores a cero para señales internas del filtro.
ORA: dos valores iguales indicando el número de términos an.(para el ejemplo 3).
ORB: dos valores iguales indicando el número de términos bn.(para el ejemplo 2).
ORD: un valor con el máximo de ORA y ORB .(para el ejemplo 3-orden del filtro).
S0: tantos términos a cero como indica el orden del filtro.
A0-A2: Valores de los coeficientes an arrojados por DISFILT.
B0-B1: Valores de los coeficientes bn arrojados por DISFILT.
DIREC: Direcciones absolutas del coeficiente S0 , A0 y B0. para el ejemplo tenemos 100ch , 100fh y 1018h respectivamente.

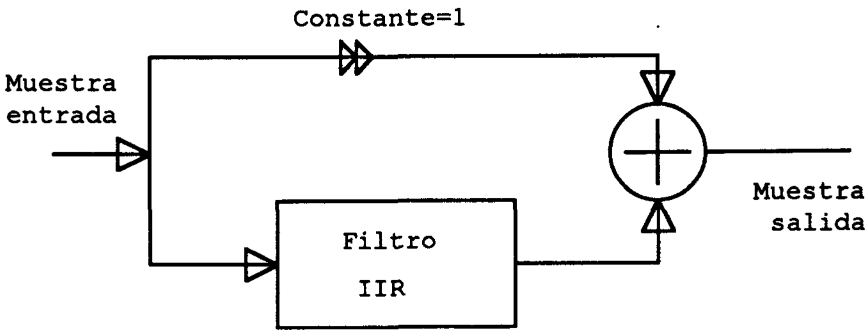
Las líneas de código que aparecen en la rutina de servicio para la interrupción de recepción del puerto serie tienen el siguiente significado:

```
LDP #F1
SACL VAL
CALL FILTRO_IIR
ADD  VAL
SAMM    DXR  ; Muestra del Acumulador al AIC
RETE ; volvemos al bucle de espera hasta la próxima muestra

.end
```

En la declaración de variables en memoria pusimos una variable denominada **VAL** en la misma página de memoria que el filtro que se va a utilizar para la señal debido a la constante de la realización en paralelo . El código hace lo siguiente:

SACL VAL Guarda el valor de la muestra actual en val.
CALL FILTRO_IIR Calculamos la salida del filtro IIR
ADD VAL y finalmente sumamos la salida del filtro con la muestra actual obteniendo el resultado de la realización en paralelo.

Esquemáticamente sería:



Tanto al diseñar filtros FIR como IIR con una realización en cascada si el margen dinámico de salida se ve reducido porque el sistema se satura para una señal de entrada pequeña es aconsejable probar a cambiar el orden en los filtros de la cascada , simplemente invocándolos en orden diferente al actual . Si por ejemplo tenemos la siguiente cascada:

```
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%%%%%% RUTINA DE SERVICIO PARA LA INT. DE RECEPCION DEL PUERTO SERIE %%%%%%
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
RECEPCION:
        LAMM    DRR  ; Muestra llegada del AIC al Acumulador


          ;*** Algoritmos para diseño de filtros en tiempo real***
          ; ( Ver tablas de coeficientes y de algoritmos de filtros)

        LDP  #F1
        CALL FILTRO_FIR
        LDP  #F2
        CALL FILTRO_FIR
        LDP  #F3
        CALL FILTRO_FIR
        LDP  #F4
        CALL FILTRO_FIR
        LDP  #F5
        CALL FILTRO_FIR
        LDP  #F6
        CALL FILTRO_FIR
        LDP  #F7
        CALL FILTRO_FIR
        LDP  #F8
        CALL FILTRO_FIR

        SAMM    DXR  ; Muestra del Acumulador al AIC
        RETE ; volvemos al bucle de espera hasta la próxima muestra

       .end
```

Podemos poner al principio de la cascada los filtros 7 y 8 simplemente escribiendo:

```
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%%%%%% RUTINA DE SERVICIO PARA LA INT. DE RECEPCION DEL PUERTO SERIE %%%%%%
```

```
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
RECEPCION:
        LAMM    DRR  ; Muestra llegada del AIC al Acumulador


          ;*** Algoritmos para diseño de filtros en tiempo real***
          ; ( Ver tablas de coeficientes y de algoritmos de filtros)

        LDP  #F7
        CALL FILTRO_FIR
        LDP  #F8
        CALL FILTRO_FIR
        LDP  #F1
        CALL FILTRO_FIR
        LDP  #F2
        CALL FILTRO_FIR
        LDP  #F3
        CALL FILTRO_FIR
        LDP  #F4
        CALL FILTRO_FIR
        LDP  #F5
        CALL FILTRO_FIR
        LDP  #F6
        CALL FILTRO_FIR

        SAMM    DXR  ; Muestra del Acumulador al AIC
        RETE ; volvemos al bucle de espera hasta la próxima muestra

        .end
```

4.3:IMPLEMENTACIÓN DE ESTRUCTURAS ALTERNATIVAS DE FILTROS Y ALGORITMOS DE PROCESADO.

En los apartados anteriores hemos visto como implementar filtros FIR o IIR tanto en realizaciones directa , cascada o paralelo según se desee para una aplicación concreta . Una vez llegado a este punto podemos implementar cualquier sistema compuesto por filtros para procesar señales analógicas en tiempo real ya que disponemos de un programa fuente donde existen subprogramas para programar los conversores A/D y D/A , filtros FIR , filtros IIR y una rutina aritmética para realizar divisiones con desplazamientos de bits hacia la derecha que pueden ser utilizadas según los requerimientos del procesado. Para más información consultar el manual [TEXT-7]:TMS320C5x USER´S GUIDE.

El programa fuente se encuentra en el fichero **FIL_TR.FTE** en cual debemos escribir en las rutinas antes mencionadas según el filtro que queramos implementar . Una vez hecho esto al fichero lo guardamos con el nombre que deseemos y una extensión **.ASM** luego lo ensamblamos con el comando:

DSK5A nom_fich.**ASM**

Generando el fichero nom_fich.**DSK** que es el ejecutable

Y por último lo cargamos en el DSP mediante el comando:

DSK5L nom_fich.**DSK**

Para hacer las modificaciones en el fichero fuente es aconsejable utilizar el editor del DOS **EDIT.COM** puesto que si utilizamos otro editor más complejo corremos el riesgo de introducir caracteres extraños que al ensamblar nos puede ocasionar problemas.

El fichero fuente **FIL_TR.FTE** tiene el siguiente listado:

```
;*******************************************************************************
;*******************************************************************************
;***   ESTRUCTURA GENERAL PARA IMPLEMENTAR FILTROS DIGITALES EN TIEMPO   ***
;***                    REAL CON EL DSP TMS32200051                      ***
;*******************************************************************************
;*******************************************************************************

;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%%%%%%%%%%%%%%%%%%% DECLARACION DE VARIABLES EN MEMORIA %%%%%%%%%%%%%%%%%%%%%
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
          .mmregs
          .ds     0f00h ; comienzo memoria datos (0f00h--2bc0h)
TA        .word
RA        .word
TB        .word          ;** Valores para frec. muestreo y filtros BP y LP **
RB        .word             ;( Ver tablas de diseño de filtros )
R_TCR     .word
R_PRD     .word
AIC_CTR   .word  29h


                    ; ** Variables temporales de los filtros  **
                    ; ** Poner los filtros en páginas contiguas **
                      ;( Ver tablas de diseño de filtros )


;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%%%%%%%%%%%%%%%% DECLARACION DE LOS VECTORES DE INTERRUPCION %%%%%%%%%%%%%%%%
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
          .ps     080ah ;memoria vectores int. (0800h--083fh)
rint:     B       RECEPCION     ;0A Recepción del puerto serie RINT.


;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%%%%%%%%%%%%%%%%%%%% INICIALIZACION DEL TMS320C50 %%%%%%%%%%%%%%%%%%%%%%%%%%
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
          .ps 0a00h  ;comienzo memoria de programa (0a00h--2bffh)
          .entry
INICIO:   SETC    INTM
          LDP     #0
          OPL     #0834h,PMST ; configuramos modo de trabajo del TMS320C50
          LACC    #0
          SAMM    CWSR
          SAMM    PDWSR
          SPLK    #022h,IMR
          CALL    AICINIT  ; Llamamos rutina de inicio AIC
          CLRC    OVM
          SPM     0
          SPLK    #012h,IMR
          CLRC    INTM

ESPERA:   NOP               ; Esperamos a que llegue una muestra del AIC
          NOP               ; para pasar a servir la rutina de recepción
          NOP
          B       ESPERA


;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%%%%%%%%%%%%       RUTINA DE INICIALIZACION DEL AIC          %%%%%%%%%%%%%%%
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;*******************************************************************************
;***   RUTINA QUE INICIALIZA EL AIC TLC320C40 SEGUN LOS PARAMETROS    ******
;***              TA, RA,TB,RB,R_PRD,R_TCR y AIC_CTR                    ****
```

```
; ************************************************************************
;***  %% Los valores de RA y RB fijan el filtro paso banda de entrada    ***
;***       y la frecuencia de muestreo del A/D                           ***
;***  %% Los valores de TA y TB fijan el filtro paso bajo de salida      ***
;***       y la frecuencia de muestreo del D/A                           ***
;***  %% Los valores de R_PRD y R_TCR fijan la frecuiencia del M_Clock   ***
;***  %% El valor de AIC_CTR fija el modo de trabajo del AIC             ***
;************************************************************************
;**  El reloj maestro vale  M_Clock=20736 KHz/[(TDDR+1)*(PRD+1)]         ***
;** PRD tiene 16bits y TDDR últimos 4 bits de TCR=2(TDDR)h               ***
;************************************************************************
;** El filtro paso banda tiene los siguientes parámetros:               ***
;**  0 dB en la banda fci=.2 kHz*(SCF/288 KHz);fcs=3.5 KHz*(SCF/288 KHz) ***
;************************************************************************
;** El filtro paso bajo tiene los siguientes parámetros:                ***
;** 0 dB en la banda fci=0 Hz;fcs=3.5 KHz*(SCF/288 KHz)                  ***
;************************************************************************
;** El resto de parámetros para las frecuencias de muestreo y los filtros **
;**              SCF=M_Clock/(2*A)   ;  Fmuestreo=SCF/B                 ***
;************************************************************************
;** El AIC_CTR tiene 6 bits . AIC_CTR=G1 GO SY AX LB BP                 ***
;** G1 GO : ganancia del AIC      G1   GO ------  Ganancia              ***
;**                                1    1          1/4                  ***
;**                                0    0          1/4                  ***
;**                                0    1          1/2                  ***
;**                                1    0           1                   ***
;** SY : recepción transmisión 0/1 asíncrona/síncrona                   ***
;** AX : entrada analógica auxiliar 0/1 deshabilitada/habilitada        ***
;** LB : función lazo entrada salida 0/1 deshabilita/habilita           ***
;** BP : filtro paso banda entrada 0/1 eliminamos/insertamos            ***
;************************************************************************
AICINIT:
        LDP     #R_PRD      ;--------------------------
        LACL    R_PRD
        LDP     #0          ;  Generamos la frecuencia del M_Clock
        SACL    PRD         ;  en función de los valores de R_TCR
        LDP     #R_TCR      ;  y R_PRD
        LACL    R_TCR
        LDP     #0
        SACL    TCR
                            ;--------------------------
        MAR     *,AR0
        LACC    #0008h
        SACL    SPC
        LACC    #00c8h
        SACL    SPC
        LACC    #080h
        SACH    DXR
        SACL    GREG
        LAR     AR0,#0FFFFh
        RPT     #10000
        LACC    *,0,AR0
        SACH    GREG
        ;-----------------------
        LDP     #TA             ;
        SETC    SXM             ;
        LACC    TA,9            ; Inicializamos registro TA y RA
        ADD     RA,2            ;
        CALL    AIC_2ND         ;
        ;-----------------------
        LDP     #TB
        LACC    TB,9            ; Inicializamos registro TB y RB
        ADD     RB,2            ;
        ADD     #02h            ;
        CALL    AIC_2ND         ;
        ;-----------------------
        LDP     #AIC_CTR
        LACC    AIC_CTR,2       ; INICIALIZAMOS EL REGISTRO  DE CONTROL
        ADD     #03h            ; DEL AIC (Circuito Interfaz Analógico)
        CALL    AIC_2ND         ;
        RET                     ;

AIC_2ND:
        LDP     #0
        SACH    DXR
        CLRC    INTM
        IDLE
        ADD     #6h,15
```

```
        SACH    DXR
        IDLE
        SACL    DXR
        IDLE
        LACL    #0
        SACL    DXR
        IDLE
        SETC    INTM
        RET

;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%%%%%%      RUTINA DE DIVIVION CON DESPLAZAMIENTOS A LA DERECHA     %%%%%%%
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
; Al desplazar bits a la derecha  sólo dividirá entre 2,4,8,16,32, ...
; y el resutado es la parte entera. p.e. 9/8=1.125 ----> 1
;%%%%   Los parámetros de entrada se le pasan mediante los registros
;%%%%   auxiliares ARn y el resultado es puesto en estos registros de la
;%%%%   siguiente manera:
;%%%    * Número corrector en AR0
;%%%    * Número de desplazamientos en AR1
;%%%    * Número a dividir en AR2
;%%%    * resultado en ACC
;  La rutina utiliza los registros auxiliares siguientes para realizar
;  las operaciones: AR4,AR5 y AR6.
;%%% Los números correctores que hay que pasarle están en función del número
;%%%  de desplazamientos a realizar y son los siguientes:
; 1 des----3ffch; 2 des----1ffch; 3 des----0ffch; 4 des----07fch;
; 5 des----03fch; 6 des----01fch; 7 des----00fch; 8 des----007ch;
; 9 des----003ch; 10 des----001ch; 11 des----000ch; 12 des----0004h;

DIVISION:
        SST  #0,100 ; guardamos la página actual
        LDP  #0h ; página cero en el puntero de página
        LAMM AR2
        SAMM AR4  ; guardamos el valor a dividir en AR4
        AND #8000h
        SAMM AR6  ; guardamos el signo del valor en AR6
        BCND SIGUE,EQ  ; si es positivo vamos a SIGUE
        LAMM AR4
        AND #7fffh  ; si el valor en negativo estará en complemento a dos
        SUB #1      ; entonces lo tansformamos a binario natural
        CMPL
        SAMM AR5

ROTN:  LAMM AR5     ; OPERAMOS EL NUMERO SI ES NEGATIVO
       ROR          ; desplamos tantas veces a la derecha como indica AR1
       SAMM AR5
       LAMM AR1
       SUB #1
       SAMM AR1
       BCND SIGUE1,EQ
       B ROTN
SIGUE1: LAMM AR5
       AND AR0 ; correjimos con el número corrector y convertimos
       CMPL    ; nuevamente en complemento a dos
       ADD #1
       B SG ; nos vamos al final de la rutina

SIGUE: LAMM AR4 ;OPERAMOS EL NUMERO SI ES POSITIVO
       SAMM AR5

ROTP:  LAMM AR5
       ROR        ;rotamos tantas veces a la derecha como indica AR1
       SAMM AR5
       LAMM AR1
       SUB #1
       SAMM AR1
       BCND SIGUE2,EQ
       B ROTP

SIGUE2: LAMM AR5
       AND AR0  ; correjimos con el número corrector

SG:    LST #0,100 ;recuperamos la página  actual
           ; acabamos la rutina
       RET

;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
;%%%%%%%%%%          RUTINA FILTRO FIR ESTRUCTURA DIRECTA          %%%%%%%%%%%
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%% Este subprograma coge una muestra del acumulador y una vez procesada %%%
;%% devuelve el resultado nuevamente en el acumulador.                    %%%
;%% La manera de proceder es la siguiente:                                %%%
;%%  * direccionamos la página de memoria donde están los coeficientes    %%%
;%%      y las señales auxiliares del filtro en cuestión.                  %%%
;%%  * luego ponemos la muestra a procesar en el acumulador y ejecutamos %%%
;%%      el filtro con " CALL FILTRO_FIR"                                  %%%
;%%  * por último recogemos la muestra procesada en el acumulador         %%%
;%%  Las variables en memoria se ponen de la siguiente manera:             %%%
;%%           .ds    XXXXh  pagina de trabajo                             %%%
;%%            nom_filtro   .word     0                                   %%%
;%%            se_aux       .word     0,0                                 %%%
;%%            orden        .word     X,X                                 %%%
;%%            direc_coef   .word     X(S0),X(B0)                         %%%
;%%            A0           .word     X,X,X                               %%%
;%%            S0           .word     0                                   %%%
;%%            S1           .word     0                                   %%%
;%%            :              :       :                                   %%%
;%%            B0           .word     X,X,X                               %%%
;%%            B1           .word     X,X,X                               %%%
;%%            :              :       :                                   %%%
;%%  Y el programa se ejecuta como:                                        %%%
;%%               LDP  #nom_filtro                                         %%%
;%%               CALL FILTRO_FIR                                          %%%
;%%  (Informacioón más detallada en la documentación)                      %%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

FILTRO_FIR:         ;(* Ver estructura filtro FIR directa *)

              SACL 0ah ; guardamos la muestra que llega en S0
              MAR *,AR7
              LAR  AR7,6 ; AR7 es el puntero del los BN
              LACL *+
              BCND  SD1,EQ ; Si B0 es cero vamos a SD1
              SUB  #1
              BCND  SD2,EQ ; Si B0 es negativo y menor que 1 vamos a SD2
              SUB  #1
              BCND  SD3,EQ ; Si B0 es positivo y menor que 1 vamos a SD3
              SUB  #1
              BCND  SD4,EQ ; Si B0 es -1 vamos a SD4
              B     SD5    ; Si B0 es 1 vamos a SD5
SD1:     LACL 1  ;Operamos con B0 igual a cero
         SACL 2
         B  SD6
SD2:     LAR AR2,0ah ; Operamos con B0 negativo y menor que 1
         LAR AR0,*+
         LAR AR1,*+
         CALL DIVISION
         NEG
         ADD 1
         SACL 2
         B  SD6
SD3:     LAR AR2,0ah  ; Operamos con B0 positivo y menor que 1
         LAR AR0,*+
         LAR AR1,*+
         CALL DIVISION
         ADD 1
         SACL 2
         B  SD6
SD4:     LACL 0ah ; Operamos con B0 igual a -1
         NEG
         ADD 1
         SACL 2
         B  SD6
SD5:     LACL 0ah ; Operamos con B0 igual a 1
         ADD 1
         SACL 2

SD6:     LACL 7   ; Trabajamos con 1/A0
         SUB #1
         BCND SDD1,EQ ; Si 1/A0 es mayor de 1 vamos a SDD1, sino a SDD2
         B  SDD2
SDD1:    LTP 2  ; Multiplicamos por 1/A0
         MPY 8
         LTP 2
         SACL 2
```

```
SDD2:   LACL 9  ;Observamos el signo de 1/A0 y si es negativo vamos a SDD3
        SUB #1  ;sino a SDD4
        BCND SDD3,EQ
        B SDD4
SDD3:   LACL 2 ; Operamos cuando 1/A0 es negativo
        NEG
        SACL 2

SDD4:  ; INTERCAMBIAMOS LAS SEÑALES Si
        LACL 4
        SUB  #1
        SAMM AR1
        LAR AR0,5
        LAMM AR0
        ADD 4
        SUB #2
        SAMM AR0
        MAR *,AR0
        LACL *
PP:     LACL *+
        SACL *
        LAMM AR1
        SUB  #1
        BCND SPP,EQ
        SAMM AR1
        LAMM AR0
        SUB  #2
        SAMM AR0
        B  PP
SPP:     ; Fin de intercambio de las señales Si

        MAR  *,AR7;#### OPERAMOS LOS COEFICIENTES BN ####
        LAR AR7,6 ; AR7 puntero coeficientes BN
        LAR AR3,5 ; AR3 puntero señales Si
        LACL  #0
        SACL 1
        LACL 3
        SUB #1
        SACL 3
        LAMM AR7
        ADD #3
        SAMM AR7
        LAMM AR3
        ADD #1
        SAMM AR3

SFX:    LACL *+,AR3
        BCND SGF5,EQ ; Si BN es igual a cero vamos a SGF5
        SUB  #1
        BCND SGF6,EQ ; Si BN es negativo y menor que 1 vamos a SGF6
        SUB  #1
        BCND SGF7,EQ ; Si BN es positivo y menor que 1 vamos a SGF7
        SUB  #1
        BCND SGF8,EQ ; Si BN es -1 vamos a SGF8
        B  SGF9      ; Si BN es 1 vamos a SGF9
SGF5:   LAMM AR7 ; Operamos con BN igual a cero
        ADD  #2
        SAMM AR7
        LAMM AR3
        ADD  #1
        SAMM AR3
        B SGF10
SGF6:   LAR AR2,*+,AR7 ; Operamos con BN negativo y menor que 1
        LAR AR0,*+
        LAR AR1,*+,AR3
        CALL DIVISION
        NEG
        ADD  1
        SACL 1
        B  SGF10
SGF7:   LAR AR2,*+,AR7  ; Operamos con BN positivo y menor que 1
        LAR AR0,*+
        LAR AR1,*+,AR3
        CALL DIVISION
        ADD  1
        SACL 1
        B  SGF10
SGF8:   LACL *+   ; Operamos con BN igual a -1
```

```
        NEG
        ADD  1
        SACL 1
        LAMM AR7
        ADD  #2
        SAMM AR7
        B  SGF10
SGF9:   LACL  *+  ; Operamos con BN igual a !
        ADD  1
        SACL 1
        LAMM AR7
        ADD  #2
        SAMM AR7
SGF10:  MAR  *,AR7
        LACL 3
        SUB  #1        ; Comprobamos si hemos operado con todos los
        BCND SGF11,EQ  ; coeficientes y si es así salimos del bucle, sino
        SACL 3         ; pasamos al siguiente coeficiente BN
        B SFX
SGF11:  LACL 4
        SACL 3

        LACL 2 ; *** Dejamos la muestra procesada en el acumulador ***
        RET

;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%%%%%%%%%      RUTINA FILTRO IIR ESTRUCTURA DIRECTA TIPO II      %%%%%%%%%%%
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%% Este subprograma coge una muestra del acumulador y una vez procesada %%%
;%% devuelve el resultado nuevamente en el acumulador.                   %%%
;%% La manera de proceder es la siguiente:                               %%%
;%%  * direccionamos la página de memoria donde están los coeficientes   %%%
;%%     y las señales auxiliares del filtro en cuestión.                 %%%
;%%  * luego ponemos la muestra a procesar en el acumulador y ejecutamos %%%
;%%     el filtro con " CALL FILTRO_IIR"                                 %%%
;%%  * por último recogemos la muestra procesada en el acumulador        %%%
;%%  Las variables en memoria se ponen de la siguiente manera:           %%%
;%%          .ds     XXXXh  pagina de trabajo                            %%%
;%%          nom_filtro    .word    0                                    %%%
;%%          se_aux        .word    0,0,0                                %%%
;%%          ord_an        .word    X,X                                  %%%
;%%          ord_bn        .word    X,X                                  %%%
;%%          orden         .word    X                                    %%%
;%%          direc_coef    .word    X(S0),X(A0),X(B0)                    %%%
;%%          S0            .word    0                                    %%%
;%%          S1            .word    0                                    %%%
;%%          :               :      :                                    %%%
;%%          A0            .word    X,X,X                                %%%
;%%          A1            .word    X,X,X                                %%%
;%%          :               :      :                                    %%%
;%%          B0            .word    X,X,X                                %%%
;%%          B1            .word    X,X,X                                %%%
;%%          :               :      :                                    %%%
;%%  Y el programa se ejecuta como:                                      %%%
;%%                LDP  #nom_filtro                                      %%%
;%%                CALL FILTRO_IIR                                       %%%
;%%  (Informacioón más detallada en la documentación)                    %%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

FILTRO_IIR:
                  ;(*** Ver diagrama filtro IIR tipo II ***)
        ADD  1
        SACL 0ch

        MAR  *,AR7
        LAR  AR7,0ah ;  AR7 registro puntero del coeficiente A0
        LACL *+  ; cargamos A0
        SUB  #1
        BCND SG1,EQ  ; Si 1/A0 es mayor de 1 vamos a SG1
        LACL *+
        B SG2   ; Si 1/A0 es igual a 1 vamos a SG2

SG1:    LTP 0ch
        MPY *+   ; Multiplicamos por 1/A0 para obtener S0
        LTP 0ch
        SACL 0ch
SG2:    LACL *+
        SUB  #1    ; Si 1/A0 es negativo saltamos a SG3 sino, a SG4
```

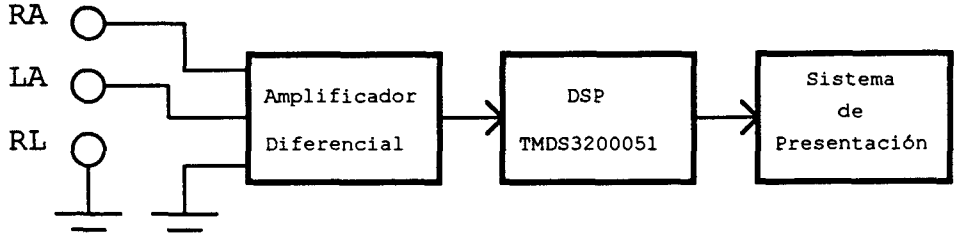
```
        BCND SG3,EQ
        B SG4
SG3:    LACL 0ch
        NEG    ; Como 1/A0 es negativo multiplicamos la muestra por -1
        SACL 0ch
SG4:
        MAR *,AR3
        LAR AR3,0bh  ; Registro AR3 puntero del coeficiente B0
        LACL  *+
        BCND SIGG1,EQ ; Si B0 es 0 saltamos a SIGG1
        SUB  #1
        BCND SIGG2,EQ ; Si B0 es negativo y menor que 1 vamos a SIGG2
        SUB  #1
        BCND SIGG3,EQ ; Si B0 es positivo y menor que 1 vamos a SIGG3
        SUB  #1
        BCND SIGG4,EQ ; Si B0 es -1 vamos a SIGG4
        B    SIGG5    ; Si B0 es 1 vamos a SIGG5

SIGG1:  LACL 2  ; operamos con B0 igual a 0
        SACL 3
        B     SIGG6
SIGG2:  LAR  AR2,0ch  ;operamos con B0 negativo y menor que 1
        LAR  AR0,*+
        LAR  AR1,*+
        CALL DIVISION
        NEG
        ADD  2
        SACL 3
        B     SIGG6
SIGG3:  LAR  AR2,0ch  ; operamos con B0 positivo y menor que 1
        LAR  AR0,*+
        LAR  AR1,*+
        CALL DIVISION
        ADD  2
        SACL 3
        B     SIGG6
SIGG4:  LACL 0ch   ; operamos con B0 igual a -1
        NEG
        ADD 2
        SACL 3
        B   SIGG6
SIGG5:  LACL 0ch   ; operamos con B0 igual a 1
        ADD 2
        SACL 3
        B   SIGG6

SIGG6:   LACL 8     ; intercambiamos las señales Si , es decir ,
         SUB  #1    ;          SN-1 ----> SN
         SAMM  AR1  ;          SN-2 ----> SN-1
         LAR  AR0,9h;           :          :
         LAMM  AR0  ;           S1  ----> S2
         ADD  8     ;           S0  ----> S1
         SUB  #2
         SAMM AR0
         MAR  *,AR0
         LACL *
XX:      LACL  *+
         SACL  *
         LAMM  AR1
         SUB  #1
         BCND  SXX,EQ
         SAMM  AR1
         LAMM  AR0
         SUB   #2
         SAMM  AR0
         B   XX
SXX:     MAR    *,AR7  ;   fin de intercambio señales Si

        LACL  4   ;#### OPERAMOS LOS COEFICIENTES AN ####
        SUB  #1
        SACL  4
        LAR   AR3,9h; El registro AR3 es el punteros de las señales Si
        LAMM  AR3
        ADD   #1
        SAMM  AR3   ; El registro AR7 es el puntero de los AN
        LACL  #0
        SACL  1
```

```
SGX:    LACL  *+,AR3
        BCND  SG5,EQ  ; Si el AN es cero vamos a SG5
        SUB  #1
        BCND  SG6,EQ  ; Si el AN es negativo y menor que 1 vamos a SG6
        SUB  #1
        BCND  SG7,EQ  ; Si el AN es positivo y menor que 1 vamos a SG7
        SUB  #1
        BCND  SG8,EQ  ; Si el AN es -1 vamos a SG8
        B   SG9       ; Si el AN es 1 vamos a SG9

SG5:    LAMM  AR7 ; operamos con AN igual a cero
        ADD   #2
        SAMM  AR7
        LAMM  AR3
        ADD   #1
        SAMM  AR3
        B   SG10
SG6:    LAR AR2,*+,AR7 ; operamos con AN negativo y menor que 1
        LAR AR0,*+
        LAR AR1,*+,AR3
        CALL DIVISION
        ADD  1
        SACL 1
        B    SG10
SG7:    LAR  AR2,*+,AR7 ;operamos con AN positivo y menor que 1
        LAR  AR0,*+
        LAR  AR1,*+,AR3
        CALL DIVISION
        NEG
        ADD 1
        SACL 1
        B    SG10
SG8:    LACL *+ ; operamos con AN igual a -1
        ADD  1
        SACL 1
        LAMM AR7
        ADD  #2
        SAMM AR7
        B   SG10
SG9:    LACL *+ ; operamos con AN igual a 1
        NEG
        ADD 1
        SACL 1
        LAMM  AR7
        ADD  #2
        SAMM  AR7
SG10:   MAR *,AR7
        LACL 4
        SUB  #1    ; comprobamos si hemos operado con todos los coeficientes
        SACL 4     ; y si es así salimos del bucle sino pasamos al siguiente
        BCND SG11,EQ ; coeficiente AN
        B  SGX
SG11:   LACL  5
        SACL  4

        MAR   *,AR7 ; #### OPERAMOS LOS COEFICIENTES BN ####
        LAR   AR7,0bh     ; registro AR7 puntero de los BN
        LAR   AR3,9h      ; registro AR3 puntero de las señales Si
        LACL  #0
        SACL  2
        LACL  6
        SUB  #1
        SACL  6
        LAMM  AR7
        ADD  #3
        SAMM  AR7
        LAMM  AR3
        ADD  #1
        SAMM  AR3

SGBX:   LACL  *+,AR3
        BCND  SGB5,EQ  ; Si BN es igual a cero vamos a SGB5
        SUB   #1
        BCND  SGB6,EQ  ; Si BN es negativo y menor que 1 vamos a SGB6
        SUB   #1
        BCND  SGB7,EQ  ; Si BN es positivo y menor que 1 vamos a SGB7
        SUB   #1
        BCND  SGB8,EQ  ; Si BN es -1 vamos a SGB8
```

```
        B     SGB9      ; Si BN es 1 vamos a SGB9
SGB5:   LAMM  AR7    ; operamos con BN igual a cero
        ADD   #2
        SAMM  AR7
        LAMM  AR3
        ADD   #1
        SAMM  AR3
        B   SGB10
SGB6:   LAR AR2,*+,AR7 ; operamos con BN negativo y menor que 1
        LAR AR0,*+
        LAR AR1,*+,AR3
        CALL DIVISION
        NEG
        ADD  2
        SACL 2
        B    SGB10
SGB7:   LAR  AR2,*+,AR7 ; operamos con BN positivo y menor que 1
        LAR  AR0,*+
        LAR  AR1,*+,AR3
        CALL DIVISION
        ADD 2
        SACL 2
        B    SGB10
SGB8:   LACL *+ ; operamos con BN igual a -1
        NEG
        ADD  2
        SACL 2
        LAMM AR7
        ADD  #2
        SAMM AR7
        B   SGB10
SGB9:   LACL *+    ; operamos con BN igual a 1
        ADD 2
        SACL 2
        LAMM  AR7
        ADD  #2
        SAMM  AR7
SGB10:  MAR *,AR7
        LACL 6
        SUB  #1          ; Comprobamos si hemos operado con todos los
        BCND SGB11,EQ    ; coeficientes y si es así salimos del bucle, sino
        SACL 6           ; pasamos al siguiente coeficiente BN
        B  SGBX
SGB11:  LACL  7
        SACL  6

        LACL  3;*** Dejamos la muestra procesada en el acumulador  ****
        RET


;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%%%%%%% RUTINA DE SERVICIO PARA LA INT. DE RECEPCION DEL PUERTO SERIE %%%%%%%
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
RECEPCION:
        LAMM    DRR  ; Muestra llegada del AIC al Acumulador


          ;*** Algoritmos para diseño de filtros en tiempo real***
          ; ( Ver tablas de coeficientes y de algoritmos de filtros)

        SAMM    DXR  ; Muestra del Acumulador al AIC
        RETE ; volvemos al bucle de espera hasta la próxima muestra

     .end
```

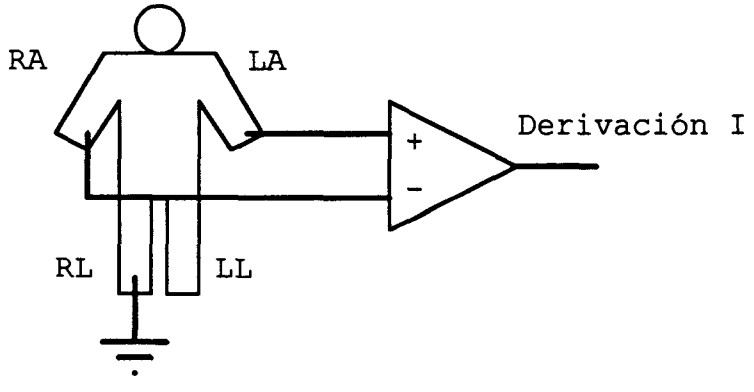
CAPITULO 5:SISTEMA DETECTOR DEL COMPLEJO QRS.

El sistema detector del complejo QRS va a estar basado en tres bloques fundamentales: Un amplificador diferencial , un DSP TMDS3200051 programado con los algoritmos de detección del complejo QRS y un sistema de presentación del ritmo cardiaco . El sistema global sigue el siguiente esquema de bloques:



5.1:AMPLIFICADOR DIFERENCIAL PARA LA SEÑAL DE ECG.

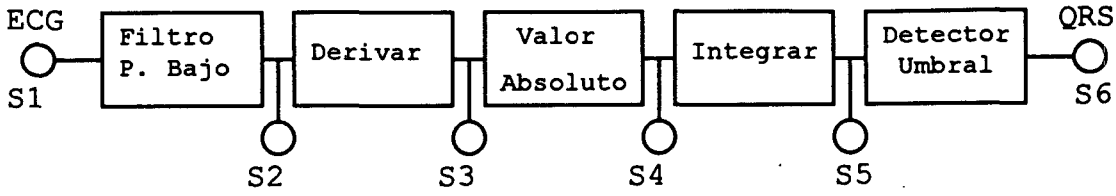
Como vimos anteriormente para obtener la señal de ECG teníamos que hacer una derivación , y mas concretamente vamos a utilizar la derivación I. Para ello necesitábamos un amplificador diferencial el cual tiene como masa la pierna derecha y como señales a diferenciar el brazo izquierdo y el brazo derecho según muestra la siguiente figura:



En el presente TFC se ha utilizado un generador de ECG el cual ya genera directamente la señal de ECG con derivación I que tiene que atacar al DSP TMDS3200051 donde se van a diseñar los algoritmos de detección del complejo QRS en tiempo real.

5.2:ALGORITMOS DE PROCESADO PARA DETECTAR EL COMPLEJO QRS.

Como vimos anteriormente los filtros digitales que teníamos que implementar para detectar el complejo QRS seguían el siguiente esquema:



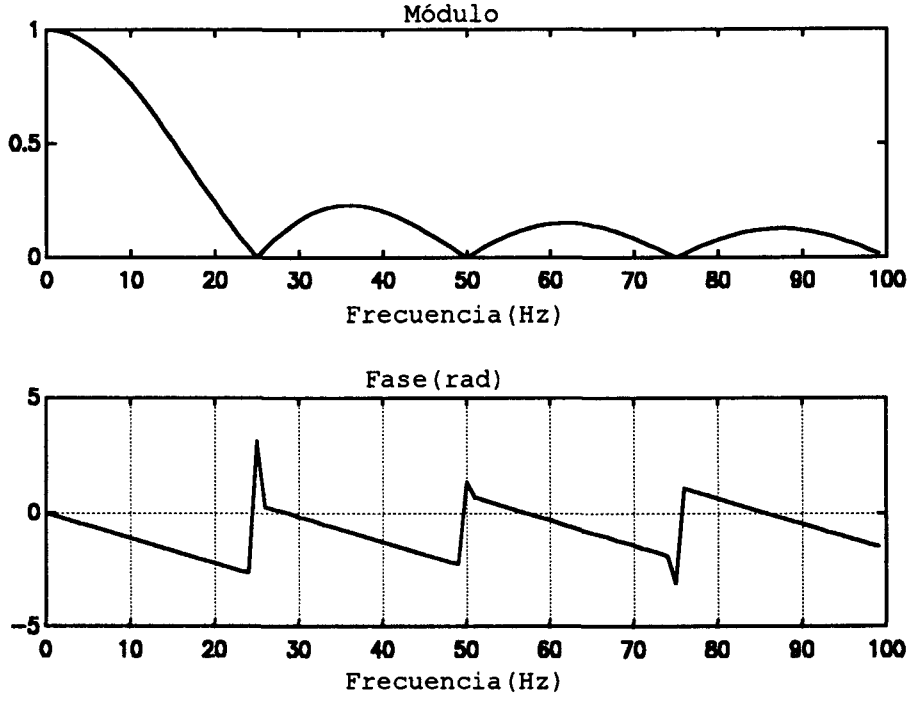
Dijimos anteriormente que íbamos a trabajar con una frecuencia de muestreo de 200 Hz y trabajando con el software simulador "DISFILT" desarrollado en el presente TFC obtuvimos los siguientes filtros:

FILTRO PASO BAJO:

El filtro paso bajo obtenido responde a la siguiente ecuación en diferencias:

$$y[n]=1/8(x[n]+x[n-1]+x[n-2]+x[n-3]+x[n-4]+x[n-5]+ +x[n-6]+x[n-7])$$

Esta ecuación en diferencias en el DSP se puede implementar con los coeficientes originales ya que están dentro de los márgenes de cuantificación . La respuesta en frecuencia (modulo y fase) analógica equivalente de este filtro es:



Como observamos este filtro tiene un cero en la frecuencia 50 Hz con lo cual habremos eliminado la interferencia de la red eléctrica y habremos dejado pasar

prácticamente intacta la señal de ECG ya que recordemos que se encontraba muy por debajo de los 20 Hz.

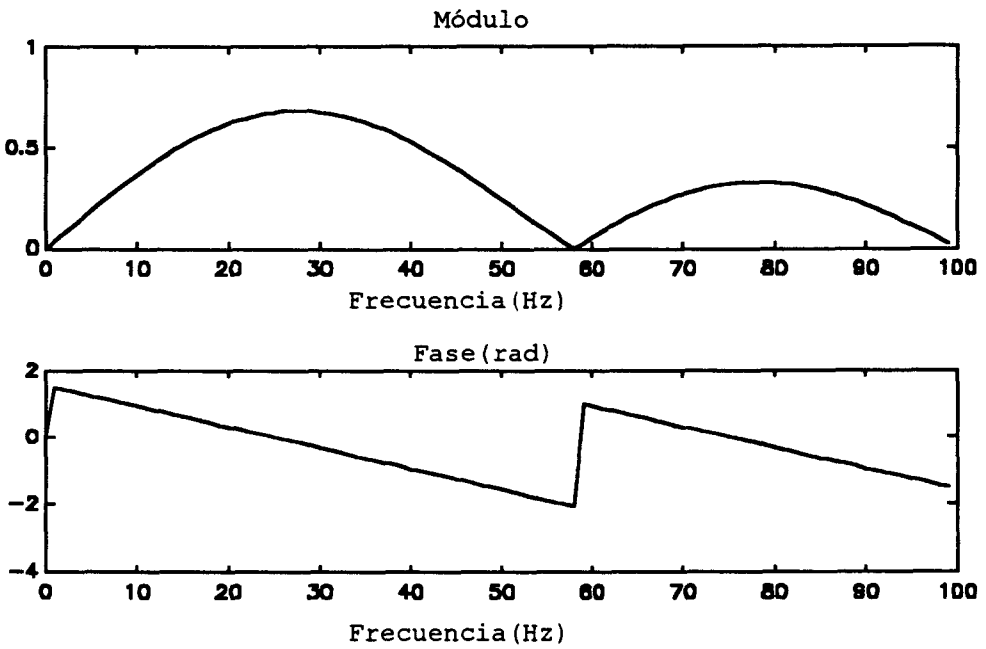
FILTRO DERIVADOR:

Este filtro lo que hará es acentuar el pico del complejo QRS ya que al diferenciar los valores más significativos aparecen donde la señal varía más rápidamente.

Obtenemos como resultado de hacer pruebas con el software citado anteriormente la siguiente ecuación en diferencias:

$$8y[n]=2x[n]+x[n-1]-x[n-3]-2x[n-4]$$

Este filtro tiene una respuesta analógica equivalente (módulo y fase) como la siguiente:



Como se observa en las gráfica anterior nos interesa la pendiente positiva del filtro en la gama de 0 a 20 Hz y será ahí donde se acentúe el la variación del complejo QRS.

VALOR ABSOLUTO:

Lo que se implementa ahora no es un sistema LTI y por tanto no tendrá una respuesta en frecuencia como los bloques anteriores y básicamente lo que hará es coger las muestras negativas y cambiarles el signo . Este sistema en el DSP responde a un algoritmo como el siguiente:

Leemos muestra de entrada
¿muestra positiva?
si: sacamos tal cual.
no: invertimos y sacamos.

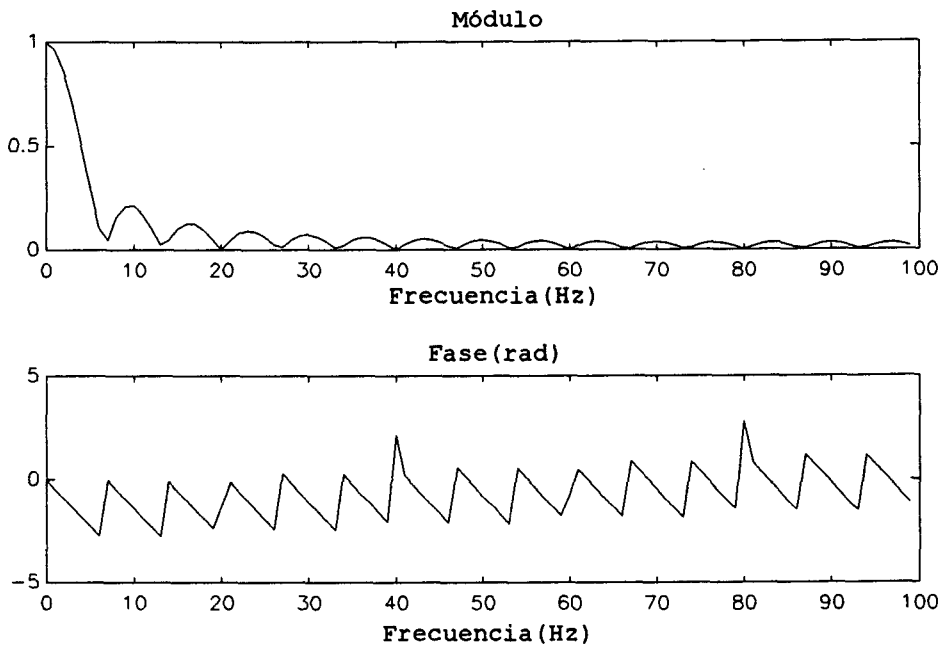
Mas adelante se verá con que instrucciones de ensamblador se programa este sistema.

FILTRO INTEGRADOR:

Ya tenemos las muestras con valores positivos en los lugares donde existía el complejo QRS y de valores despreciables en el resto de la señal . Para aproximar lo mas posible esta señal a un pulso cuadrado el siguiente paso será hacer un integrador el cual responde a la siguiente ecuación en diferencias :

$$30y[n]=x[n]+x[n-1]+x[n-2]+x[n-3]+....+x[n-29]$$

Este sistema nuevamente es un filtro FIR que tendrá una respuesta en frecuencia analógica equivalente como la siguiente:



DETECTOR DE UMBRAL:

Tras el integrador sólo nos falta poner un detector de umbral para hacer la señal de salida compatible con un sistema TTL o con cualquier otro que vaya a tratar dicha señal . La manera de programar esto es mediante un algoritmo como el siguiente:

Leemos la muestra de entrada
¿ Muestra mayor que V-umbral ?
si: sacamos un valor positivo.
no: sacamos valor cero.

El programa fuente para programar el DSP está en el fichero **CQRS.ASM** y tiene el siguiente listado:

```
;*************************************************************************
;*************************************************************************
;***   ESTRUCTURA GENERAL PARA IMPLEMENTAR FILTROS DIGITALES EN TIEMPO   ***
```

```
;***                    REAL CON EL DSP TMDS3200051                          ***
;*****************************************************************************
;*****************************************************************************

;*****************************************************************************
;******************** DECLARACION DE VARIABLES EN MEMORIA ********************
;*****************************************************************************
          .mmregs
          .ds     0f00h ; comienzo memoria datos (0f00h--2bc0h)
TA        .word  31
RA        .word  31
TB        .word  21         ;** Valores para frec. muestreo y filtros BP y LP **
RB        .word  21            ;( Ver tablas de diseño de filtros )
R_TCR     .word  2fh
R_PRD     .word  04h
AIC_CTR   .word  9h

          .ds    1000h   % Filtro paso bajo
fpb       .word  0
sea1      .word  0,0
ord1      .word  8,8
dco1      .word  100ah,1012h
a01       .word  0,1,0
s1        .word  0,0,0,0,0,0,0,0,0
b01       .word  4,0,0
b11       .word  4,0,0
b21       .word  4,0,0
b31       .word  4,0,0
b41       .word  4,0,0
b51       .word  4,0,0
b61       .word  4,0,0
b71       .word  4,0,0

          .ds    1100h   % Filtro derivador
fderi     .word  0
sea2      .word  0,0
ord2      .word  5,5
dco2      .word  110ah,110fh
a02       .word  0,1,0
s2        .word  0,0,0,0,0
b02       .word  2,8188,2
b12       .word  2,4092,3
b22       .word  0,0,0
b32       .word  1,4092,3
b42       .word  1,8188,2

          .ds  1200h    % Valor absoluto
absol     .word    0
xposi     .word    8000h

          .ds    1300h   % Filtro integrador
fint      .word  0
sea3      .word  0,0
ord3      .word  30,30
dco3      .word  130ah,1328h
a03       .word  0,1,0
s31       .word  0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
b03       .word  2,60,9
b13       .word  2,60,9
b23       .word  2,60,9
b33       .word  2,60,9
b43       .word  2,60,9
b53       .word  2,60,9
b63       .word  2,60,9
b73       .word  2,60,9
b83       .word  2,60,9
b93       .word  2,60,9
b103       .word  2,60,9
b113       .word  2,60,9
b123       .word  2,60,9
b133       .word  2,60,9
b143       .word  2,60,9
b153       .word  2,60,9
b163       .word  2,60,9
b173       .word  2,60,9
b183       .word  2,60,9
b193       .word  2,60,9
b203       .word  2,60,9
```

```
b213        .word  2,60,9
b223        .word  2,60,9
b233        .word  2,60,9
b243        .word  2,60,9
b253        .word  2,60,9
b263        .word  2,60,9
b273        .word  2,60,9
b283        .word  2,60,9
b293        .word  2,60,9

            .ds   1400h  ; Detector de umbral
xaun        .word   0
xpun        .word   8000h
xpoun       .word   1ff0h

;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%%%%%%%%%%%%%%%% DECLARACION DE LOS VECTORES DE INTERRUPCION %%%%%%%%%%%%%%%%%%
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
            .ps     080ah ;memoria vectores int. (0800h--083fh)
rint:     B       RECEPCION     ;0A Recepción del puerto serie RINT.


;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%%%%%%%%%%%%%%%%%%%% INICIALIZACION DEL TMS320C50 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
            .ps 0a00h  ;comienzo memoria de programa (0a00h--2bffh)
            .entry
INICIO:   SETC    INTM
          LDP     #0
          OPL     #0834h,PMST ; configuramos modo de trabajo del TMS320C50
          LACC    #0
          SAMM    CWSR
          SAMM    PDWSR
          SPLK    #022h,IMR
          CALL    AICINIT  ; Llamamos rutina de inicio AIC
          CLRC    OVM
          SPM     0
          SPLK    #012h,IMR
          CLRC    INTM

ESPERA:   NOP             ; Esperamos a que llegue una muestra del AIC
          NOP             ; para pasar a servir la rutina de recepción
          NOP
          B       ESPERA


;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%%%%%%%%%%%%       RUTINA DE INICIALIZACION DEL AIC          %%%%%%%%%%%%%%%%
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;*****************************************************************************
;***   RUTINA QUE INICIALIZA EL AIC TLC320C40 SEGUN LOS PARAMETROS     *******
;***               TA, RA,TB,RB,R_PRD,R_TCR y AIC_CTR                    ******
; ***************************************************************************
;***  %% Los valores de RA y RB fijan el filtro paso banda de entrada     ***
;***       y la frecuencia de muestreo del A/D                           ***
;***  %% Los valores de TA y TB fijan el filtro paso bajo de salida       ***
;***       y la frecuencia de muestreo del D/A                           ***
;***  %% Los valores de R_PRD y R_TCR fijan la frecuiencia del M_Clock    ***
;***  %% El valor de AIC_CTR fija el modo de trabajo del AIC              ***
;*****************************************************************************
;**  El reloj maestro vale  M_Clock=20736 KHz/[(TDDR+1)*(PRD+1)]          ***
;** PRD tiene 16bits y TDDR últimos 4 bits de TCR=2(TDDR)h                ***
;*****************************************************************************
;** El filtro paso banda tiene los siguientes parámetros:                 ***
;**  0 dB en la banda fci=.2 kHz*(SCF/288 KHz);fcs=3.5 KHz*(SCF/288 KHz)  ***
;*****************************************************************************
;** El filtro paso bajo tiene los siguientes parámetros:                  ***
;** 0 dB en la banda fci=0 Hz;fcs=3.5 KHz*(SCF/288 KHz)                   ***
;*****************************************************************************
;** El resto de parámetros para las frecuencias de muestreo y los filtros **
;**               SCF=M_Clock/(2*A)   ;  Fmuestreo=SCF/B                  ***
;*****************************************************************************
;** El AIC_CTR tiene 6 bits . AIC_CTR=G1 GO SY AX LB BP                   ***
;** G1 GO : ganacia del AIC      G1   GO ------  Ganancia                 ***
;**                              1    1          1/4                      ***
;**                              0    0          1/4                      ***
;**                              0    1          1/2                      ***
```

```
;**                                1    0          1                          ***
;** SY : recepción transmisión 0/1 asíncrona/síncrona                        ***
;** AX : entrada analógica auxiliar 0/1 deshabilitada/habilitada             ***
;** LB : función lazo entrada salida 0/1 deshabilita/habilita                ***
;** BP : filtro paso banda entrada 0/1 eliminamos/insertamos                 ***
;*****************************************************************************
AICINIT:
        LDP     #R_PRD      ;---------------------------
        LACL    R_PRD
        LDP     #0          ;  Generamos la frecuencia del M_Clock
        SACL    PRD         ;  en función de los valores de R_TCR
        LDP     #R_TCR      ;  y R_PRD
        LACL    R_TCR
        LDP     #0
        SACL    TCR
                            ;---------------------------
        MAR     *,AR0
        LACC    #0008h
        SACL    SPC
        LACC    #00c8h
        SACL    SPC
        LACC    #080h
        SACH    DXR
        SACL    GREG
        LAR     AR0,#0FFFFh
        RPT     #10000
        LACC    *,0,AR0
        SACH    GREG
        ;-------------------------
        LDP     #TA             ;
        SETC    SXM             ;
        LACC    TA,9            ; Inicializamos registro TA y RA
        ADD     RA,2            ;
        CALL    AIC_2ND         ;
        ;-------------------------
        LDP     #TB
        LACC    TB,9            ; Inicializamos registro TB y RB
        ADD     RB,2            ;
        ADD     #02h            ;
        CALL    AIC_2ND         ;
        ;-------------------------
        LDP     #AIC_CTR
        LACC    AIC_CTR,2       ; INICIALIZAMOS EL REGISTRO  DE CONTROL
        ADD     #03h            ; DEL AIC (Circuito Interfaz Analógico)
        CALL    AIC_2ND         ;
        RET                     ;

AIC_2ND:
        LDP     #0
        SACH    DXR
        CLRC    INTM
        IDLE
        ADD     #6h,15
        SACH    DXR
        IDLE
        SACL    DXR
        IDLE
        LACL    #0
        SACL    DXR
        IDLE
        SETC    INTM
        RET

;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%%%%%%      RUTINA DE DIVIVION CON DESPLAZAMIENTOS A LA DERECHA      %%%%%%%
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
; Al desplazar bits a la derecha  sólo dividirá entre 2,4,8,16,32, ...
; y el resutado es la parte entera. p.e. 9/8=1.125 ----> 1
;%%%%   Los parámetros de entrada se le pasan mediante los registros
;%%%%   auxiliares ARn y el resultado es puesto en estos registros de la
;%%%%   siguiente manera:
;%%%    * Número corrector en AR0
;%%%    * Número de desplazamientos en AR1
;%%%    * Número a dividir en AR2
;%%%    * resultado en ACC
;  La rutina utiliza los registros auxiliares siguientes para realizar
;  las operaciones: AR4,AR5 y AR6.
;%%% Los números correctores que hay que pasarle están en función del número
```

```
;%%%  de desplazamientos a realizar y son los siguientes:
; 1 des----3ffch; 2 des----1ffch; 3 des----0ffch; 4 des----07fch;
; 5 des----03fch; 6 des----01fch; 7 des----00fch; 8 des----007ch;
; 9 des----003ch; 10 des---001ch; 11 des----000ch; 12 des----0004h;

DIVISION:
        SST  #0,100 ; guardamos la página actual
        LDP  #0h ; página cero en el puntero de página
        LAMM AR2
        SAMM AR4  ; guardamos el valor a dividir en AR4
        AND #8000h
        SAMM AR6  ; guardamos el signo del valor en AR6
        BCND SIGUE,EQ  ; si es positivo vamos a SIGUE
        LAMM AR4
        AND #7fffh  ; si el valor en negativo estará en complemento a dos
        SUB #1      ; entonces lo tansformamos a binario natural
        CMPL
        SAMM AR5

ROTN:  LAMM AR5    ; OPERAMOS EL NUMERO SI ES NEGATIVO
       ROR         ; desplamos tantas veces a la derecha como indica AR1
       SAMM AR5
       LAMM AR1
       SUB #1
       SAMM AR1
       BCND SIGUE1,EQ
       B ROTN
SIGUE1: LAMM AR5
       AND AR0 ; correjimos con el número corrector y convertimos
       CMPL    ; nuevamente en complemento a dos
       ADD #1
       B SG ; nos vamos al final de la rutina

SIGUE: LAMM AR4 ;OPERAMOS EL NUMERO SI ES POSITIVO
       SAMM AR5

ROTP:  LAMM AR5
       ROR       ;rotamos tantas veces a la derecha como indica AR1
       SAMM AR5
       LAMM AR1
       SUB #1
       SAMM AR1
       BCND SIGUE2,EQ
       B ROTP

SIGUE2: LAMM AR5
       AND AR0  ; correjimos con el número corrector

SG:    LST #0,100 ;recuperamos la página  actual
            ; acabamos la rutina
       RET

;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%%%%%%%%%%          RUTINA FILTRO FIR ESTRUCTURA DIRECTA          %%%%%%%%%%
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%% Este subprograma coge una muestra del acumulador y una vez procesada %%%
;%% devuelve el resultado nuevamente en el acumulador.                    %%%
;%% La manera de proceder es la siguiente:                                %%%
;%%  * direccionamos la página de memoria donde están los coeficientes    %%%
;%%      y las señales auxiliares del filtro en cuestión.                  %%%
;%%  * luego ponemos la muestra a procesar en el acumulador y ejecutamos %%%
;%%      el filtro con " CALL FILTRO_FIR"                                  %%%
;%%  * por último recogemos la muestra procesada en el acumulador         %%%
;%%  Las variables en memoria se ponen de la siguiente manera:             %%%
;%%           .ds    XXXXh  pagina de trabajo                              %%%
;%%            nom_filtro   .word     0                                   %%%
;%%            se_aux       .word     0,0                                 %%%
;%%            orden        .word     X,X                                 %%%
;%%            direc_coef   .word     X(S0),X(B0)                         %%%
;%%            A0           .word     X,X,X                               %%%
;%%            S0           .word     0                                   %%%
;%%            S1           .word     0                                   %%%
;%%            :              :       :                                   %%%
;%%            B0           .word     X,X,X                               %%%
;%%            B1           .word     X,X,X                               %%%
;%%            :              :       :                                   %%%
;%%  Y el programa se ejecuta como:                                        %%%
;%%               LDP  #nom_filtro                                         %%%
```

```
;%%              CALL FILTRO_FIR                                          %%%
;%%  (Informacioón más detallada en la documentación)                     %%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

FILTRO_FIR:        ;(* Ver estructura filtro FIR directa *)

              SACL 0ah ; guardamos la muestra que llega en S0
              MAR *,AR7
              LAR  AR7,6 ; AR7 es el puntero del los BN
              LACL *+
              BCND  SD1,EQ ; Si B0 es cero vamos a SD1
              SUB  #1
              BCND  SD2,EQ ; Si B0 es negativo y menor que 1 vamos a SD2
              SUB  #1
              BCND  SD3,EQ ; Si B0 es positivo y menor que 1 vamos a SD3
              SUB  #1
              BCND  SD4,EQ ; Si B0 es -1 vamos a SD4
              B     SD5    ; Si B0 es 1 vamos a SD5
SD1:     LACL 1  ;Operamos con B0 igual a cero
         SACL 2
         B  SD6
SD2:     LAR AR2,0ah ; Operamos con B0 negativo y menor que 1
         LAR AR0,*+
         LAR AR1,*+
         CALL DIVISION
         NEG
         ADD 1
         SACL 2
         B  SD6
SD3:     LAR AR2,0ah  ; Operamos con B0 positivo y menor que 1
         LAR AR0,*+
         LAR AR1,*+
         CALL DIVISION
         ADD 1
         SACL 2
         B  SD6
SD4:     LACL 0ah ; Operamos con B0 igual a -1
         NEG
         ADD 1
         SACL 2
         B  SD6
SD5:     LACL 0ah ; Operamos con B0 igual a 1
         ADD 1
         SACL 2

SD6:     LACL 7   ; Trabajamos con 1/A0
         SUB #1
         BCND SDD1,EQ ; Si 1/A0 es mayor de 1 vamos a SDD1, sino a SDD2
         B  SDD2
SDD1:    LTP 2  ; Multiplicamos por 1/A0
         MPY 8
         LTP 2
         SACL 2
SDD2:    LACL 9  ;Observamos el signo de 1/A0 y si es negativo vamos a SDD3
         SUB #1  ;sino a SDD4
         BCND SDD3,EQ
         B SDD4
SDD3:    LACL 2 ; Operamos cuando 1/A0 es negativo
         NEG
         SACL 2

SDD4:  ; INTERCAMBIAMOS LAS SEÑALES Si
         LACL 4
         SUB  #1
         SAMM AR1
         LAR AR0,5
         LAMM AR0
         ADD 4
         SUB #2
         SAMM AR0
         MAR *,AR0
         LACL *
PP:      LACL *+
         SACL *
         LAMM AR1
         SUB  #1
         BCND SPP,EQ
         SAMM AR1
```

```
        LAMM AR0
        SUB  #2
        SAMM AR0
        B  PP
SPP:     ; Fin de intercambio de las señales Si

        MAR  *,AR7;#### OPERAMOS LOS COEFICIENTES BN ####
        LAR AR7,6 ; AR7 puntero coeficientes BN
        LAR AR3,5 ; AR3 puntero señales Si
        LACL  #0
        SACL 1
        LACL 3
        SUB #1
        SACL 3
        LAMM AR7
        ADD #3
        SAMM AR7
        LAMM AR3
        ADD #1
        SAMM AR3

SFX:    LACL *+,AR3
        BCND SGF5,EQ ; Si BN es igual a cero vamos a SGF5
        SUB  #1
        BCND SGF6,EQ ; Si BN es negativo y menor que 1 vamos a SGF6
        SUB  #1
        BCND SGF7,EQ ; Si BN es positivo y menor que 1 vamos a SGF7
        SUB  #1
        BCND SGF8,EQ ; Si BN es -1 vamos a SGF8
        B  SGF9      ; Si BN es 1 vamos a SGF9
SGF5:   LAMM AR7 ; Operamos con BN igual a cero
        ADD  #2
        SAMM AR7
        LAMM AR3
        ADD  #1
        SAMM AR3
        B SGF10
SGF6:   LAR AR2,*+,AR7 ; Operamos con BN negativo y menor que 1
        LAR AR0,*+
        LAR AR1,*+,AR3
        CALL DIVISION
        NEG
        ADD  1
        SACL 1
        B  SGF10
SGF7:   LAR AR2,*+,AR7  ; Operamos con BN positivo y menor que 1
        LAR AR0,*+
        LAR AR1,*+,AR3
        CALL DIVISION
        ADD  1
        SACL 1
        B  SGF10
SGF8:   LACL *+   ; Operamos con BN igual a -1
        NEG
        ADD  1
        SACL 1
        LAMM AR7
        ADD  #2
        SAMM AR7
        B  SGF10
SGF9:   LACL  *+  ; Operamos con BN igual a !
        ADD  1
        SACL 1
        LAMM AR7
        ADD  #2
        SAMM AR7
SGF10:  MAR  *,AR7
        LACL 3
        SUB  #1         ; Comprobamos si hemos operado con todos los
        BCND SGF11,EQ   ; coeficientes y si es así salimos del bucle, sino
        SACL 3          ; pasamos al siguiente coeficiente BN
        B SFX
SGF11:  LACL 4
        SACL 3

        LACL 2 ; *** Dejamos la muestra procesada en el acumulador ***
        RET
```

```
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%%%%%%%%%      RUTINA FILTRO IIR ESTRUCTURA DIRECTA TIPO II      %%%%%%%%%%%
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%% Este subprograma coge una muestra del acumulador y una vez procesada %%%
;%% devuelve el resultado nuevamente en el acumulador.                    %%%
;%% La manera de proceder es la siguiente:                                %%%
;%%  * direccionamos la página de memoria donde están los coeficientes    %%%
;%%     y las señales auxiliares del filtro en cuestión.                  %%%
;%%  * luego ponemos la muestra a procesar en el acumulador y ejecutamos %%%
;%%     el filtro con " CALL FILTRO_IIR"                                  %%%
;%%  * por último recogemos la muestra procesada en el acumulador         %%%
;%%  Las variables en memoria se ponen de la siguiente manera:            %%%
;%%          .ds     XXXXh  pagina de trabajo                             %%%
;%%          nom_filtro     .word     0                                   %%%
;%%          se_aux         .word     0,0,0                               %%%
;%%          ord_an         .word     X,X                                 %%%
;%%          ord_bn         .word     X,X                                 %%%
;%%          orden          .word     X                                   %%%
;%%          direc_coef     .word     X(S0),X(A0),X(B0)                   %%%
;%%          S0             .word     0                                   %%%
;%%          S1             .word     0                                   %%%
;%%          :                :       :                                   %%%
;%%          A0             .word     X,X,X                               %%%
;%%          A1             .word     X,X,X                               %%%
;%%          :                :       :                                   %%%
;%%          B0             .word     X,X,X                               %%%
;%%          B1             .word     X,X,X                               %%%
;%%          :                :       :                                   %%%
;%%  Y el programa se ejecuta como:                                       %%%
;%%                LDP  #nom_filtro                                       %%%
;%%                CALL FILTRO_IIR                                        %%%
;%%  (Informacioón más detallada en la documentación)                     %%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

FILTRO_IIR:
                   ;(*** Ver diagrama filtro IIR tipo II ***)
        ADD  1
        SACL 0ch

        MAR  *,AR7
        LAR  AR7,0ah ;  AR7 registro puntero del coeficiente A0
        LACL *+   ; cargamos A0
        SUB  #1
        BCND SG1,EQ  ; Si 1/A0 es mayor de 1 vamos a SG1
        LACL *+
        B SG2    ; Si 1/A0 es igual a 1 vamos a SG2

SG1:    LTP 0ch
        MPY *+    ; Multiplicamos por 1/A0 para obtener S0
        LTP 0ch
        SACL 0ch
SG2:    LACL *+
        SUB  #1     ; Si 1/A0 es negativo saltamos a SG3 sino, a SG4
        BCND SG3,EQ
        B SG4
SG3:    LACL 0ch
        NEG     ; Como 1/A0 es negativo multiplicamos la muestra por -1
        SACL 0ch
SG4:
        MAR *,AR3
        LAR AR3,0bh  ; Registro AR3 puntero del coeficiente B0
        LACL  *+
        BCND SIGG1,EQ ; Si B0 es 0 saltamos a SIGG1
        SUB  #1
        BCND SIGG2,EQ ; Si B0 es negativo y menor que 1 vamos a SIGG2
        SUB  #1
        BCND SIGG3,EQ ; Si B0 es positivo y menor que 1 vamos a SIGG3
        SUB  #1
        BCND SIGG4,EQ ; Si B0 es -1 vamos a SIGG4
        B    SIGG5    ; Si B0 es 1 vamos a SIGG5

SIGG1:  LACL 2  ; operamos con B0 igual a 0
        SACL 3
        B     SIGG6
SIGG2:  LAR  AR2,0ch  ;operamos con B0 negativo y menor que 1
        LAR  AR0,*+
        LAR  AR1,*+
        CALL DIVISION
```

```
        NEG
        ADD  2
        SACL 3
        B     SIGG6
SIGG3:  LAR  AR2,0ch  ; operamos con B0 positivo y menor que 1
        LAR  AR0,*+
        LAR  AR1,*+
        CALL DIVISION
        ADD  2
        SACL 3
        B     SIGG6
SIGG4:  LACL 0ch   ; operamos con B0 igual a -1
        NEG
        ADD 2
        SACL 3
        B   SIGG6
SIGG5:  LACL 0ch   ; operamos con B0 igual a 1
        ADD 2
        SACL 3
        B   SIGG6

SIGG6:   LACL 8     ; intercambiamos las señales Si , es decir ,
         SUB  #1    ;          SN-1 ----> SN
         SAMM  AR1  ;          SN-2 ----> SN-1
         LAR  AR0,9h;           :          :
         LAMM  AR0  ;           S1  ----> S2
         ADD  8     ;           S0  ----> S1
         SUB  #2
         SAMM AR0
         MAR  *,AR0
         LACL *
XX:      LACL  *+
         SACL  *
         LAMM  AR1
         SUB  #1
         BCND  SXX,EQ
         SAMM  AR1
         LAMM  AR0
         SUB   #2
         SAMM  AR0
         B   XX
SXX:     MAR    *,AR7  ;   fin de intercambio señales Si

        LACL  4   ;#### OPERAMOS LOS COEFICIENTES AN ####
        SUB  #1
        SACL  4
        LAR   AR3,9h; El registro AR3 es el punteros de las señales Si
        LAMM  AR3
        ADD   #1
        SAMM  AR3   ; El registro AR7 es el puntero de los AN
        LACL  #0
        SACL  1

SGX:    LACL  *+,AR3
        BCND  SG5,EQ  ; Si el AN es cero vamos a SG5
        SUB  #1
        BCND  SG6,EQ  ; Si el AN es negativo y menor que 1 vamos a SG6
        SUB  #1
        BCND  SG7,EQ  ; Si el AN es positivo y menor que 1 vamos a SG7
        SUB  #1
        BCND  SG8,EQ  ; Si el AN es -1 vamos a SG8
        B   SG9       ; Si el AN es 1 vamos a SG9

SG5:    LAMM  AR7 ; operamos con AN igual a cero
        ADD   #2
        SAMM  AR7
        LAMM  AR3
        ADD   #1
        SAMM  AR3
        B   SG10
SG6:    LAR AR2,*+,AR7 ; operamos con AN negativo y menor que 1
        LAR AR0,*+
        LAR AR1,*+,AR3
        CALL DIVISION
        ADD  1
        SACL 1
        B    SG10
SG7:    LAR  AR2,*+,AR7 ;operamos con AN positivo y menor que 1
```

```
        LAR  AR0,*+
        LAR  AR1,*+,AR3
        CALL DIVISION
        NEG
        ADD 1
        SACL 1
        B    SG10
SG8:    LACL *+ ; operamos con AN igual a -1
        ADD  1
        SACL 1
        LAMM AR7
        ADD  #2
        SAMM AR7
        B   SG10
SG9:    LACL *+ ; operamos con AN igual a 1
        NEG
        ADD 1
        SACL 1
        LAMM  AR7
        ADD  #2
        SAMM  AR7
SG10:   MAR *,AR7
        LACL 4
        SUB  #1    ; comprobamos si hemos operado con todos los coeficientes
        SACL 4     ; y si es asi salimos del bucle sino pasamos al siguiente
        BCND SG11,EQ ; coeficiente AN
        B  SGX
SG11:   LACL  5
        SACL  4

        MAR   *,AR7 ; #### OPERAMOS LOS COEFICIENTES BN ####
        LAR   AR7,0bh     ; registro AR7 puntero de los BN
        LAR   AR3,9h      ; registro AR3 puntero de las señales Si
        LACL  #0
        SACL  2
        LACL  6
        SUB  #1
        SACL  6
        LAMM  AR7
        ADD  #3
        SAMM  AR7
        LAMM  AR3
        ADD  #1
        SAMM  AR3

SGBX:   LACL  *+,AR3
        BCND  SGB5,EQ  ; Si BN es igual a cero vamos a SGB5
        SUB   #1
        BCND  SGB6,EQ  ; Si BN es negativo y menor que 1 vamos a SGB6
        SUB   #1
        BCND  SGB7,EQ  ; Si BN es positivo y menor que 1 vamos a SGB7
        SUB   #1
        BCND  SGB8,EQ  ; Si BN es -1 vamos a SGB8
        B     SGB9     ; Si BN es 1 vamos a SGB9
SGB5:   LAMM  AR7   ; operamos con BN igual a cero
        ADD   #2
        SAMM  AR7
        LAMM  AR3
        ADD   #1
        SAMM  AR3
        B   SGB10
SGB6:   LAR AR2,*+,AR7 ; operamos con BN negativo y menor que 1
        LAR AR0,*+
        LAR AR1,*+,AR3
        CALL DIVISION
        NEG
        ADD  2
        SACL 2
        B    SGB10
SGB7:   LAR  AR2,*+,AR7 ; operamos con BN positivo y menor que 1
        LAR  AR0,*+
        LAR  AR1,*+,AR3
        CALL DIVISION
        ADD 2
        SACL 2
        B    SGB10
SGB8:   LACL *+ ; operamos con BN igual a -1
        NEG
```

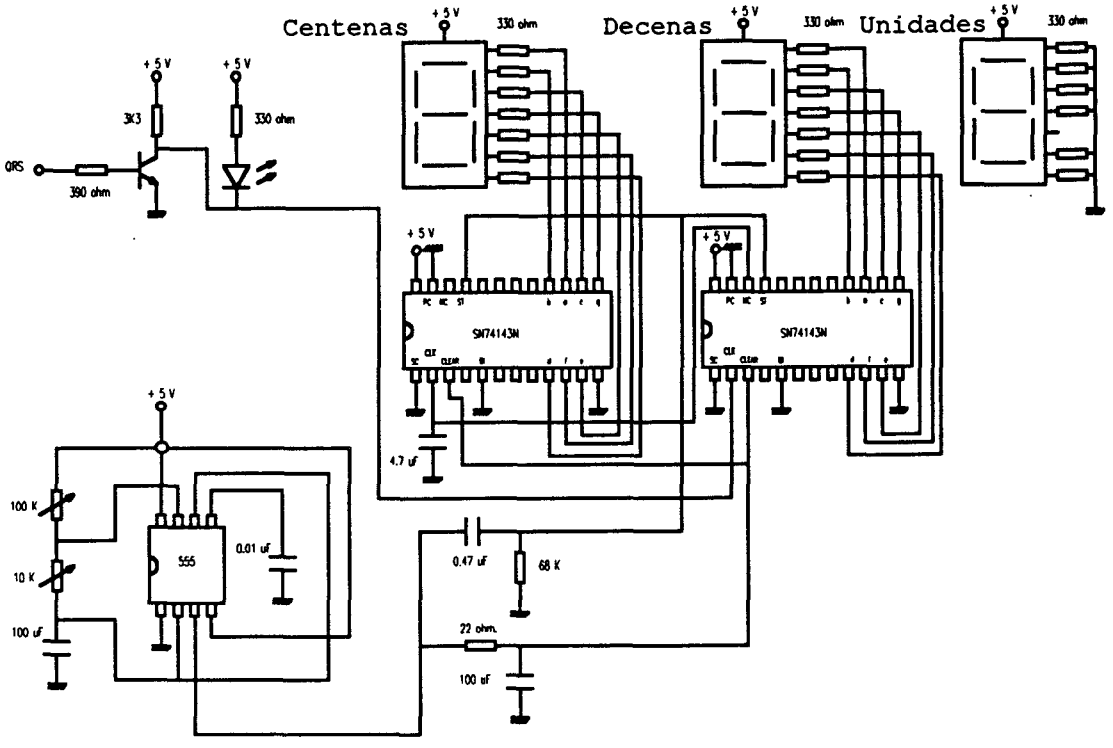
```
        ADD  2
        SACL 2
        LAMM AR7
        ADD  #2
        SAMM AR7
        B   SGB10
SGB9:   LACL *+   ; operamos con BN igual a 1
        ADD 2
        SACL 2
        LAMM  AR7
        ADD  #2
        SAMM  AR7
SGB10:  MAR *,AR7
        LACL 6
        SUB  #1          ; Comprobamos si hemos operado con todos los
        BCND SGB11,EQ    ; coeficientes y si es así salimos del bucle, sino
        SACL 6           ; pasamos al siguiente coeficiente BN
        B  SGBX
SGB11:  LACL  7
        SACL  6

        LACL  3;*** Dejamos la muestra procesada en el acumulador  ****
        RET


;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;%%%%%% RUTINA DE SERVICIO PARA LA INT. DE RECEPCION DEL PUERTO SERIE %%%%%%
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
RECEPCION:

        LAMM    DRR  ; Muestra llegada del AIC al Acumulador
        LDP #fpb
        CALL FILTRO_FIR
        LDP #fderi
        CALL FILTRO_FIR

        LDP     #absol
        SACL    absol
        AND     xposi
        BCND    SEGXY,EQ
        LACL    absol
        NEG
        B       SXY
SEGXY:  LACL    absol

SXY:
        LDP  #fint
        CALL FILTRO_FIR
        NEG

        LDP  #xaun
        SACL  xaun
        AND   xpun
        BCND  SEGUN,EQ
        LACL  #0
        B     SGUN
SEGUN:  LACL  xaun
        OR    xpoun
SGUN:

        SAMM    DXR  ; Muestra del Acumulador al AIC
        RETE ; volvemos al bucle de espera hasta la próxima muestra

        .end
```

Con esta cascada de sistemas dentro del DSP TMDS3200051 ya hemos obtenido el complejo QRS a partir de la señal de ECG . Ahora podemos , por ejemplo , calcular el ritmo cardiaco con un contador TTL a la salida del DSP. Este contador se describe en el siguiente apartado.

5.3:SISTEMA PARA CONTAR PULSACIONES CARDIACAS.

Este sistema es un simple contador de pulsos TTL el cual nos va a dar un lectura del ritmo cardiaco en pulsaciones por minuto.

El esquema electrónico del sistema utilizado para este cometido es el siguiente:



Como se observa en el esquema anterior la señal QRS ataca un transistor que conforma la señal para luego atacar a los contadores . La señal QRS conformada ataca a un contador decimal SN74143N que contará las unidades , cuando este llegue a nueve mandará una señal por la patilla 22 (MC cuenta máxima) que atacará al siguiente contador (contador de centenas) y así podremos contar desde 0 hasta 990 . El display de las unidades siempre estará marcando cero por las razones que se detalla a continuación:

El sistema de conteo está basado en contar la señal durante 6 segundos y luego este valor lo multiplicamos por 10 y obtendremos las pulsaciones en un minuto . por lo tanto contamos cada 6 segundos y el display de la derecha al ponerlo a cero es similar a multiplicar por diez.

El reloj encargado de contar la señal cada 6 segundos es un simple "555" el cual genera una señal de pulsos de duración 6 segundos que va a atacar los clear y los strobes de los contadores tras unos retardos previos con células RC . Nos interesa primero atacar los strobes (ST patilla 21 de los SN74143N que pasa el valor del contador al display y lo mantiene hasta una próxima excitación) por esa razón utilizamos una célula RC como derivador y mas tarde atacaremos los clear (CLEAR patilla 3 de los SN74143N que

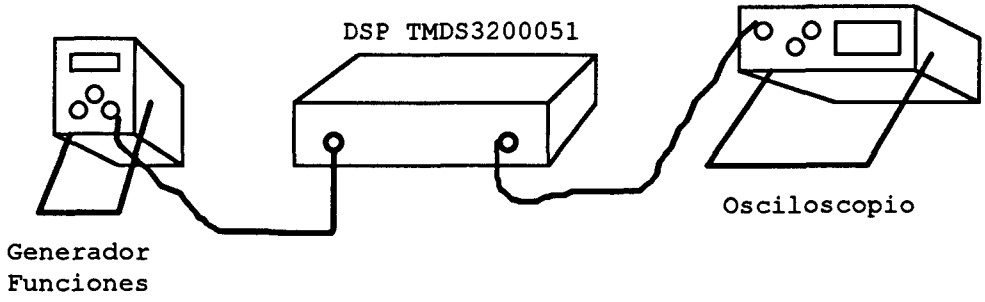
pone los contadores a cero) con una célula RC como integrador.

También el sistema cuenta con una indicación luminosa cada vez que llega un pulso QRS mediante un diodo LED.

CAPITULO 6: RESULTADOS.

Para contrastar los resultados teóricos con los prácticos se han programado una serie de filtros en el DSP y mediante un osciloscopio y ayudados de un generador de funciones se ha medido la respuesta en frecuencia (módulo lineal) para compararla con la gráficas que arroja el programa DISFILT.

El esquema utilizado para realizar las medidas ha sido el siguiente:



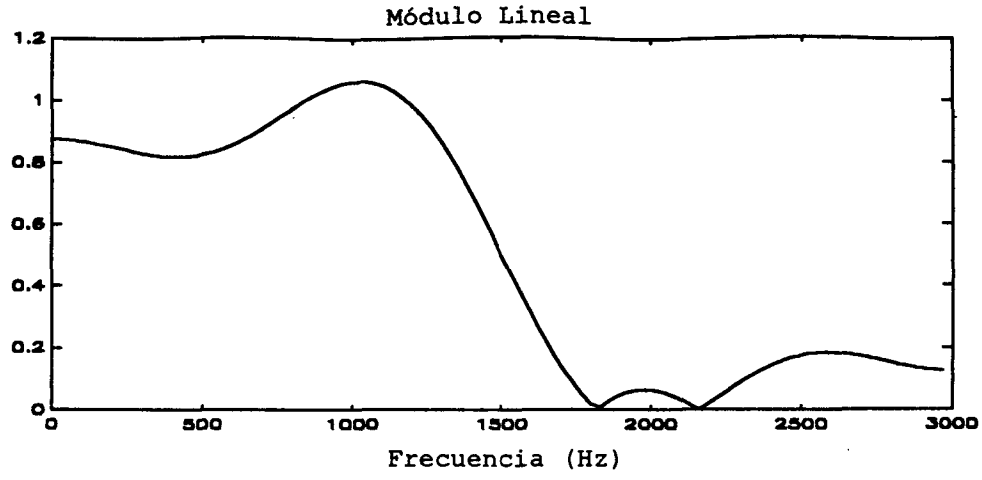
Las gráficas que genera el programa DISFILT se representan en línea continua y las que se obtuvieron en el laboratorio son puntos que luego se interpolaron con tramos rectos entre punto y punto.

Los filtros que se han utilizado para realizar las medidas son:

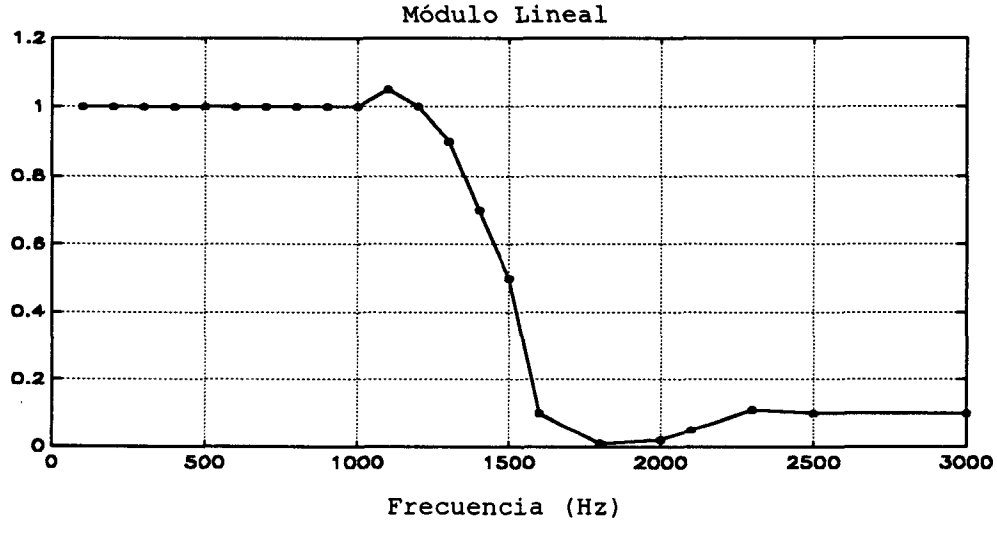
FILTRO N°1:

 # Filtro FIR ventana Rectangular paso bajo.
 # Orden 10.
 # Frecuencia de corte digital $\pi/2$.
 # Frecuencia muestreo 6 KHz.

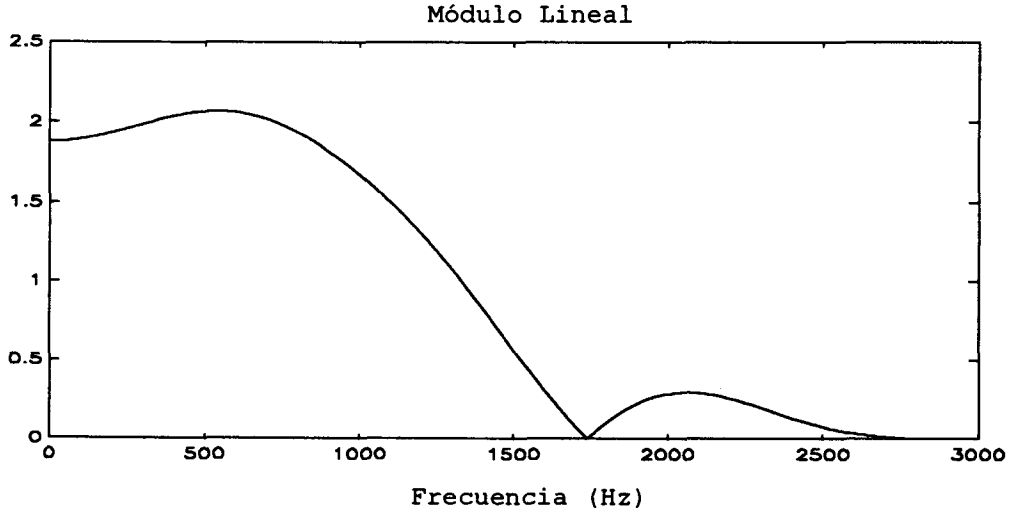
Gráfica de DISFILT para realización directa:



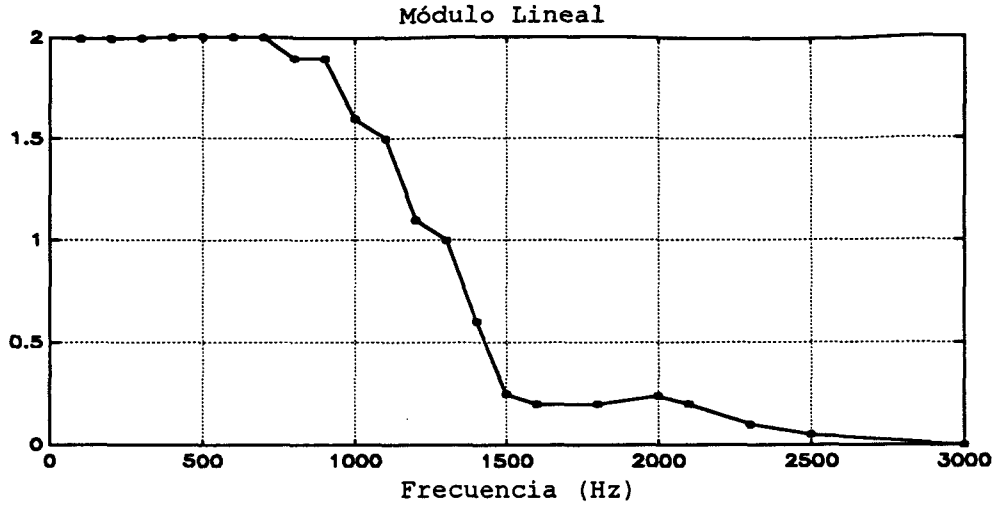
Gráfica obtenida en el laboratorio:



Gráfica de DISFILT para la realización en cascada:



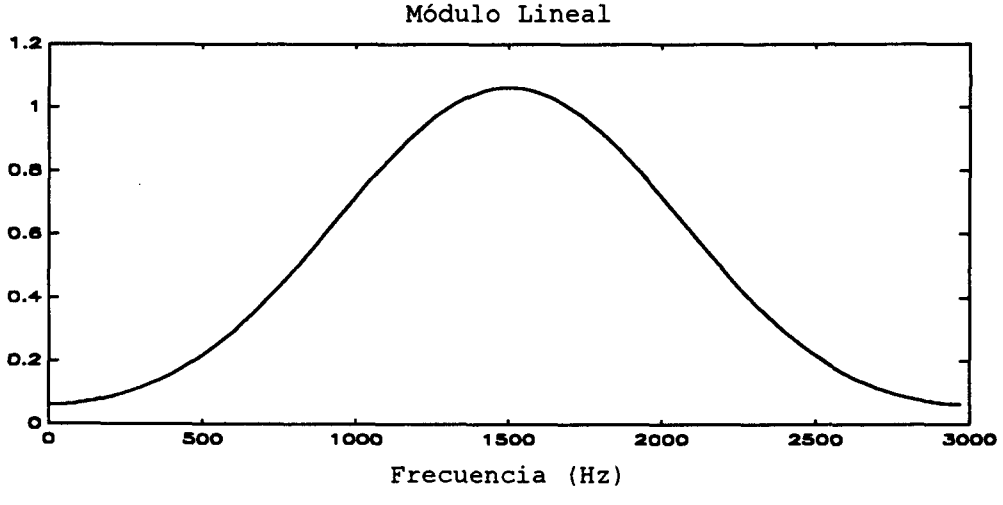
Gráfica obtenida en el laboratorio:



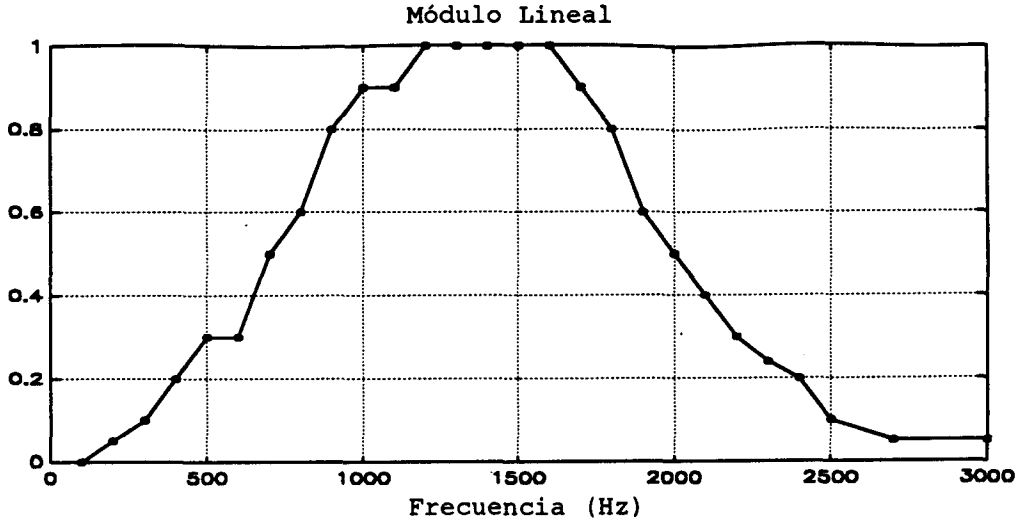
FILTRO Nº2:

 # Filtro FIR ventana Hamming paso banda.
 # Orden 8.
 # Frecuencia de corte digital inferior 0.4*π.
 # Frecuencia de corte superior digital 0.6*π.
 # Frecuencia muestreo 6 KHz.

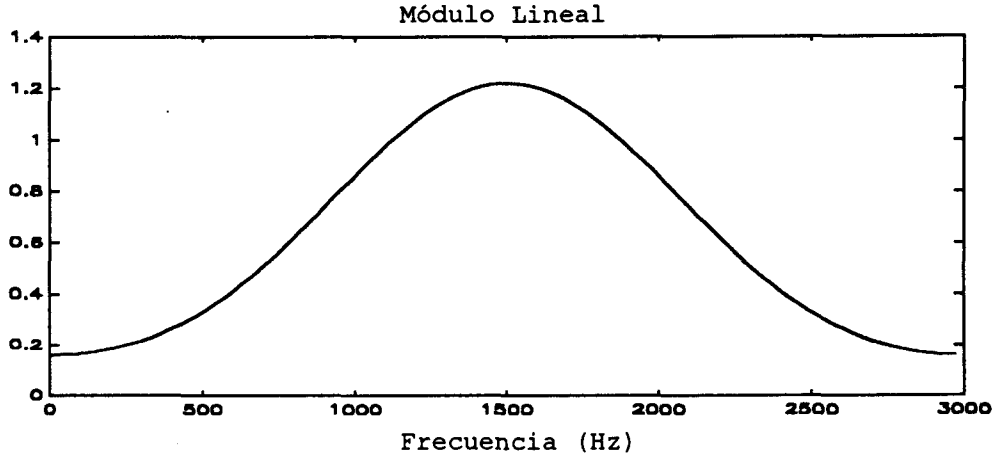
Gráfica de DISFILT para la realización directa:



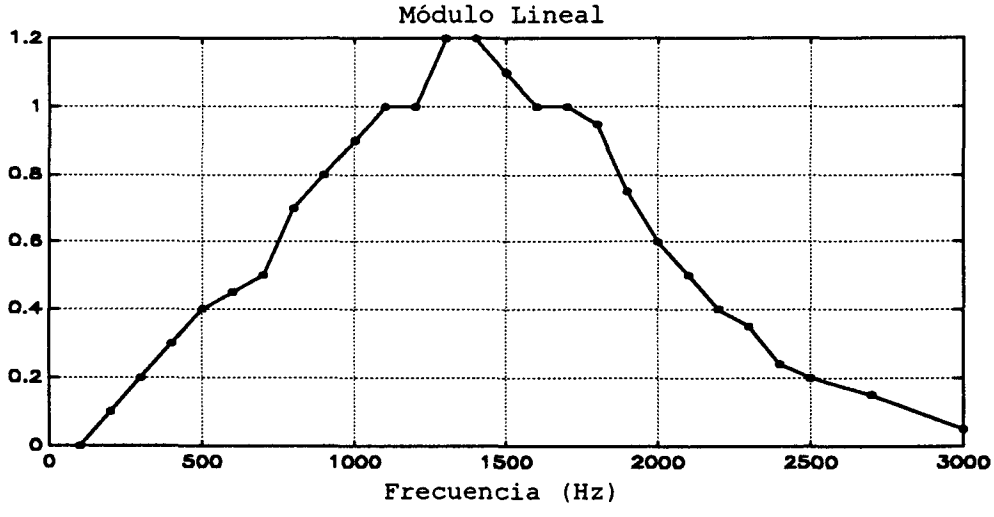
Gráfica obtenida en el laboratorio:



Gráfica de DISFILT para la realización en cascada:



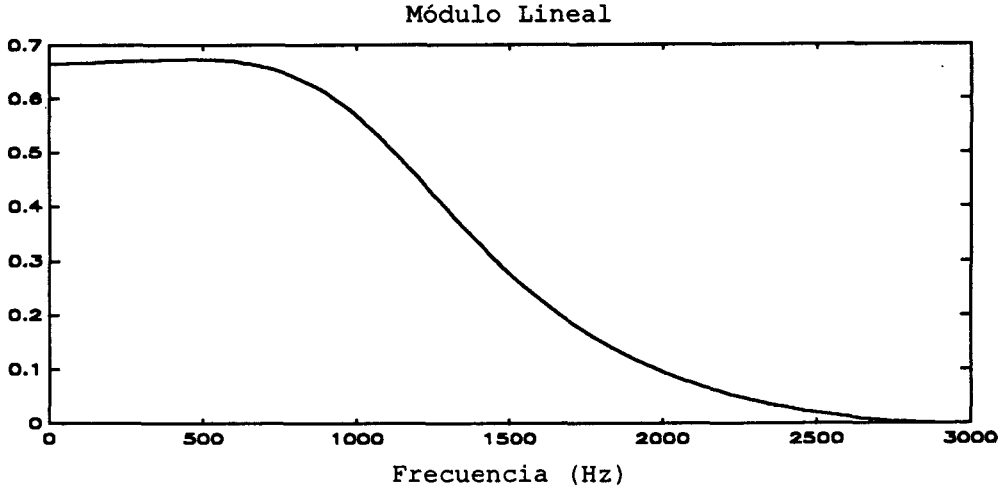
Gráfica obtenida en el laboratorio:



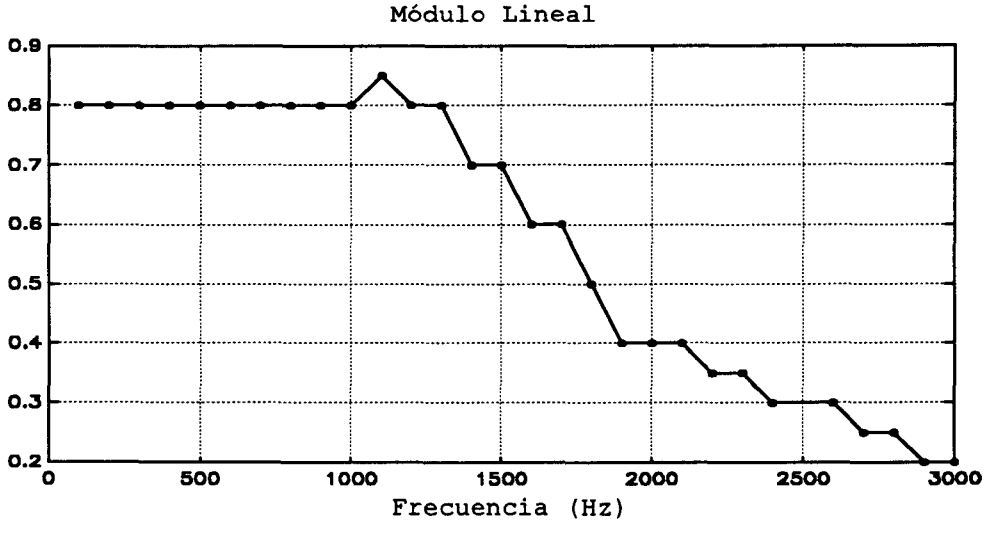
FILTRO N°3:

 # Filtro IIR Butterworth paso bajo.
 # Frecuencia de paso digital $\pi/4$.
 # Frecuencia de atenuación $\pi/2$.
 # Atenuación máxima en la banda de paso 1 dB.
 # Atenuación mínima en la banda atenuada 9 dB.
 # Frecuencia muestreo 6 KHz.

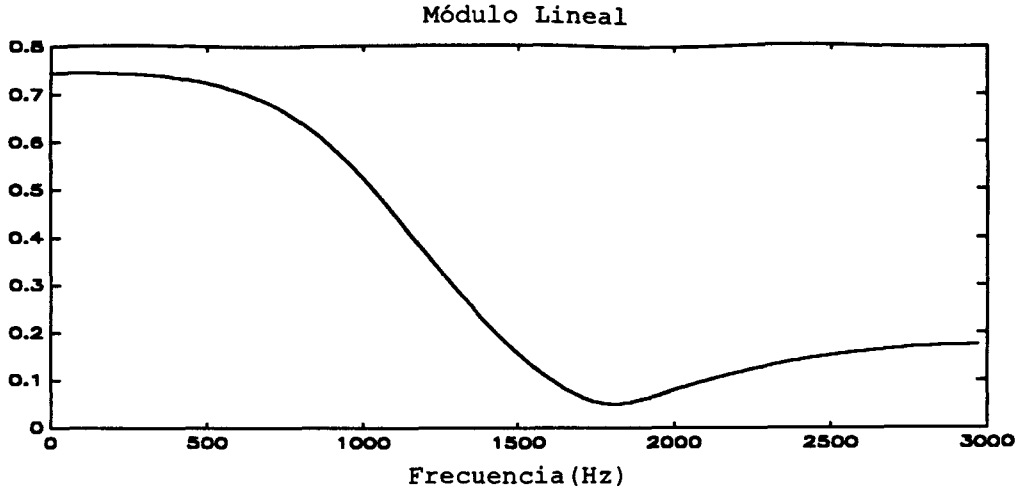
Gráfica de DISFILT para la realización directa:



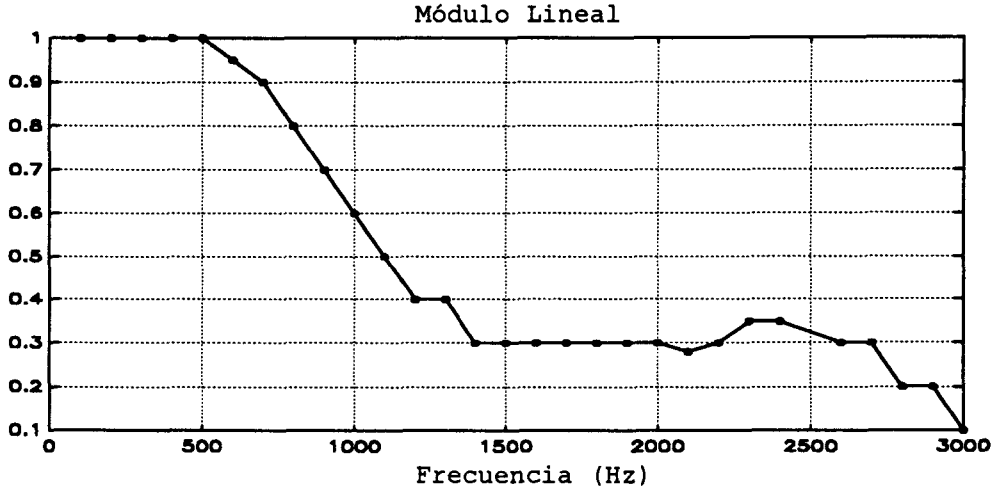
Gráfica obtenida en el laboratorio:



Gráfica de DISFILT para la realización en paralelo:



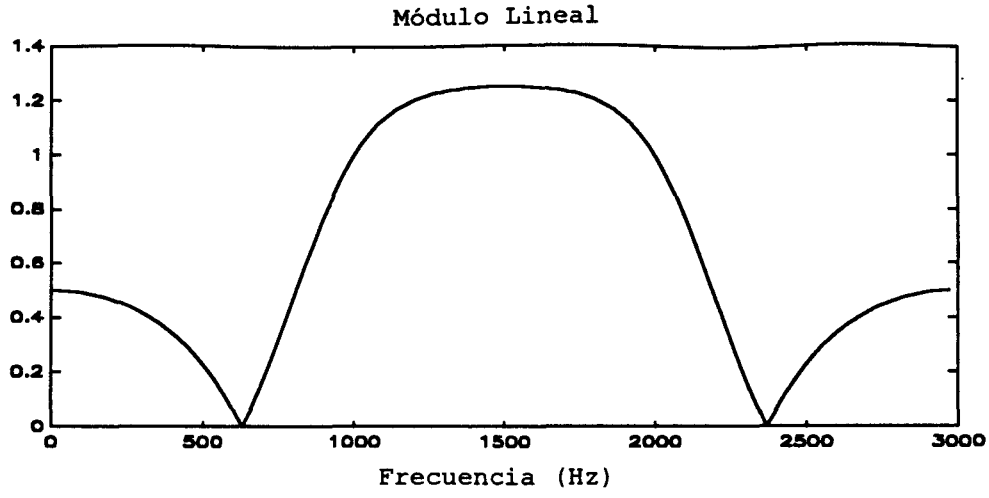
Gráfica obtenida en el laboratorio:



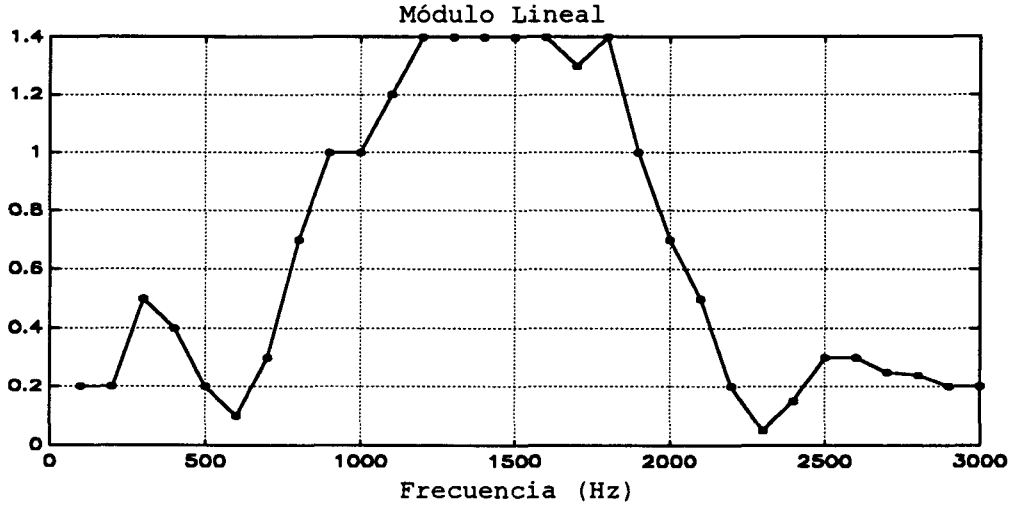
FILTRO N°4:

 # Filtro IIR Chebyshev Inverso paso banda.
 # Frecuencia de paso inferior digital 0.4*π.
 # Frecuencia de paso superior digital 0.6*π.
 # Frecuencia de atenuación inferior 0.2*π.
 # Frecuencia de atenuación superior 0.8π.
 # Atenuación máxima en la banda de paso 1 dB.
 # Atenuación mínima en la banda atenuada 10 dB.
 # Frecuencia muestreo 6 KHz.

Gráfica de DISFILT para la realización directa:



Gráfica obtenida en el laboratorio:



CAPITULO 7: TRABAJOS FUTUROS.

El presente TFC queda abierto para posibles ampliaciones o mejoras sobre la siguiente línea de trabajo:

Mejorar el programa DISFILT de forma que permita sintetizar señales.

Automatizar el proceso de generación de ficheros fuente en ensamblador.

Independizar el DSP del sistema de desarrollo y hacerlo autónomo mediante baterías.

Utilizar el complejo QRS en la detección de patologías cardiacas específicas.

Como aplicación inmediata se podría plantear un detector de arritmias portátil

CAPITULO 8: BIBLIOGRAFIA.

Los textos consultados para la realización del presente Trabajo fin de carrera son:

[TEXT-1]:DISCRETE-TIME SIGNAL PROCESSING.
Oppenheim,A.V.,Shafer,R.W.
[TEXT-2]:SIGNALS AND SYSTEMS.
Oppenheim,A.V.,Willsky,A.S.
[TEXT-3]:BIOMEDICAL DIGITAL SIGNAL PROCESSING.
Willis J. Tompkins.
[TEXT-4]:PC-MATLAB MANUAL.
The mathWorks,Inc.
[TEXT-5]:SIGNAL PROCESSING TOOLBOX.
John N. Little and Loren Shure.
[TEXT-6]:TMS320C5x DSP STARTER KIT.USER´S GUIDE.
Texas Instruments.
[TEXT-7]:TMS320C5x USER´S GUIDE.
Texas Instruments.

CAPITULO 9:ANEXO MANUAL DE USUARIO <<DISFILT>>.

El programa DISFILT es un programa para el diseño y análisis de filtros digitales tanto para sistemas de alta precisión (DSP's con punto flotante) como para sistemas de punto fijo como el DSP TMDS3200051 . El programa está elaborado en **MATLAB para WINDOWS** y consta de las siguientes aplicaciones:

1.- DISEÑO DE FILTROS DIGITALES.
2.- ANÁLISIS DE FILTROS DIGITALES.
3.- ESTUDIAR FILTRO ANALÓGICO GLOBAL.
4.- SIMULAR CON SEÑALES DIGITALIZADAS.
5.- SIMULAR LOS FILTROS EN EL DSP TMDS3200051.
6.- FILTRO ANALÓGICO GLOBAL DEL DSP TMDS3200051.
7.- SIMULAR CON SEÑALES DIGITALIZADAS EN EL DSP TMDS3200051.
8.- PROGRAMACIÓN DEL DSP TMDS3200051.
9.- AYUDA PARA UTILIZAR EL PROGRAMA.

El programa se ejecuta simplemente escribiendo el comando: **disfilt** y ya el resto se actúa con el ratón salvo algún dato numérico que hay que introducirle desde el teclado.

9.1:DISEÑO DE FILTROS DIGITALES.

Dentro de esta aplicación se pueden diseñar filtros digitales tanto FIR (método de las ventanas)como IIR (método de la transformación bilineal) con especificaciones de módulo.

Al entrar en la aplicación nos pide qué filtro queremos diseñar:

 # Filtro FIR.
 # Filtro IIR.

Diseño de filtro FIR:

Lo primero que nos pide es con qué tipo de ventana queremos diseñar el filtro y tenemos las siguientes posibilidades:

Ventana Rectangular.
Ventana Hamming.
Ventana Hanning.
Ventana Bartlett.
Ventana Blackman.

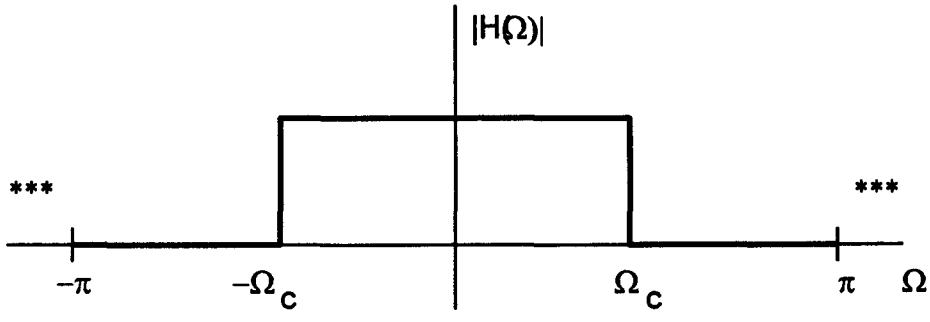
Una vez elegida la ventana nos preguntará el tipo de filtro:

Paso bajo.
Paso alto.
Paso banda.
Rechazo banda.

Seguidamente tenemos que darle el orden del filtro que debe ser un numero entero mayor que 1 . Recordemos que a medida que el orden es mayor , más se ajustará el filtro a la característica ideal.

Cuando le hayamos dado el orden nos pedirá el tipo de especificacíon en frecuencia teniendo la posibilidad de dársela en analógica o digital . Si se la damos en analógica tenemos que indicarle la frecuencia de muestreo y posteriormente las frecuencias de corte del filtro en Hz que deben ir desde cero hasta la mitad de la frecuencia de muestreo . Por el contrario si la especificación en frecuencia se la damos en digital la frecuencias deben estar entre cero y π.

A continuación se muestra una carátula de especificación paso bajo el resto de tipos (paso alto, paso banda y rechazo banda) se extratapolan fácilmente de esta.



Una vez introducidos los parámetros del filtro este lo guarda en el disco para posteriores aplicaciones.

Diseño de filtro IIR:

Cuando vayamos a diseñar un filtro IIR nos pedirá que tipo de aproximación queremos hacer , teniendo las siguientes posibilidades:

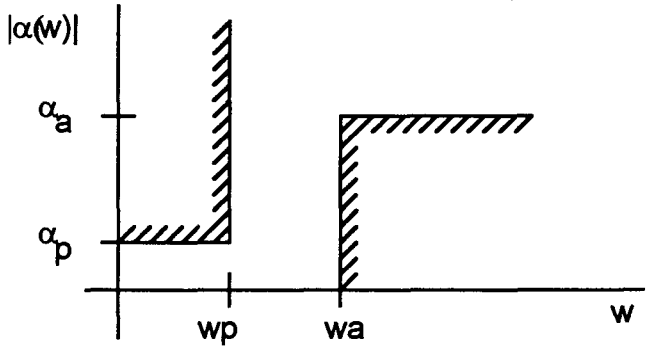
Aproximación Butterworth.
Aproximación Chebyshev Directa.
Aproximación Chebyshev Inversa.
Aproximación Cauer o Elíptica.

Más tarde nos pedirá el tipo de filtro teniendo la posibilidad de:

Paso bajo.
Paso alto.
Paso banda.

Rechazo banda.

Una vez introducidos estos datos tenemos que pasarle las especificaciones en frecuencia que pueden ser en digital o en analógica . Si la especificación es en analógica nos pedirá la frecuencia de muestreo y más tarde las frecuencias de corte del filtro que deben ir entre cero y la mitad de la frecuencia de muestreo. Por el contrario si la especificación en frecuencia es en digital las frecuencias deben ir entre 0 y π . Por último nos pedirá la atenuación en la banda de paso (atenuación máxima en la banda de paso en dB) que debe ser mayor de 0 dB y la atenuación en la banda atenuada (atenuación mínima en la banda atenuada en dB) que ha de ser mayor de 0 dB . A continuación se representa una carátula de especificación paso bajo , el resto de tipos (paso alto , paso banda y rechazo banda) son fácilmente extrapolables a partir de ésta:



Una vez introducidos estos parámetros el programa diseña el filtro y lo guarda en el disco para posteriores aplicaciones.

9.2:ANÁLISIS DE FILTROS DIGITALES.

En esta aplicación tenemos la posibilidad de analizar los filtros diseñados en la aplicación anterior o analizar uno que le indiquemos al programa mediante una ecuación en diferencias , Función racional H(z) o diagrama de polos y ceros.

Al entrar en la aplicación lo primero que nos pedirá es que filtro queremos analizar teniendo las siguientes posibilidades:

 # Respuesta del filtro diseñado anteriormente.
 # Respuesta de un filtro a especificar.

Respuesta del filtro diseñado anteriormente:

Los filtros que vamos a analizar aqui son los que diseñamos en la aplicación anterior dándonos la posibilidad de elegir entre:

FIR Rectangular.
FIR Hamming.
FIR Hannig.
FIR Bartlett.
FIR Blackman.
IIR Butterworth.
IIR Chebyshev Directo.
IIR Chebyshev Inverso.
IIR Eliptico.

Una vez elegido el tipo de filtro a analizar el programa nos mostrará el orden , el tipo (paso bajo, paso alto,...), los coeficientes y nos dirá si es estable o inestable . Si el filtro es estable nos mostrará más características de éste de forma gráfica como son:

Módulo Lineal y en dB.
Fase y retardo de grupo.
Diagrama de polos.
Diagrama de ceros.

En cambio si cojemos la opción:
Respuesta de un filtro a especificar.
Lo primero que nos preguntará es qué tipo de especificación del filtro queremos hacerle teniendo las siguientes posibilidades:

Ecuación en diferencias.
Ecuación Racional de H(Z).
Diagrama de polos y ceros.
Último filtro especificado.

Si elejimos la Ecuación en diferencias tenemos que introducirles los coeficientes de la ecuación en diferencias que caracteriza al filtro . Primero los coeficientes de x[n-k] y más tarde los de y[n-k] separados por comas todos dentro de unos corchetes . Si por ejemplo la ecuación en diferencias es:

$$y[n]=x[n]-3x[n-2]$$

Deberemos escribir:

Coeficientes x[n-k]:**[1,0,-3]**
Coeficientes y[n-k]:**[1]**

Cuando elijamos ecuación racional H(z) deberemos poner los coeficientes de la siguiente ecuación:

$$H(z)=\frac{a_0+a_1z^{-1}+\ldots+a_Nz^{-N}}{b_0+b_1z^{-1}+\ldots+b_Mz^{-M}}$$

Y se introducen de igual manera que la ecuación en diferencias.

Si escogemos diagrama de polos y ceros el programa internamente va a normalizar el filtro para que tenga una ganancia máxima igual a la unidad (salvo cuando sea inestable). Tenemos la posibilidad de entrar los polos y los ceros tanto en coordenadas rectangulares como en polares

Si por ejemplo ejemplo queremos entrar dos polos cuyas coordenadas sean

p1=0.5+0.5*i
p2=0.5-0.5*i

En coordenadas rectangulares tendríamos que escribir:

Polos:**[0.5+i*0.5,0.5-i*.5]**

En cambio en coordenadas polares los polos tendrían las siguientes valores:

p1=0.707|45°
p2=0.707|-45°

y deberíamos escribir:

Polos:**[0.707,45;0.707,-45]**

Si por último cogemos la opoción último filtro especificado analizaremos el ultimo filtro que se le haya especificado al programa.

Una vez entrados los parámetros del filtro especificado el programa nos mostrará los coeficientes de dicho filtro , el orden y si es estable o inestable . Si el filtro resulta ser estable nos mostrará el resto de características de manera gráfica que son las siguientes:

Módulo Lineal y en dB.
Fase y retardo de grupo.
Diagrama de polos.
Diagrama de ceros.

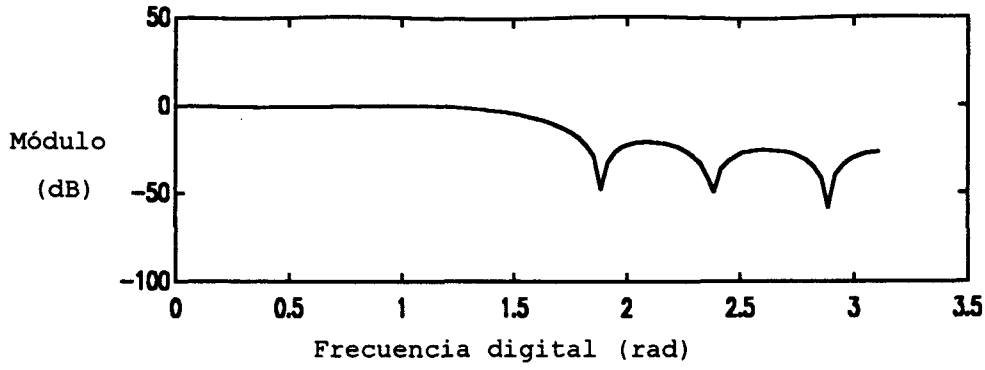
Si por ejemplo el filtro diseñado es un FIR Rectangular de orden 10 , paso bajo y cuya frecuencia de corte es pi/2 el resultado del análisis del programa sería:

Una ecuación en diferencia con los siguientes coeficientes:

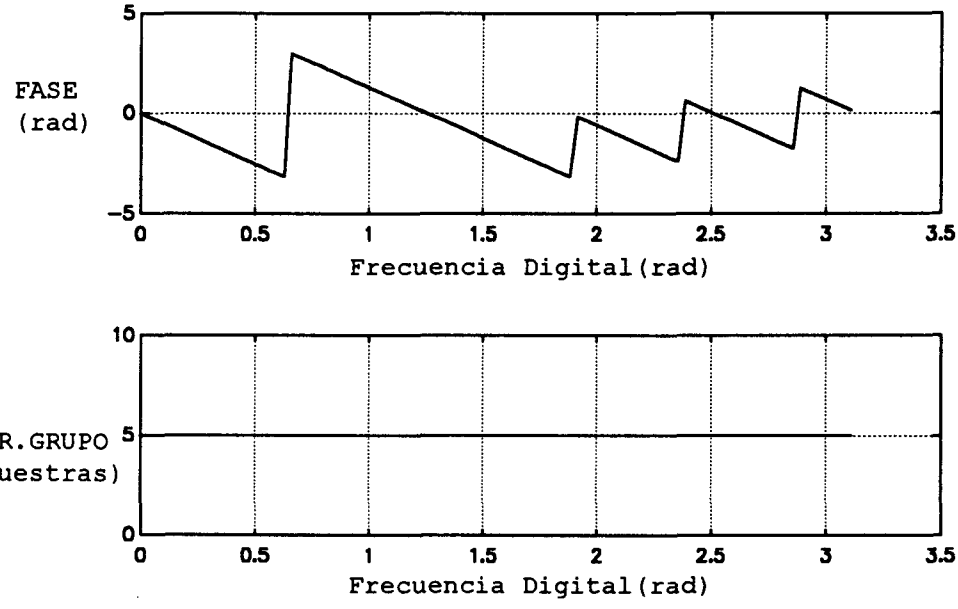
x_n:0.0605;0;-0.1009;0;0.3027;0.4754;0.3027;0;-0.1009;0;0.065

y_n:1

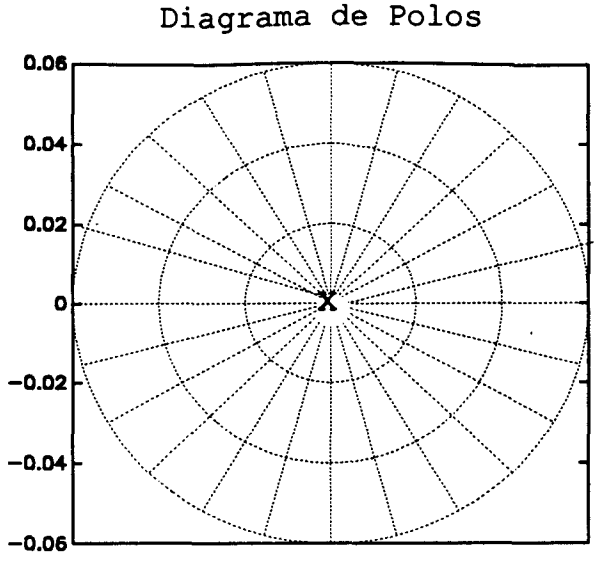
Un módulo como el siguiente:



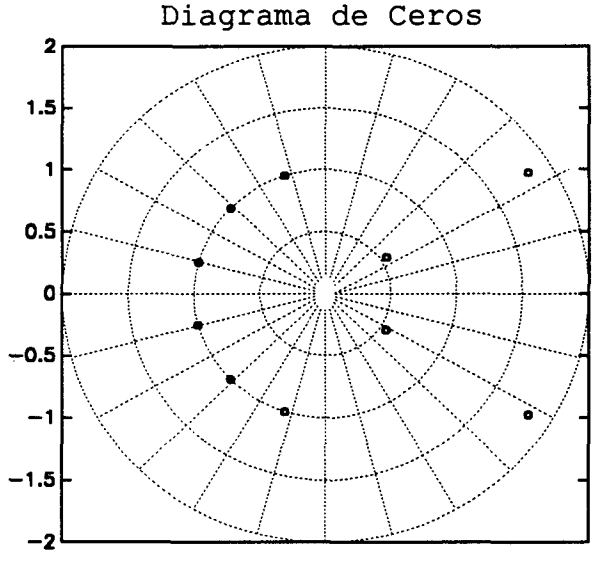
Una fase y un retardo de grupo:



Un diagrama de polos:



Un diagrama de ceros:



9.3:ESTUDIAR FILTRO ANALÓGICO GLOBAL.

En esta aplicación podemos estudiar como se van a comportar los filtros digitales cuando estén trabajando junto con los conversores A/D y D/A con lo cual la respuesta global será analógica y será la que se estudie aquí.

Al entrar en la aplicación nos preguntará que respuesta queremos estudiar , pudiendo elegir entre las siguientes:

 # Respuesta global del filtro diseñado anteriormente.
 # Respuesta global del filtro especificado anteriormente.

Respuesta global del filtro diseñado anteriormente:
Dentro de esta opción lo primero que nos pedirá será la frecuencia de muestreo de los conversores A/D y D/A y seguidamente el filtro que queremos estudiar, donde podremos elegir entre:

FIR Rectangular.
FIR Hamming.
FIR Hannig.
FIR Bartlett.
FIR Blackman.
IIR Butterworth.
IIR Chebyshev Directo.
IIR Chebyshev Inverso.
IIR Eliptico.

Una vez elegido el filtro a estudiar nos mostrará de una manera gráfica y en función de la frecuencia analógica los siguientes parámetros:

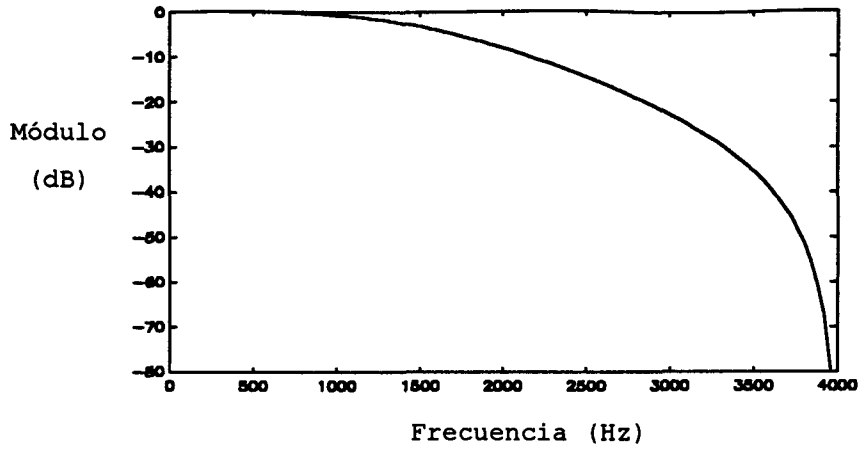
Módulo Lineal.
Módulo en dB.
Retardo de Grupo.

Respuesta global del filtro especificado anteriormente:
En esta opción se hace lo mismo que la anterior solo que con el filtro que le especificamos en la aplicación ANÁLISIS DE FILTROS DIGITALES , por consiguiente una vez dentro nos pedirá la frecuencia de muestreo y una vez introducida nos mostrará de manera gráfica con respecto a la frecuencia analógica los siguientes parámetros del filtro:

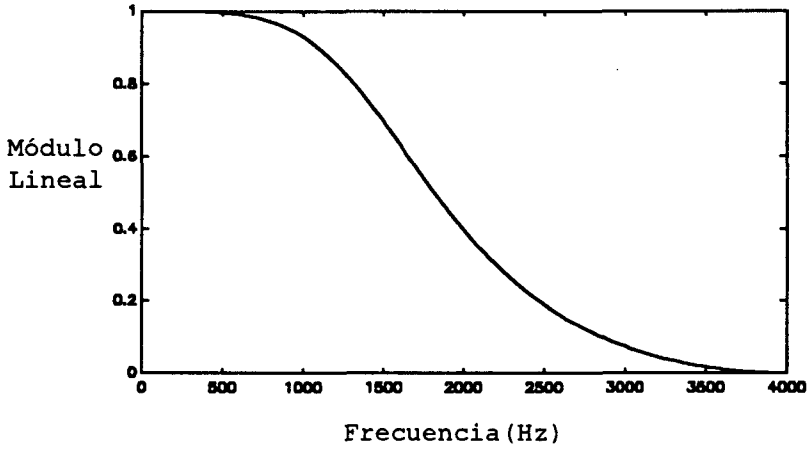
Módulo Lineal.
Módulo en dB.
Retardo de Grupo.

Si como ejemplo diseñamos un filtro de Butterworth con las siguientes características:
Paso bajo.
Frecuecia de paso 0.2*pi y atenuación máxima de 1 dB.
Frecuencia de atenuación 0.5*pi y atenuación mínima 8dB.
Para el equivalente analógico una frecuencia de muestreo de 8000Hz.
Obtenemos los siguientes resultados:

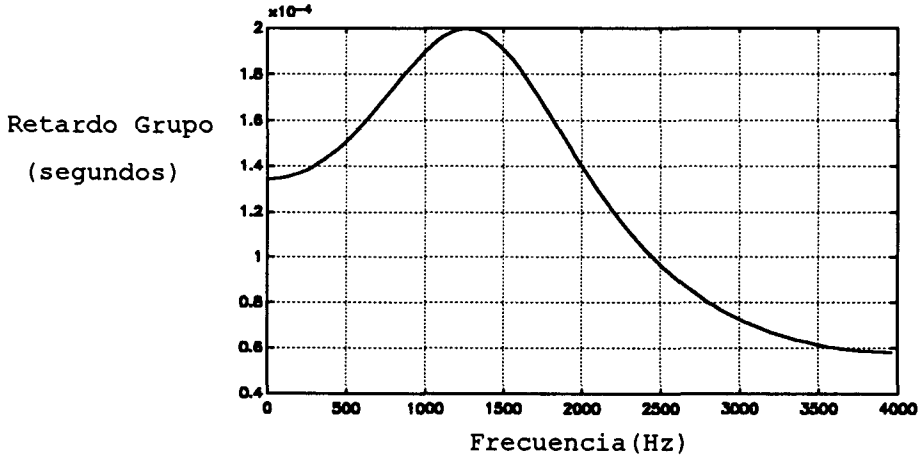
Un módulo en dB:



Un módulo lineal:



Y un retardo de grupo:



9.4:SIMULAR CON SEÑALES DIGITALIZADAS.

En la presente aplicación se pueden pasar señales digitalizadas a través de los filtros para ver como se comportan ante estas señales , además el análisis se hace

de forma global , es decir , se observan las señales analógicas a la entrada del filtro , la respuesta del filtro global y la señal analógica de salida.

Lo primero que nos pide al entrar es el tipo de señal de entrada , pudiendo escoger entre:

Señal de ECG (fm=200 Hz).
Señal de VOZ (fm=8000 Hz).
Otra señal guardada en disco.
Ultima señal tratada.

Estas señales son señales digitalizadas compatibles con MATLAB las cuales se encuantran en los siguientes ficheros: **ecg.mat , voz.mat , ult_sen.mat y otra_s.mat** respectivamente en los cuales el usuario puede poner las señales que le interese tanto si se han obtenido con un sistema de adquisición de datos como si son generadas por un programa como MATLAB.

La señal correspondiente a "otra señal guardada en disco" se encuentra en el fichero **otra_s.mat** y en este el usuario puede guardar la señal que quiera analizar con el programa , la manera de hacerlo es la siguiente:

Mediante un sistema de adquisición (por ejemplo **GLOBAL LAB** o similar) de datos , digitaliza la señal a analizar.
Más tarde se convierte a formato MATLAB mediante el mismo sistema de adquisición de datos.
El fichero debe contener un vector que se debe llamar **otra_s** que será donde esté la señal a analizar.

Por otro lado la señal correspondiente a "Ultima señal tratada" se encuentra en el fichero **ult_sen.mat** y es el resultado del último procesado realizado en esta aplicación ,la cual puede servir para:

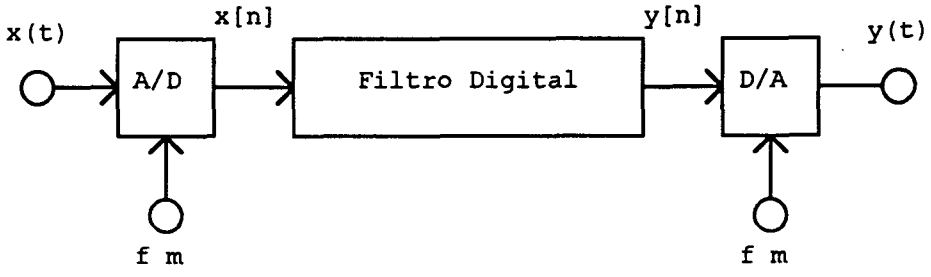
Poder ir realizando algoritmos de procesado en cascada , ya que podemos ir procesando estos resultados e irlos introduciendo en filtros diferentes para ir viendo cómo va evolucionando la señal.
Una vez hayamos hecho un procesado coger esta señal que se encuentra en **ult_sen.mat** e introducirla en un sistema de adquisición como GLOBAL LAB y pasarla a una secuencia analógica para analizarla en el dominio de tiempo contínuo.

Una vez le hayamos indicado el tipo de señal que vamos a pasar por un filtro concreto el programa nos pedirá por que filtro queremos hacerlo y tenemos las siguientes posibilidades:

Filtro FIR diseñado anteriormente.
Filtro IIR diseñado anteriormente.
Filtro especificado anteriormente.

Como último dato de entrada nos pedirá a que frecuencia está muestreada la señal en el disco y que tendrá que coincidir con la que nosotros supusimos en el diseño del filtro digital.

Una vez hecho esto el programa pasará a mostrarnos los resultados de una manera gráfica y considerando un análisis analógico ,es decir , que nos mostrará la señal a la entrada y a la salida junto con la respuesta global del siguiente sistema:



El primer resultado gráfico que muestra es la respuesta del filtro elegido mostrando:

Módulo lineal.
Módulo en dB.
Retardo de grupo.

Más tarde muestra simultaneamente la señal a la entrada del filtro y a la salida , ambas en el dominio temporal y con la posibilidad de hacer un ZOOM para ver una zona concreta de la señal con la posibilidad de ver nuevamente la señal oroginal.

Por último nos muestra la DEE de la señal de entrada simultaneamente con la de la salida para poder observar que componentes de frecuencia a eliminado el filtro . También aquí existe la posibilidad de ZOOM que se detallo en el párrafo anterior.

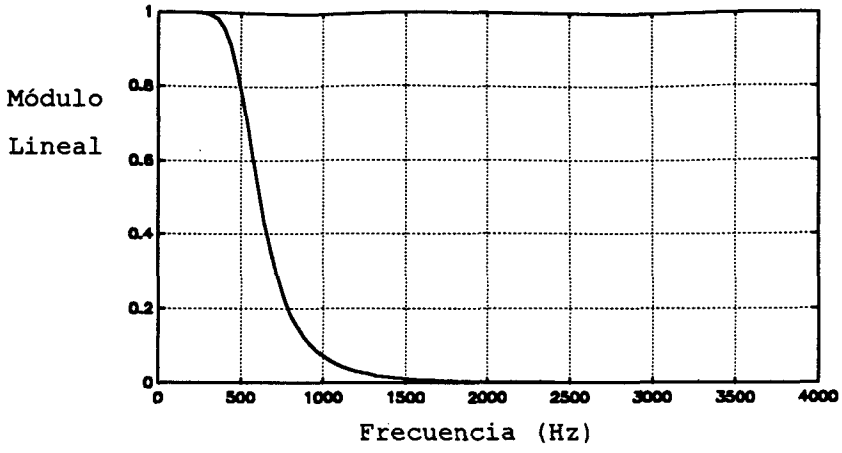
Si por ejemplo diseñamos un filtro de Butterworth paso bajo para una frecuencia de muestreo de 8000Hz cuyas características son:

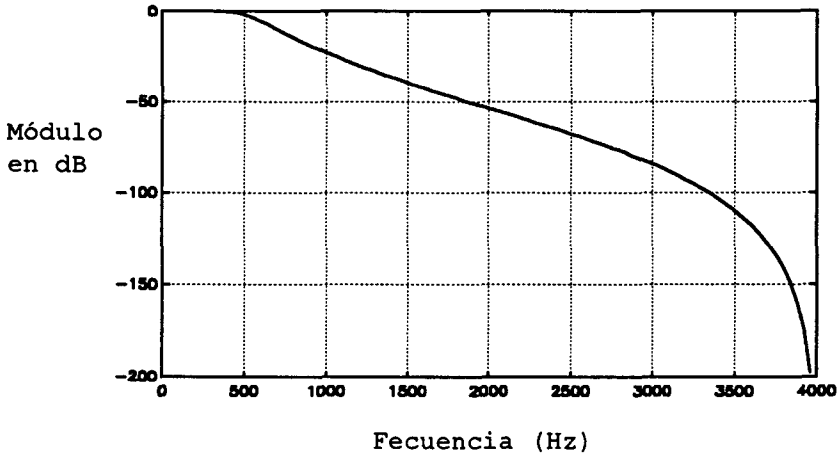
Frecuencia de paso 400 Hz con atenuación máxima de 1 dB.

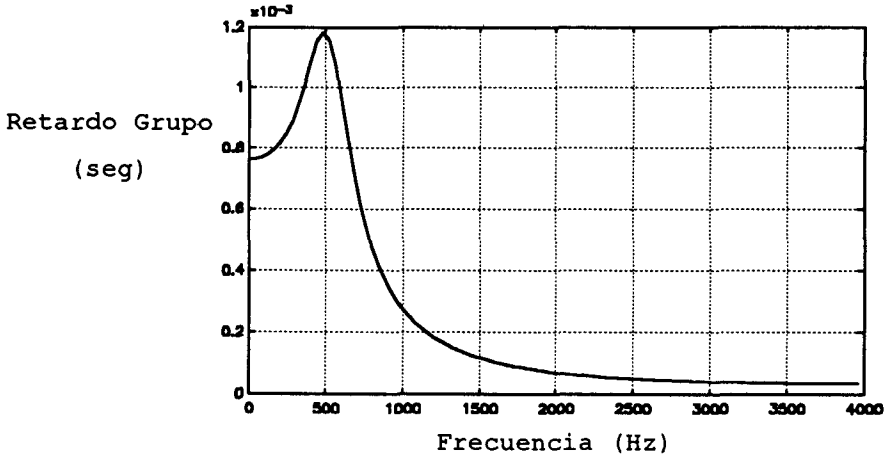
Frecuencia de atenuación 700 Hz con una atenuación mínima 10 dB.

Y le hacemos pasar una señal de voz que se encuentra en el disco muestreada a 8000 Hz los resultados del programa son los siguientes:

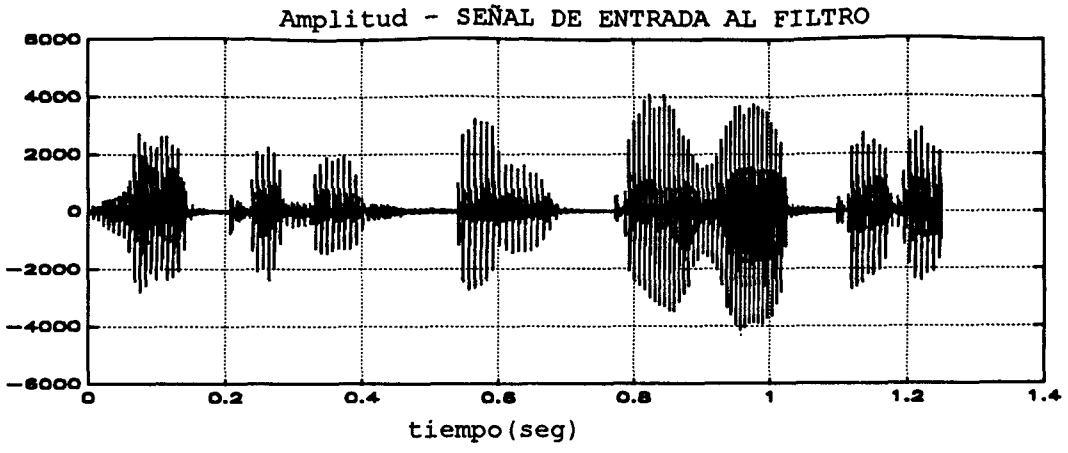
Características del filtro:

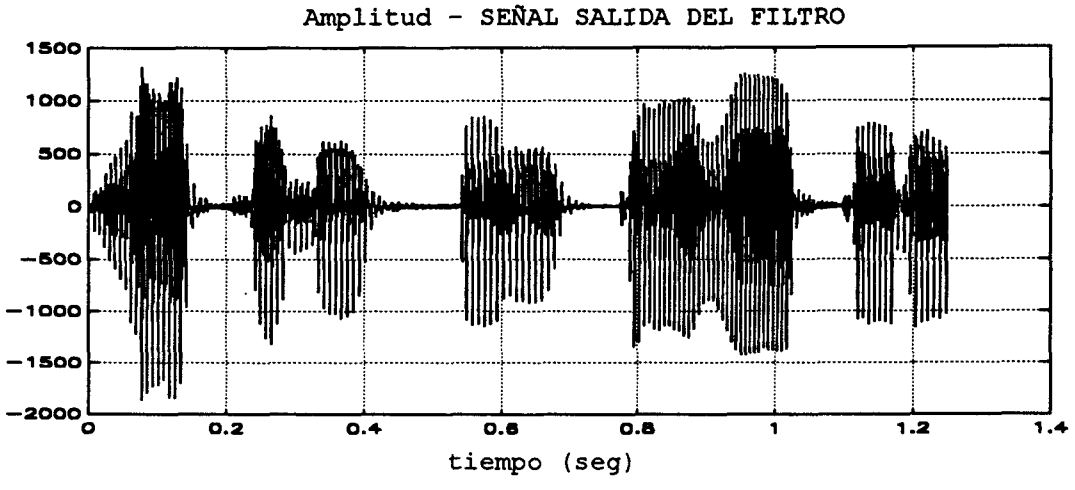




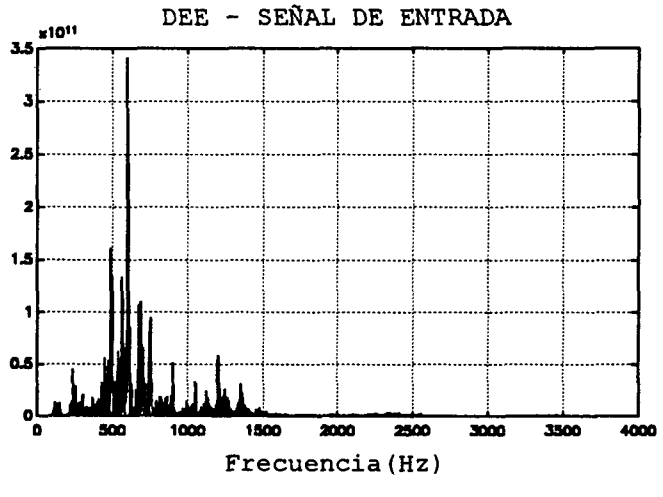


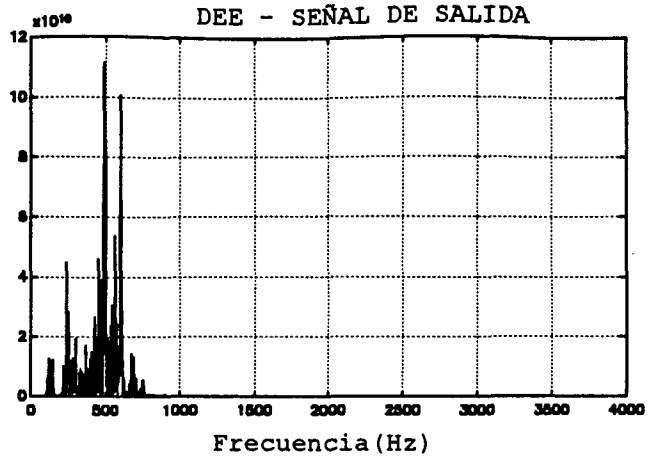
Señal de entrada y salida en el dominio temporal:





DEE de la señal de entrada y salida:





9.5:SIMULAR LOS FILTROS EN EL DSP TMDS3200051.

En todos los apartados anteriores hemos estado utilizando filtros con la presición que nos ofrece el MATLAB (muchísima presición) , a partir de ahora vamos a estudiar esos mismos filtros al introducirlos en un DSP en punto fijo como es el TMDS3200051 y veremos como cambian las características ideales al cuantificar los coeficientes del filtro debido a que el TMDS3200051 trabaja en punto fijo.

Al entrar en la aplicación lo primero que nos pide es que filtro queremos implementar en el DSP de los que hemos estado tratando anteriormente pudiendo elegir entre:

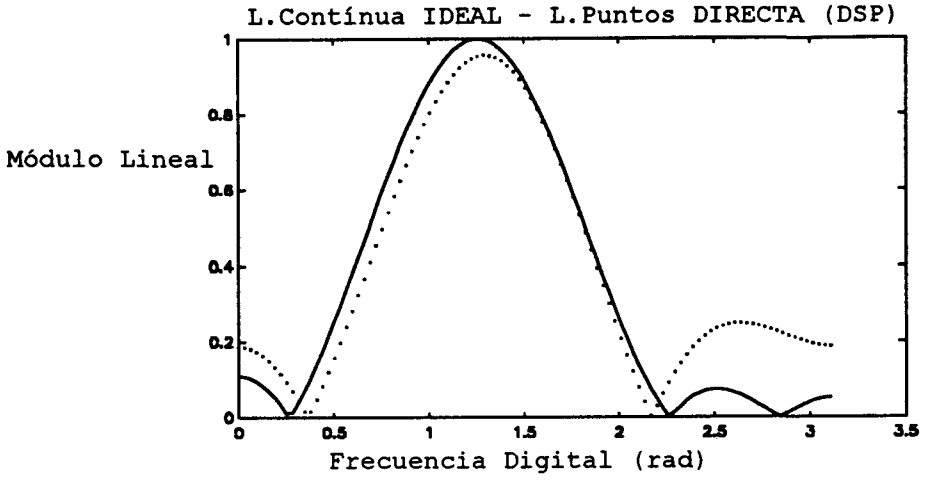
FIR Rectangular.
FIR Hamming.
FIR Hannig.
FIR Bartlett.
FIR Blackman.
IIR Butterworth.
IIR Chebyshev Directo.
IIR Chebyshev Inverso.
IIR Eliptico.
Filtro especificado.

Una vez elegido el filtro a analizar muestra simultaneamente en pantalla las características del filtro ideal y las del mismo filtro trabando en el DSP
tanto para las realizaciones directa , cascada o paralelo según corresponda con filtro FIR o IIR . Al mismo tiempo que se hace la representación gráfica se indica al usuario que realizaciones del filtro serán inestables para que las tenga en cuenta a la hora de implementar dicho filtro en el DSP TMDS3200051.

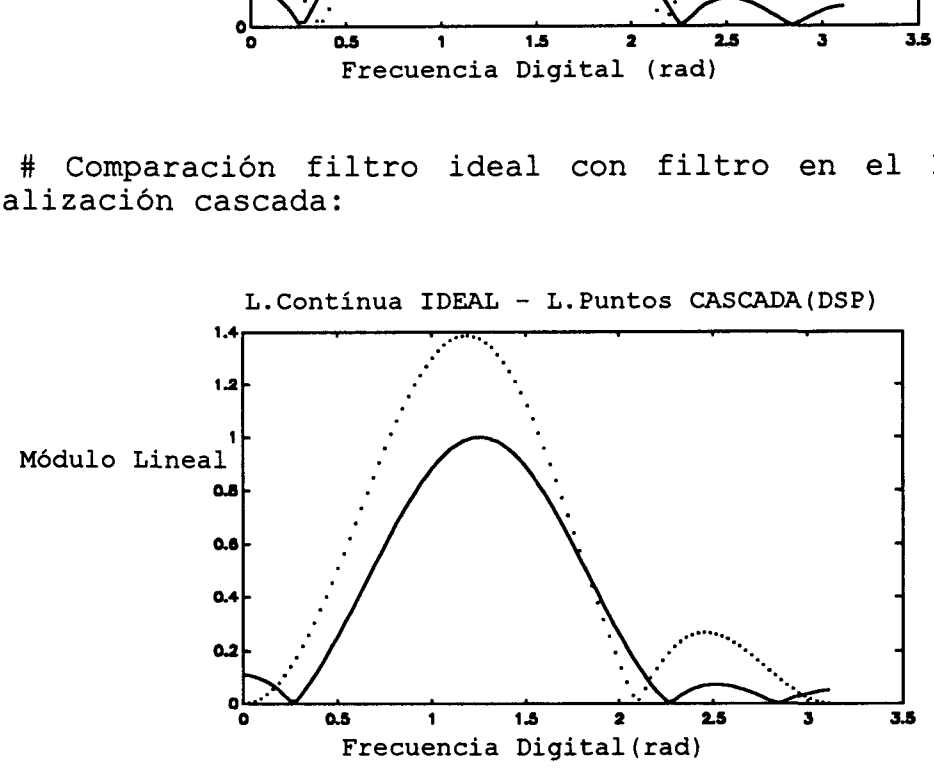
Si por ejemplo diseñamos un filtro FIR paso banda con ventana rectangular de orden 10 y con frecuencias de corte de 0.2*pi y 0.6*pi el programa nos dará unas gráficas de salida como las siguientes (aquí mostraremos sólo el módulo

lineal , el programa también representa el módulo en dB y la fase).

Comparación filtro ideal con filtro en el DSP para realización directa:



Comparación filtro ideal con filtro en el DSP para realización cascada:



9.6:FILTRO ANALÓGICO GLOBAL DEL DSP TMDS3200051.

En esta aplicación podemos analizar como se va comportar el DSP TMDS3200051 desde la entrada analógica hasta la salida analógica , es decir , como se va comportar el sistema global . Lo primero que nos pedirá el programa es qué filtro queremos simular de los diseñados o especificados en los apartados anteriores , seguidamente nos preguntará por las frecuencias de muestreo para el DSP que van de unos valores típicos de 15 KHz hasta 100Hz y por

último según sea el filtro FIR o IIR nos preguntará que realización queremos simular DIRECTA , CASCADA o PARALELO según corresponda . El análisis nos va a arrojar un resultado gráfico mostrándonos:

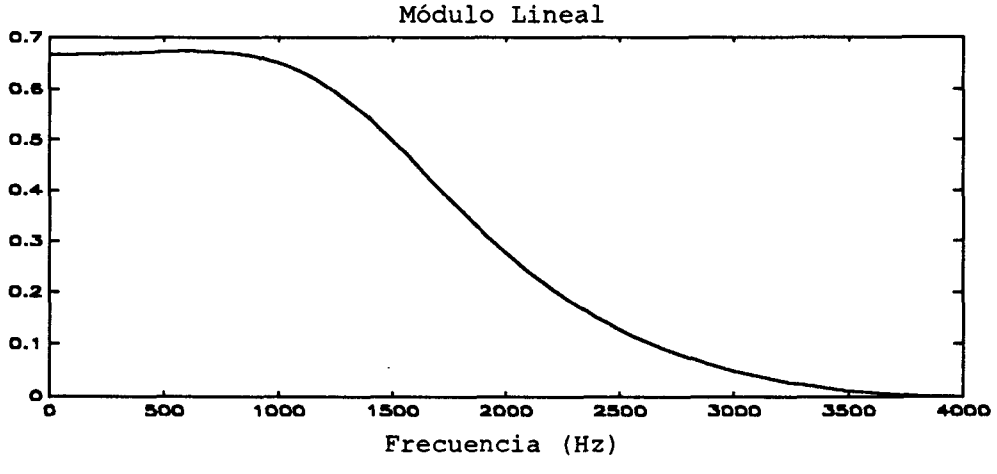
Módulo del filtro lineal.
Módulo en dB.
Fase.

Si por ejemplo diseñamos un filtro con las siguientes características:

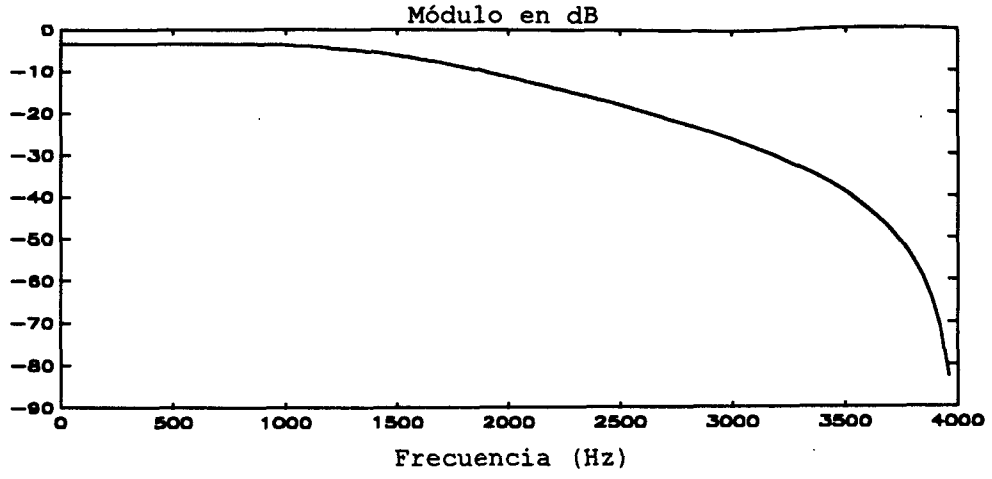
 # Filtro IIR aproximación Butterworth paso bajo.
 # Frecuencia de paso digital $\pi/4$.
 # Frecuencia de atenuación digital $\pi/2$.
 # Atenuación máxima en la banda de paso 1dB.
 # Atenuación mínima en la banda atenuada 9 dB.
 # Frecuencia de muestreo 8 KHz.

Si en el programa queremos ver como es la respuesta del DSP cuando implementemos este filtro con la realización en cascada , el programa arroja el siguiente análisis gráfico:

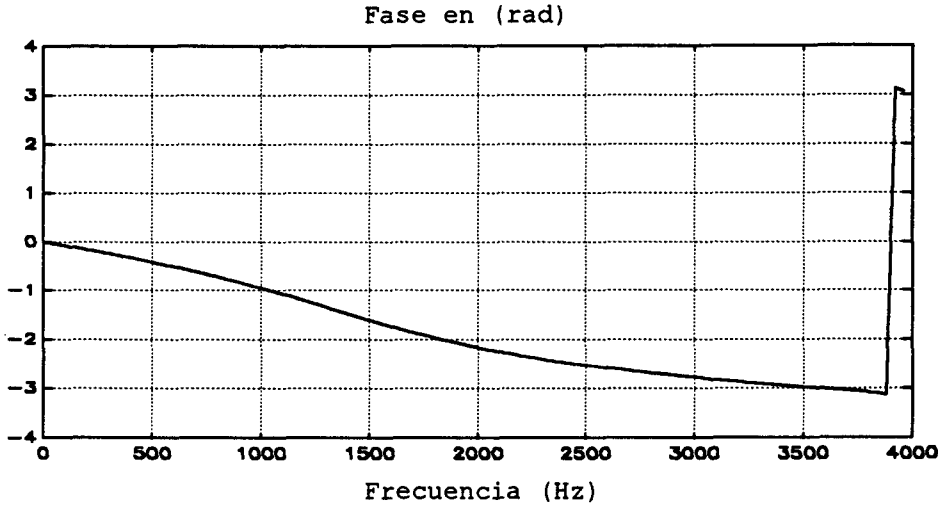
Módulo Lineal:



Módulo en dB:



Fase:



9.7:SIMULAR CON SEÑALES DIGITALIZADAS EN EL DSP TMDS3200051.

En esta aplicación podemos realizar los siguientes estudios:

Al principio el programa nos da a escoger la señal que queremos poner en la entrada del filtro:

Señal de ECG.
Señal de VOZ.
Otra señal guardada en disco.
Ultima señal tratada.

Estas señales se pueden utilizar de manera análoga a como se utilizaban en el **apartado 9.4** con la salvedad de que ahora las señales digitalizadas deben estar codificadas con 14 bits (13 bits de módulo + 1 bit de signo) ya que es así como las trata el DSP TMDS3200051 . Igual que

explicamos en el apartado 9.4 las señales pueden ser cambiadas por otras nuevas para un análisis concreto o para sacar los resultados de un determinado procesado para ello solo basta con utilizar los ficheros con formato Matlab siguientes:

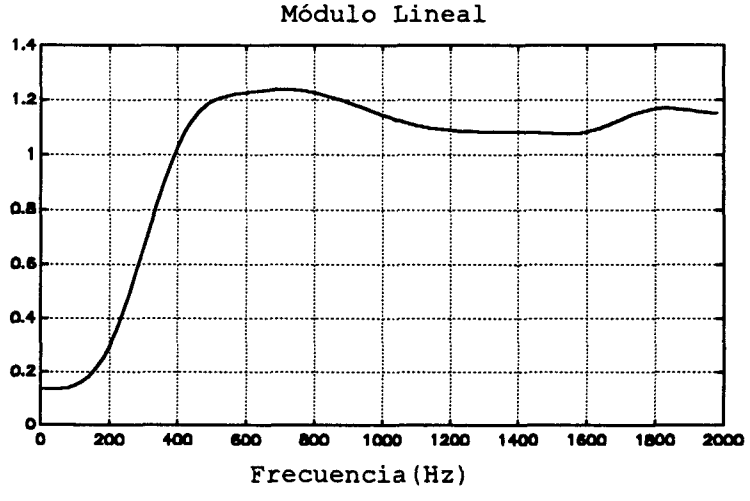
Señal de ECG ; fichero **ecgt.mat**
Señal de VOZ ; fichero **vozt.mat**
Otra señal en disco ; fichero **otra_st.mat**
Ultima señal tratada ; fichero **ult_st.mat**

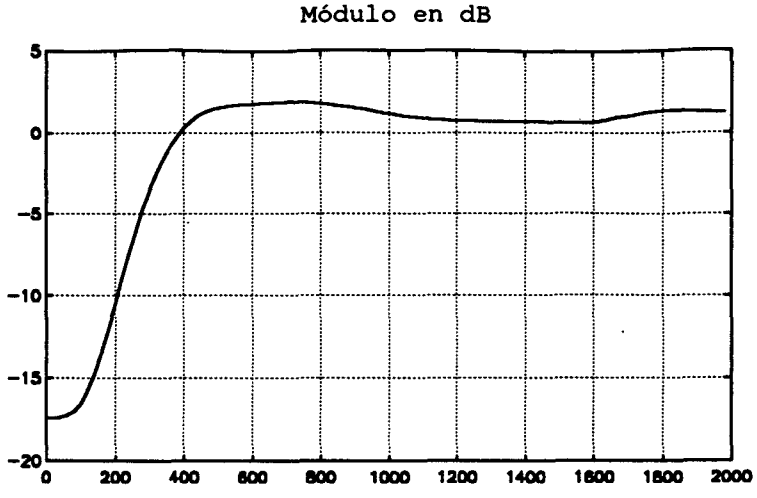
Una vez que le indiquemos al programa que señal vamos a procesar este nos pedirá con que filtro queremos tratarla , luego nos pedirá la frecuencia de muestreo que va típicamente de 15KHz hasta 100Hz y según el tipo de filtro elegido nos pedirá el tipo de realización a utilizar . Como resultado del análisis el programa arroja los siguientes resultados gráficos:

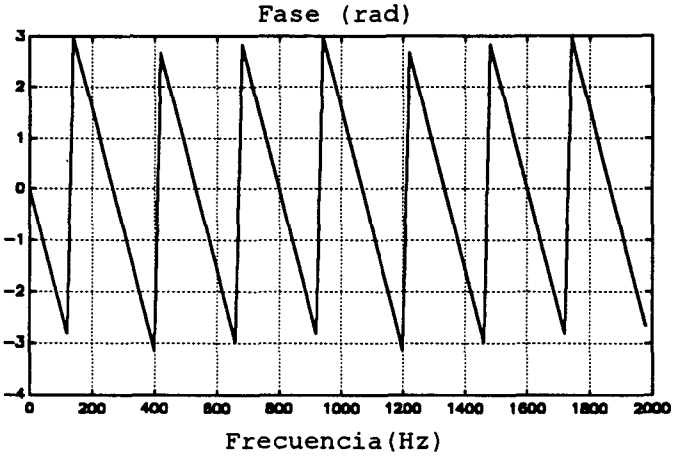
Primeramente muestra las características del filtro que se está utilizando mostrando módulo lineal , módulo en dB y fase todo esto en función de la frecuencia analógica . A continuación muestra las señales de entrada y salida en el dominio temporal pudiendo hacer un ZOOM para observar mejor la señal en estudio . Por último muestra la DEE de la señal de entrada frente a la DEE de la señal de salida con la posibilidad de hacer un ZOOM como en el apartado anterior.

Si por ejemplo hemos diseñado un filtro FIR HAMMING paso alto de orden 30 con una frecuencia de muestreo de 4000 Hz y una frecuencia de corte de 300 Hz y lo atacamos con una señal arbitraria(En este caso son dos tonos) y simulamos la realización directa el DSP tendrá una respuesta como la siguiente:

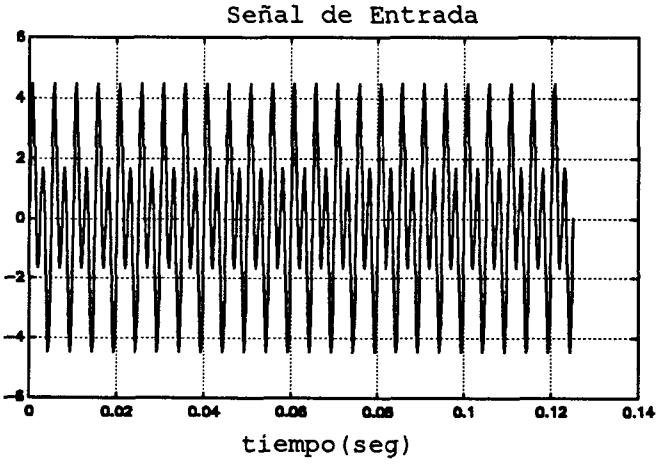
La respuesta global del DSP será:

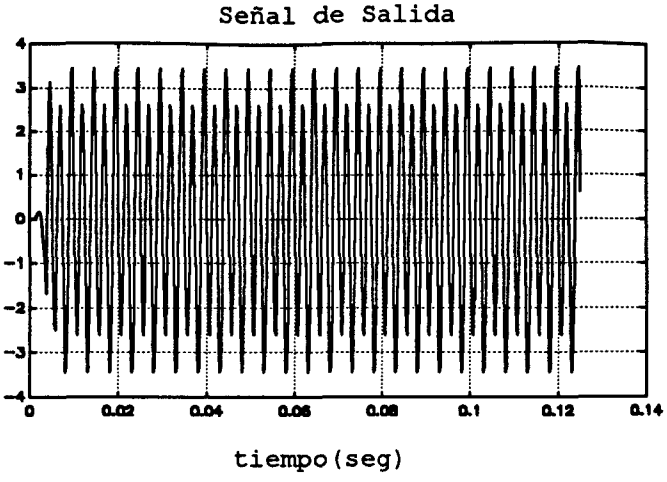




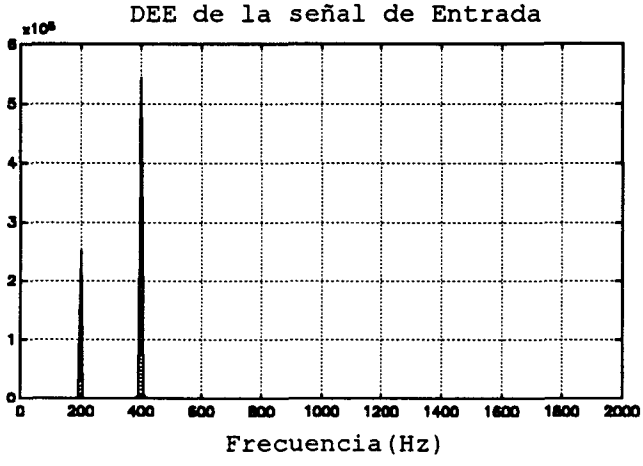


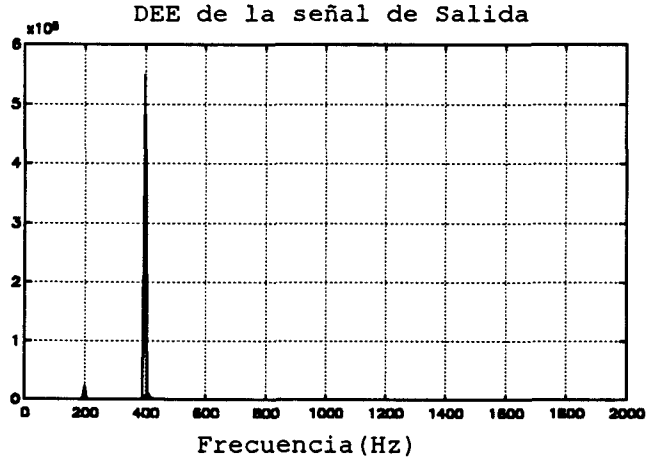
La señal de entrada y salida en el dominio temporal son:





La DEE de la señal de entrada y salida son:





Como se observa en esta aplicación podemos saber como se va comportar el DSP ante una señal de entrada concreta sin tener que trabajar con éste.

Un procedimiento bastante importante que podemos realizar tanto en esta aplicación como la descrita en el apartado 9.4 es que si no tenemos señales digitalizadas con

las que queremos trabajar podemos sintetizarlas con un lenguaje como el MATLAB.

9.8:PROGRAMACIÓN DEL DSP TMDS3200051.

Esta aplicación nos dará la posibilidad de programar los conversores A/D y D/A del DSP así como los filtros que hayamos diseñado con el programa DISFILT . La manera de hacerlo es la siguiente:

En el presente Trabajo Fin de Carrera se han desarrollado unos programas en ensamblador para el DSP TMDS3200051 los cuales ya tienen rutinas para programar los conversores A/D y D/A y otras para programar filtros FIR como IIR . Para que estos algoritmos funcionen solo les faltan unos valores que se deben poner en memoria para indicarle las frecuencias de muestreo de los conversores y los coeficientes de los filtros en cuestión que queramos implementar . Por consiguiente esta aplicación nos dará los valores que se deben poner en memoria para una aplicación concreta y así ya tendremos los algoritmos programados en el DSP.

Al entrar en esta aplicación nos pedirá que queremos programar del DSP:

Conversores A/D y D/A.
Filtro FIR diseñado.
Filtro IIR diseñado.
Filtro Especificado.

Si elegimos conversores A/D y D/A nos pedirá a que frecuencia de muestreo queremos que trabajen teniendo un margen que va desde 100 Hz hasta los 15 KHz , una vez elegida esta nos mostrará que valores deben tener los registros **R_TCR** , **P_PRD** , **TA** , **RA** , **TB** y **RB** que serán los valores que tengamos que poner en la memoria del DSP . También nos dirá las frecuencias mínima y máxima de entrada que podemos procesar con esta frecuencia de muestreo.

Cuando elijamos programar tanto filtro FIR , IIR o Especificado el programa nos dará los valores de los coeficientes de dicho filtro , es decir , los coeficientes de la ecuación en diferencias (coeficientes ya cuantificados para poderlos implementar en el DSP TMDS3200051) que lo caracteriza y que tiene la siguiente forma:

$$\sum_{k=0}^{N} a_k y[n-k] = \sum_{k=0}^{M} b_k x[n-k]$$

Si por ejemplo diseñamos un filtro con las siguientes especificaciones:

Aproximación: **Butterworth paso bajo.**

Frecuencia muestreo: **8 KHz.**
Frecuencia de paso: **2 KHz.**
Frecuencia de atenuación: **3 KHz.**
Atenuación máxima banda de paso: **1 dB.**
Atenuación mínima banda atenuada: **6 dB.**

Si una vez realizado un análisis con DISFILT nos interesa programarlo por ejemplo con la realización directa el programa nos arrojará los siguientes resultados:

Conversores A/D y D/A:

Registro R_TCR: **20h.**
Registro P_PRD: **01h.**
Registros TA y RA: **16.**
Registros TB y RB: **41.**
Frecuencia mínima de entrada: **230 Hz.**
Frecuencia máxima de entrada: **4 KHz.**

Filtro IIR diseñado:

```
Los coeficientes de x[n-k]:
  b0:  2   16380   1
  b1:  4     0     0
  b2:  2   16380   1

Los coeficientes de y[n-k]:
  a0:  0     1     0
  a1:  2   16380   1
  a2:  2    8188   2
```

9.9:AYUDA PARA UTILIZAR EL PROGRAMA.

En esta parte el programa lo que muestra es una serie de ayudas para cada una de las aplicaciones antes explicadas , si esta ayuda nos es suficiente se debe consultar el presente Manual de usuario.

CAPITULO 10: ANEXO
MANUAL DE PROGRAMACIÓN DISFILT.

El programa DISFILT consta de una serie de ficheros los cuales deben estar en un directorio en el cual el MATLAB pueda acceder a ellos cuando se invoquen , es decir , que exista un **path** de Matlab hacia ese directorio.

DISFILT consta de un programa principal y de una serie de funciones asociadas a éste . El programa principal se encuentra en el fichero **disfilt.m** y las funciones asociadas en los ficheros:

casc_fir.m
casc_iir.m
coef_ent.m
con_ad.m
cu_coef.m
cu_cte.m
cuan_sen.m
elip_es.m
estab_f.m
f_but_es.m
f_fir.m
menu_fir.m
menu_iir.m
parl_iir.m
present.m
pro_coef.m
re_c_s.m
re_cas.m
re_co_fi.m
re_fi_a.m
re_fi_an.m
re_filt.m
re_p_r.m
re_par.m
re_se_d.m
re_sen.m
re_sent.m
rf.m
tchd_es.m
tchi_es.m

Las líneas de código que tienen estos ficheros son:

LISTADO DE DISFILT.M

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%      PROGRAMA PRINCIPAL DE DISEÑO Y ANALISIS DE FILTROS              %%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
clear;
clc;
close;
present   % pantalla de presentaci³n inicial
kiko=menu('DISFILT','Iniciar..');
close;
clc;

salir=-1;
disp('<ESPERE> verificando la estructura del programa.....');

while ((salir<1)|(salir>1)) %%%  BUCLE PRINCIPAL DEL PROGRAMA
                            %%% Engloba todo el programa.

%%%%%%%%%%%%%%%%%      ELECCION DE LA APLICACION    %%%%%%%%%%%%%%%%%%%%%%%%%
 eleccion=-1; %%% Escogemos la aplicaci³n sobre la que queremos trabajar
 while ((eleccion<1)|(eleccion>10))
   clc
   disp('                      ----------------------------');
   disp('                      ------ MENU PRINCIPAL ------');
   disp('                      ----------------------------');
   eleccion=menu('ELIGE TIPO DE APLICACION:','DISEÑO DE FILTROS DIGITALES.','AN-LISIS DE
FILTROS DIGITALES.','ESTUDIAR FILTRO ANALËGICO GLOBAL.',...
       'SIMULAR  CON  SEÑALES  DIGITALIZADAS.','SIMULAR  LOS  FILTROS  EN  EL  DSP
TMDS3200051.','FILTRO ANALËGICO GLOBAL DEL DSP TMDS3200051',...
    'SIMULAR CON SEÑALES DIGITALIZADAS EN EL DSP TMDS3200051.','PROGRAMACIËN DEL DSP
TMDS3200051.','AYUDA PARA UTILIZAR EL PROGRAMA.','FIN.');
 end

% ***************************************************************************

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%    PROGRAMA DE DISEÑO DE FILTROS DIGITALES    %%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%  Las variables x_n,y_n ,n y tipo se guardan en el disco.
%%%  Se trata del programa general de diseño de filtros el cual diseña
%%%    el filtro y guarda los coeficientes en le disco.
if (eleccion==1) %%%%%  Elecci³n del programa de diseño de filtros
clc
men_1 = ('          ');  % mensajes para mostrar por pantalla
men_2 = ('             Espere calculando el FILTRO ...  ');
men_4 = ('           *** DISEÑO DE FILTROS DIGITALES *** ');

   elec_d_f=-1;  %% Eleccion de tipo de filtro FIR o IIR
   while ((elec_d_f<1)|(elec_d_f>2))
      clc
      disp (men_4);
          elec_d_f=menu('TIPO  DE  FILTRO  DIGITAL:','FIR  (mÚtodo  de  ventanas).','IIR
(transformaci³n bilineal).');
   end

   %*************************************************************************
   if (elec_d_f==1)  %%%%%%%   Cuando elijamos diseño de filtros FIR
                     %%%%%%%    mediante el método de las ventanas
        %% En el disco guardamos coeficientes,orden y tipo.
       elec_fir=-1;
       while((elec_fir<1)|(elec_fir>5))
          clc
          disp(men_4); % elección de diferentes tipos de ventanas
              elec_fir=menu('TIPO DE FILTRO FIR:','FIR - V. RECTANGULAR.','FIR - V.
HAMMING.','FIR - V. HANNING','FIR - V. BARTLETT','FIR - V. BLACKMAN');
       end

            if (elec_fir==1) %%%%%%%%%%% diseño filtro FIR RECTANGULAR
              [n,w_paso,tipo,error] = menu_fir; % pedimos parámetros del filtro
              disp (men_1);
              disp (men_2);
              if (error==0)
                [x_n,y_n] = f_fir(w_paso,n,tipo,1); % calculamos el filtro
                save fir_rect.mat x_n y_n n tipo; % guardamos en el disco
                kiko=menu('Filtro en DISCO','CONTINUAR');
              else
                kiko=menu('parametros **ERRONEOS**','CONTINUAR');
              end
            end
```

```
            if (elec_fir==2) %%%%%%%%%%%% diseño filtro FIR HAMMING
              [n,w_paso,tipo,error] = menu_fir; % pedimos parámetros del filtro
              disp (men_1);
              disp (men_2);
              if (error==0)
                [x_n,y_n] = f_fir(w_paso,n,tipo,2); % calculamos el filtro
                save fir_hamm.mat x_n y_n n tipo; % guardamos en el disco
                kiko=menu('Filtro en DISCO','CONTINUAR');
              else
                kiko=menu('parametros **ERRONEOS**','CONTINUAR');
              end
            end

            if (elec_fir==3) %%%%%%%%%%%% diseño filtro FIR HANNING
              [n,w_paso,tipo,error] = menu_fir; % pedimos parámetros del filtro
              disp (men_1);
              disp (men_2);
              if (error==0)
                [x_n,y_n] = f_fir(w_paso,n,tipo,3); % calculamos el filtro
                save fir_hann.mat x_n y_n n tipo; % guardamos en el disco
                kiko=menu('Filtro en DISCO','CONTINUAR');
              else
                kiko=menu('parametros **ERRONEOS**','CONTINUAR');
              end
            end

            if (elec_fir==4) %%%%%%%%%%%% diseño filtro FIR BARTLETT
              [n,w_paso,tipo,error] = menu_fir; % pedimos parámetros del filtro
              disp (men_1);
              disp (men_2);
              if (error==0)
                [x_n,y_n] = f_fir(w_paso,n,tipo,4); % calculamos el filtro
                save fir_bart.mat x_n y_n n tipo; % guardamos en el disco
                kiko=menu('Filtro en DISCO','CONTINUAR');
              else
                kiko=menu('parametros **ERRONEOS**','CONTINUAR');
              end
            end

            if (elec_fir==5) %%%%%%%%%%%% diseño filtro FIR BLACKMAN
              [n,w_paso,tipo,error] = menu_fir; % pedimos parámetros del filtro
              disp (men_1);
              disp (men_2);
              if (error==0)
                [x_n,y_n] = f_fir(w_paso,n,tipo,5); % calculamos el filtro
                save fir_blac.mat x_n y_n n tipo; % guardamos en el disco
                kiko=menu('Filtro en DISCO','CONTINUAR');
              else
                kiko=menu('parametros **ERRONEOS**','CONTINUAR');
              end
            end

   end
   %*************************************************************************
   if (elec_d_f==2)  %%%%%%%   Cuando elijamos diseño de filtros IIR
                     %%% mediante transformación bilineal
        elec_iir=-1
        while((elec_iir<1)|(elec_iir>4))
           clc
           disp(men_4);
                     elec_iir=menu('TIPO  DE  FILTRO  IIR:','BUTTERWORTH.','CHEBYSHEV
DIRECTO.','CHEBYSHEV INVERSO.','ELIPTICO.');
        end

            if (elec_iir==1) %%%%%%%% diseño filtro IIR Butterworth
               [wp,wa,ap,aa,tipo,error]=menu_iir; % pedimos parámetros
               disp(men_1);
               disp(men_2);
               if (error==0)
                 [x_n,y_n,n]=f_but_es(wp,wa,ap,aa,tipo); % calculamos el filtro

                   if (tipo>2)  % cuando sea Paso-banda o Banda-eliminada
                     n=2*n;     % el orden es el doble.
                   end

                 save butter.mat x_n y_n n tipo; % guardamos en el disco
                 kiko=menu('Filtro en DISCO','CONTINUAR');
               else
```

```
            kiko=menu('parametros **ERRONEOS**','CONTINUAR');
          end
        end

        if (elec_iir==2) %%%%%%%% diseño filtro IIR Chebyshev directo
          [wp,wa,ap,aa,tipo,error]=menu_iir; % pedimos parámetros
          disp(men_1);
          disp(men_2);
          if (error==0)
            [x_n,y_n,n]=tchd_es(wp,wa,ap,aa,tipo); % calculamos el filtro

              if (tipo>2)  % cuando sea Paso-banda o Banda-eliminada
                n=2*n;     % el orden es el doble.
              end

            save ched.mat x_n y_n n tipo; % guardamos en el disco
            kiko=menu('Filtro en DISCO','CONTINUAR');
          else
            kiko=menu('parametros **ERRONEOS**','CONTINUAR');
          end
        end

        if (elec_iir==3) %%%%%%%% diseño filtro IIR Chebyshev inverso
          [wp,wa,ap,aa,tipo,error]=menu_iir; % pedimos parámetros
          disp(men_1);
          disp(men_2);
          if (error==0)
            [x_n,y_n,n]=tchi_es(wp,wa,ap,aa,tipo); % calculamos el filtro

              if (tipo>2)  % cuando sea Paso-banda o Banda-eliminada
                n=2*n;     % el orden es el doble.
              end

            save chei.mat x_n y_n n tipo; % guardamos en el disco
            kiko=menu('Filtro en DISCO','CONTINUAR');
          else
            kiko=menu('parametros **ERRONEOS**','CONTINUAR');
          end
        end

        if (elec_iir==4) %%%%%%%% diseño filtro IIR Eliptico
          [wp,wa,ap,aa,tipo,error]=menu_iir; % pedimos parámetros
          disp(men_1);
          disp(men_2);
          if (error==0)
            [x_n,y_n,n]=elip_es(wp,wa,ap,aa,tipo);% calculamos el filtro

              if (tipo>2)  % cuando sea Paso-banda o Banda-eliminada
                n=2*n;     % el orden es el doble.
              end

            save elip.mat x_n y_n n tipo; % guardamos en el disco
            kiko=menu('Filtro en DISCO','CONTINUAR');
          else
            kiko=menu('parametros **ERRONEOS**','CONTINUAR');
          end
        end

    end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%   fin de diseño de filtros digitales      %%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% ************************************************************************

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%    PROGRAMA DE ANALISIS DE FILTROS DIGITALES    %%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%  Las variables x_n,y_n,n y tipo están en el disco

if (eleccion==2) %%%%%  Analisis de filtros
clc
more on
men_1=('          *** AN-LISIS DE FILTROS DIGITALES *** ');
   elec_a_f=-1;
   while ((elec_a_f<1)|(elec_a_f>2))
```

```
      clc
      disp (men_1);
         elec_a_f=menu('ELIGE EL TIPO DE AN-LISIS:','RESPUESTA DEL FILTRO DISEÑADO
ANTERIORMENTE.','RESPUESTA DE UN FILTRO A ESPECIFICAR.');
   end

   %************************************************************************
   if (elec_a_f==1)  %%%%%%%   Respuesta filtro diseñado.

        elec_f_d=-1;
        while((elec_f_d<1)|(elec_f_d>9))
           clc
           disp(men_1);
                 elec_f_d=menu('TIPO DE FILTRO A ANALIZAR DE LOS DISEÑADOS:','FIR
RECTANGULAR.','FIR          HAMMING','FIR          HANNING.','FIR          BARTLETT','FIR
BLACKMANN','BUTTERWORTH.','CHEBYSHEV DIRECTO.','CHEBYSHEV INVERSO.','ELIPTICO.');
        end
            %%%%%   filtros FIR

            if (elec_f_d==1) % Filtro FIR RECTANGULAR
              load fir_rect.mat;
              disp('--- FILTRO FIR DE ORDEN: '),disp(n);
            end

            if (elec_f_d==2) % Filtro FIR HAMMING
              load fir_hamm.mat;
              disp('--- FILTRO FIR DE ORDEN: '),disp(n);
            end

            if (elec_f_d==3) % Filtro FIR HANNING
              load fir_hann.mat;
              disp('--- FILTRO FIR DE ORDEN: '),disp(n);
            end

            if (elec_f_d==4) % Filtro FIR BARTLETT
              load fir_bart.mat;
              disp('--- FILTRO FIR DE ORDEN: '),disp(n);
            end

            if (elec_f_d==5) % Filtro FIR BLACKMAN
              load fir_blac.mat;
              disp('--- FILTRO FIR DE ORDEN: '),disp(n);
            end

           %%%%%%%   filtros IIR

            if (elec_f_d==6) % Filtro BUTTERWORTH
              load butter.mat
            end

            if (elec_f_d==7) % Filtro CHEBYSHEV DIRECTO
              load ched.mat
            end

            if (elec_f_d==8) % Filtro CHEBYSHEV INVERSO
              load chei.mat
            end

            if (elec_f_d==9) % Filtro ELIPTICO.
              load elip.mat
            end

                 if(elec_f_d>5)
                   disp('--- FILTRO IIR DE ORDEN: '),disp(n);
                 end

   end

   %************************************************************************
   if (elec_a_f==2)  %%%%%%%   Respuesta filtro a especificar.

        elec_f_e=-1;
        while((elec_f_e<1)|(elec_f_e>4))
           clc
           disp(men_1);
            elec_f_e=menu('TIPO DE FILTRO A ANALIZAR SEGUN ESPECIFICACION:','ECUACION EN
DIFERENCIAS.','FUNCION   RACIONAL   H(Z).','DIAGRAMA   POLOS-CEROS.','ULTIMO   FILTRO
ESPECIFICADO GUARDADO EN DISCO.');
```

```
        end

            if (elec_f_e==1) % Ecuación en diferencias
               [x_n,y_n]=coef_ent(1);
            end

            if (elec_f_e==2) % Función racional H(Z)
               [x_n,y_n]=coef_ent(2);
            end

            if (elec_f_e==3) % polos y ceros
               [x_n,y_n]=coef_ent(3);
            end

            if (elec_f_e<=3)
               save fil_ep.mat x_n y_n %%% Guardamos filtro especificado en disco
            end

            if (elec_f_e==4)
               load fil_ep.mat
            end


                  if length(y_n)>1
                      disp('--- FILTRO IIR DE ORDEN: ')
                  else
                      disp('--- FILTRO FIR DE ORDEN: ')
                  end

                  if (length(y_n)>length(x_n))
                      disp(length(y_n)-1);
                  else
                      disp(length(x_n)-1);
                  end
   end  % fin de filtro a especificar

if (elec_a_f==1)
   if (tipo==1)
      disp('--- FILTRO PASO-BAJO');
   end

   if (tipo==2)
      disp('--- FILTRO PASO-ALTO');
   end

   if (tipo==3)
      disp('--- FILTRO PASO-BANDA');
   end

   if (tipo==4)
      disp('--- FILTRO RECHAZO_BANDA');
   end
end

   disp('   Pulsa RETURN para ver los coeficientes ...');
   disp('--- Los COEFICIENTES de x[n-k] SON: '),disp(x_n);
   disp('--- Los COEFICIENTES de y[n-k] SON: '),disp(y_n);
   more off
   estabilidad=estab_f(x_n,y_n,0);

       if (estabilidad==0)
         kiko=menu('**filtro INESTABLE**','CONTINUAR');
       else
         kiko=menu('Filtro ESTABLE','CONTINUAR');
         clc
         re_filt(x_n,y_n);
       end

end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%   fin de analisis de filtros digitales      %%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%*************************************************************************

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%          PROGRAMA DE FILTROS ANALOGICOS          %%%%%%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%  Las variables x_n,y_n,n y tipo están en el disco

if (eleccion==3)
  men_1=('               *** FILTRO ANALOGICO GLOBAL *** ');
    ele_fa=-1;
      while ((ele_fa<1)|(ele_fa>2))
        clc
        disp(men_1);


         ele_fa=menu('TIPO DE AN-LISIS ANALÊGICO:','RESPUESTA GLOBAL DEL FILTRO DISEÐADO
ANTERIORMENTE.','RESPUESTA GLOBAL DEL FILTRO ESPECIFICADO ANTERIORMENTE.');
      end
      f_m=input('Entra la FRECUENCIA de MUESTREO (Hz): ');

 if (f_m>0)
           if(ele_fa==1)  %%%%%%%   Respuesta global del filtro diseñado ideal
              disp(men_1);
                   elec_f_d=-1
                    while((elec_f_d<1)|(elec_f_d>9))
                      clc
                      disp(men_1);
                        elec_f_d=menu('TIPO DE FILTRO A ANALIZAR DE LOS DISEÐADOS:','FIR
RECTANGULAR.','FIR          HAMMING.','FIR          HANNING.','FIR          BARTLETT.','FIR
BLACKMAN.','BUTTERWORTH.','CHEBYSHEV DIRECTO.','CHEBYSHEV INVERSO.','ELIPTICO.');
                    end

                    if (elec_f_d==1) % Filtro FIR Rectangular
                      load fir_rect.mat;
                    end

                    if (elec_f_d==2) % Filtro FIR Hamming
                      load fir_hamm.mat;
                    end

                    if (elec_f_d==3) % Filtro FIR Hanning
                      load fir_hann.mat;
                    end

                    if (elec_f_d==4) % Filtro FIR Bartlett
                      load fir_bart.mat;
                    end

                    if (elec_f_d==5) % Filtro FIR Blackman
                      load fir_blac.mat;
                    end

                    if (elec_f_d==6) % Filtro BUTTERWORTH
                      load butter.mat
                    end

                    if (elec_f_d==7) % Filtro CHEBYSHEV DIRECTO
                      load ched.mat
                    end

                    if (elec_f_d==8) % Filtro CHEBYSHEV INVERSO
                      load chei.mat
                    end

                    if (elec_f_d==9) % Filtro ELIPTICO.
                      load elip.mat
                    end
                re_fi_a(x_n,y_n,f_m);
                kiko=menu('  ','CONTINUAR');
                close
           end  %%%%  fin de respuesta global del filtro diseñado

           if (ele_fa==2) %%%% Filtro global Especificado anteriormente ideal %%%%
                 clc
                 disp(men_1);
                 load fil_ep.mat %leemos el filtro del disco
                 re_fi_a(x_n,y_n,f_m);
                 kiko=menu('    ','CONTINUAR');
                 close
           end    %%% Fin filtro digital equivalente del analógico %%%%
 else
    kiko=menu('Frecuencia muestreo **ERRONEA**','CONTINUAR');
```

```
 end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%          fin de filtros analógicos          %%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%*************************************************************************

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%          PROGRAMA DE SIMULACION CON SEÑALES DIGITALIZADAS          %%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if eleccion==4
  clc
  men_1=('      *** SIMULACION CON SEDALES DIGITALIZADAS *** ');
    tip_sen=-1
    while ((tip_sen<1)|(tip_sen>4));
       clc
       disp(men_1);
         tip_sen=menu('TIPO DE SEDAL A TRATAR:','SEDAL DE ECG.','SEDAL DE VOZ.','OTRA
SEDAL GUARDADA EN DISCO','ULTIMA SEDAL TRATADA.');
    end

    tip_fil=-1
    while ((tip_fil<1)|(tip_fil>10))
       clc
       disp(men_1);
            tip_fil=menu('FILTRO A UTILIZAR:','FIR RECTANGULAR.','FIR HAMMING','FIR
HANNING','FIR BARTLETT','FIR BLACKMAN','BUTTERWORTH.','CHEBYSHEV DIRECTO.','CHEBYSHEV
INVERSO.','ELIPTICO.','FILTRO ESPECIFICADO ANTERIORMENTE.');
    end
    f_muest=input('Entra frecuencia de muestreo (Hz): ');

 if (f_muest>0)

    disp('Espere leyendo en el disco ....');
       % Leemos la señal guardada en el disco
    if(tip_sen==1)
      load ecg.mat  %% SEDAL DE ECG
      senal=ecg;
      ecg=[];
    end
    if(tip_sen==2)
      load voz.mat  %% SEDAL DE VOZ
      senal=voz;
      voz=[];
    end
    if(tip_sen==3)
      load otra_s.mat %% OTRA SEÑAL A TRATAR
      senal=otra_s;
      otra_s=[];
    end
    if(tip_sen==4)
      load ult_sen.mat %% ULTIMA SEÑAL TRATADA
      senal=ult_sen;
      ult_sen=[];
    end

          %%% Leemos el filtro del disco

    if (tip_fil==1) % Filtro FIR Rectangular
       load fir_rect.mat;
    end
    if (tip_fil==2) % Filtro FIR Hamming
       load fir_hamm.mat;
    end
    if (tip_fil==3) % Filtro FIR Hanning
       load fir_hann.mat;
    end
    if (tip_fil==4) % Filtro FIR Bartlett
       load fir_bart.mat;
    end
    if (tip_fil==5) % Filtro FIR Blackman
       load fir_blac.mat;
    end
    if (tip_fil==6) % Filtro BUTTERWORTH
       load butter.mat
    end
```

```
    if (tip_fil==7) % Filtro CHEBYSHEV DIRECTO
       load ched.mat
    end
    if (tip_fil==8) % Filtro CHEBYSHEV INVERSO
       load chei.mat
    end
    if (tip_fil==9) % Filtro ELIPTICO.
       load elip.mat
    end
    if (tip_fil==10) % Filtro ESPECIFICADO ANTERIORMENTE
       load fil_ep.mat
    end


      %%% Una vez leidas del disco la señal y el filtro pasamos a simular
    ult_sen=re_sen(x_n,y_n,f_muest,senal); %%% Simulación
    save ult_sen.mat ult_sen  %guardamos ultima señal en el disco.
    ult_sen=[];
 else
    kiko=menu('Frecuencia muestreo **ERRONEA**','CONTINUAR');
 end

end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%          fin de simulación con señales digitalizadas          %%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% ************************************************************************

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%      PROGRAMA DE SIMULACIÓN DE LOS FILTROS EN EL TMDS3200051    %%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if eleccion==5
    men_1=('      *** SIMULACIËN DE LOS FILTROS CON EL DSP TMDS3200051 *** ');

    clc
    tip_fil=-1
    while ((tip_fil<1)|(tip_fil>10))
       clc
       disp(men_1);   %% Elegimos el tipo de filtro a simular
             tip_fil=menu('FILTRO A SIMULAR:','FIR RECTANGULAR.','FIR HAMMING','FIR
HANNING','FIR BARTLETT','FIR BLACKMAN','BUTTERWORTH.','CHEBYSHEV DIRECTO.','CHEBYSHEV
INVERSO.','ELIPTICO.','FILTRO ESPECIFICADO ANTERIORMENTE.');
    end


       %%% Leemos el filtro del disco
    if (tip_fil==1) % Filtro FIR Rectangular
       load fir_rect.mat;
    end
    if (tip_fil==2) % Filtro FIR Hamming
       load fir_hamm.mat;
    end
    if (tip_fil==3) % Filtro FIR Hanning
       load fir_hann.mat;
    end
    if (tip_fil==4) % Filtro FIR Bartlett
       load fir_bart.mat;
    end
    if (tip_fil==5) % Filtro FIR Blackman
       load fir_blac.mat;
    end
    if (tip_fil==6) % Filtro BUTTERWORTH
       load butter.mat
    end
    if (tip_fil==7) % Filtro CHEBYSHEV DIRECTO
       load ched.mat
    end
    if (tip_fil==8) % Filtro CHEBYSHEV INVERSO
       load chei.mat
    end
    if (tip_fil==9) % Filtro ELIPTICO.
       load elip.mat
    end
    if (tip_fil==10) % Filtro ESPECIFICADO ANTERIORMENTE
       load fil_ep.mat
    end
```

```
if ((tip_fil>5)&(tip_fil<10))
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  tendremos 3 realizaciones
        %%%%%     filtros IIR    %%%%  * Realización directa
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  * Realización en cascada
        x_n_c_c=[];                 %  * Realización en paralelo
        y_n_c_c=[];
        x_n_c_p=[];
        y_n_c_p=[];

        [x_n_c,y_n_c]=casc_iir(x_n,y_n);% Calculamos la realización
                                        % en CASCADA
        q=size(y_n_c);
          f=q(1);c=q(2);

           for i=1:f
                 for j=1:c
                    x(j)=x_n_c(i,j);
                    y(j)=y_n_c(i,j);
                 end
              [xc,yc]=cu_coef(x,y);
              x_n_c_c=[x_n_c_c;xc];
              y_n_c_c=[y_n_c_c;yc];
           end

        [x_n_p,y_n_p,cte]=parl_iir(x_n,y_n);%Calculamos la realización
        cte=cu_cte(cte);                    %en paralelo
        q=size(y_n_p);
          f=q(1);c=q(2);

           for i=1:f
                 for j=1:c
                    x(j)=x_n_p(i,j);
                    y(j)=y_n_p(i,j);
                 end
              [xc,yc]=cu_coef(x,y);
              x_n_c_p=[x_n_c_p;xc];
              y_n_c_p=[y_n_c_p;yc];
           end

        [x_n_d,y_n_d]=cu_coef(x_n,y_n);
        delta=[1 zeros(1,199)];
        h_i=filter(x_n,y_n,delta);
        delta=[4096 zeros(1,199)];
        h_d=filter(x_n_d,y_n_d,delta);
        h_d=round(h_d);
        h_d=h_d/4096;
        h_c=re_cas(x_n_c_c,y_n_c_c);
        h_p=re_par(x_n_c_p,y_n_c_p,cte);
        re_co_fi(h_i,h_d,h_c,h_p,0); % representacion grafica
        kiko=menu(' ','CONTINUAR');
        close

end %fin filtro IIR

if (tip_fil<6)

        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  tendremos 2 realizaciones
        %%%%%%%   filtros FIR  %%%%%%%%  * Realización directa
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  * Realización en cascada
        x_n_c_c=[];
        y_n_c_c=[];
        [x_n_c]=casc_fir(x_n); %% Calculamos la realización en
        y_n_c=y_n;             %% CASCADA
           p=size(x_n_c);
           f=p(1);c=p(2);
           for i=1:f
             for j=1:c
               x(j)=x_n_c(i,j);    %% Cuantificamos los coeficientes
              end                  %% de los sistemas en cascada
               [xc,yc]=cu_coef(x,y_n_c); %% obteniendo
               x_n_c_c=[x_n_c_c;xc];     %% x_n_c_c ; y_n_c_c
               y_n_c_c=[y_n_c_c;yc,0,0];
           end
          [x_n_d,y_n_d]=cu_coef(x_n,y_n);
                        %% Cuantificamos la realizacion directa

          delta=[1 zeros(1,199)];
          h_i=filter(x_n,y_n,delta);
```

```
                    delta=[4096 zeros(1,199)];
                    h_d=filter(x_n_d,y_n_d,delta);
                    h_d=round(h_d);
                    h_d=h_d/4096;
                    h_c=re_cas(x_n_c_c,y_n_c_c);
                    re_co_fi(h_i,h_d,h_c,h_c,1); % representacion grafica
                    kiko=menu(' ','CONTINUAR');
                    close

          end  %fin filtro FIR

          if (tip_fil==10)
                        %%%%%%%%%%%%%%%%%%%%%%%%
                        %% Filtro Especificado %%
                        %%%%%%%%%%%%%%%%%%%%%%%%
                    delta=[1 zeros(1,199)];
                    h_i=filter(x_n,y_n,delta);
                    delta=[4096 zeros(1,199)];
                    [x_n_d,y_n_d]=cu_coef(x_n,y_n);
                    h_d=filter(x_n_d,y_n_d,delta);
                    h_d=round(h_d);
                    h_d=h_d/4096;
                    re_co_fi(h_i,h_d,h_d,h_d,2) % reprresentacion grafica
                    kiko=menu(' ','CONTINUAR');
                    close

          end  %fin filtro especificado
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%      fin de simulación de los filtros en el TMDS3200051      %%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%*******************************************************************************

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%    PROGRAMA DE FILTROS ANALOGICOS DEL TMDS3200051    %%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%  Las variables x_n,y_n,n y tipo están en el disco

if (eleccion==6)
  men_1=('    *** FILTRO ANALËGICO GLOBAL EN EL TMDS3200051 *** ');
    ele_fa=-1;
      while ((ele_fa<1)|(ele_fa>2))
        clc
        disp(men_1);
         ele_fa=menu('TIPO DE AN-LISIS ANALËGICO CON EL TMDS3200051:','RESPUESTA GLOBAL DEL FILTRO DISEÑADO ANTERIORMENTE.','RESPUESTA GLOBAL DEL FILTRO ESPECIFICADO ANTERIORMENTE.');
      end

    clc
    f_mu=-1;
    f_m=-1;
      while ((f_m<1)|(f_m>11));
        clc
        disp(men_1);
            f_m=menu('Entra la FRECUENCIA de MUESTREO:','15 KHz','14 KHz','13 KHz','12 KHz','11 KHz','10 KHz','9 KHz','8 KHz','7 KHz','6 KHz','Otras...');
      end

    if (f_m==1)
       f_muest=15000;
    end
    if (f_m==2)
       f_muest=14000;
    end
    if (f_m==3)
       f_muest=13000;
    end
    if (f_m==4)
       f_muest=12000;
    end
    if (f_m==5)
       f_muest=11000;
    end
    if (f_m==6)
       f_muest=10000;
    end
```

```
    if (f_m==7)
       f_muest=9000;
    end
    if (f_m==8)
       f_muest=8000;
    end
    if (f_m==9)
       f_muest=7000;
    end
    if (f_m==10)
       f_muest=6000;
    end
    if (f_m==11)
       f_muest=1111;
    end

    if (f_muest==1111)
     clc
     f_mu=-1;
       while ((f_mu<1)|(f_mu>10));
         clc
         disp(men_1);
         f_mu=menu('Entra la FRECUENCIA de MUESTREO:','5 KHz','4 KHz','3 KHz','2 KHz','1
KHz','800 Hz','600 Hz','400 Hz','200 Hz','100 Hz');
       end
    end

    if (f_mu==1)
       f_muest=5000;
    end
    if (f_mu==2)
       f_muest=4000;
    end
    if (f_mu==3)
       f_muest=3000;
    end
    if (f_mu==4)
       f_muest=2000;
    end
    if (f_mu==5)
       f_muest=1000;
    end
    if (f_mu==6)
       f_muest=800;
    end
    if (f_mu==7)
       f_muest=600;
    end
    if (f_mu==8)
       f_muest=400;
    end
    if (f_mu==9)
       f_muest=200;
    end
    if (f_mu==10)
       f_muest=100;
    end

    elec_f_d=-1;

          if(ele_fa==1)  %%%%%%%  Respuesta global del filtro diseñado anteriormente
             disp(men_1);
                  elec_f_d=-1;
                   while((elec_f_d<1)|(elec_f_d>9))
                     clc
                     disp(men_1);
                       elec_f_d=menu('TIPO DE FILTRO A ANALIZAR DE LOS DISEÑADOS:','FIR
RECTANGULAR.','FIR          HAMMING.','FIR          HANNING.','FIR          BARTLETT.','FIR
BLACKMAN.','BUTTERWORTH.','CHEBYSHEV DIRECTO.','CHEBYSHEV INVERSO.','ELIPTICO.');
                   end

                   if (elec_f_d==1) % Filtro FIR Rectangular
                     load fir_rect.mat;
                   end

                   if (elec_f_d==2) % Filtro FIR Hamming
                     load fir_hamm.mat;
                   end
```

```
                if (elec_f_d==3) % Filtro FIR Hanning
                  load fir_hann.mat;
                end

                if (elec_f_d==4) % Filtro FIR Bartlett
                  load fir_bart.mat;
                end

                if (elec_f_d==5) % Filtro FIR Blackman
                  load fir_blac.mat;
                end

                if (elec_f_d==6) % Filtro BUTTERWORTH
                  load butter.mat
                end

                if (elec_f_d==7) % Filtro CHEBYSHEV DIRECTO
                  load ched.mat
                end

                if (elec_f_d==8) % Filtro CHEBYSHEV INVERSO
                  load chei.mat
                end

                if (elec_f_d==9) % Filtro ELIPTICO.
                  load elip.mat
                end
          end

          if (ele_fa==2) %%%% Filtro global Especificado anteriormente ideal %%%%
                clc
                disp(men_1);
                load fil_ep.mat %leemos el filtro del disco
          end


       if ((elec_f_d>0)&(elec_f_d<6)) % Elección estructura FIR
          estruc=-1
          while ((estruc<1)|(estruc>2));
            clc
            disp(men_1);
                        estruc=menu('Elige tipo de ESTRUCTURA FIR:','Realizaci³n
DIRECTA','Realizaci³n en CASCADA');
          end
       end

       if ((elec_f_d>5)&(elec_f_d<10)) % Elección estructura IIR
          estruc=-1
          while ((estruc<1)|(estruc>3));
            clc
            disp(men_1);
                        estruc=menu('Elige tipo de ESTRUCTURA IIR:','Realizaci³n
DIRECTA','Realizaci³n en CASCADA','Realizaci³n en PARALELO');
          end
       end

       clc
       disp(men_1);

           if (ele_fa==2) % Respuesta analógica del filtro Especificado
                delta=[4096 zeros(1,199)];
                [x_n_d,y_n_d]=cu_coef(x_n,y_n);
                h_d=filter(x_n_d,y_n_d,delta);
                h_d=round(h_d);
                h_d=h_d/4096;
                re_fi_an(h_d,f_muest);
                kiko=menu('  ','CONTINUAR');
                close;
           end

           if ((elec_f_d>0)&(elec_f_d<6)) % Respuesta analógica FIR
               if (estruc==1) % Para realización directa
                   [x_n_d,y_n_d]=cu_coef(x_n,y_n);
                   delta=[4096 zeros(1,199)];
                   h_d=filter(x_n_d,y_n_d,delta);
                   h_d=round(h_d);
                   h_d=h_d/4096;
```

```
            re_fi_an(h_d,f_muest);
            kiko=menu('  ','CONTINUAR');
            close
        end

        if (estruc==2) % Para realización en cascada
            x_n_c_c=[];
            y_n_c_c=[];
            [x_n_c]=casc_fir(x_n); %% Calculamos la realización en
            y_n_c=y_n;             %% CASCADA
               p=size(x_n_c);
               f=p(1);c=p(2);
               for i=1:f
                 for j=1:c
                   x(j)=x_n_c(i,j);    %% Cuantificamos los coeficientes
                 end                   %% de los sistemas en cascada
                  [xc,yc]=cu_coef(x,y_n_c); %% obteniendo
                  x_n_c_c=[x_n_c_c;xc];     %% x_n_c_c ; y_n_c_c
                  y_n_c_c=[y_n_c_c;yc,0,0];
               end
            h_c=re_cas(x_n_c_c,y_n_c_c);
            re_fi_an(h_c,f_muest);
            kiko=menu('  ','CONTINUAR');
            close;
        end

    end

    if ((elec_f_d>5)&(elec_f_d<10)) % Respuesta analógica IIR
        if (estruc==1) % Para realización directa
           [x_n_d,y_n_d]=cu_coef(x_n,y_n);
           delta=[4096 zeros(1,199)];
           h_d=filter(x_n_d,y_n_d,delta);
           h_d=round(h_d);
           h_d=h_d/4096;
           re_fi_an(h_d,f_muest);
           kiko=menu('  ','CONTINUAR');
           close
        end

        if (estruc==2) % Para realización en cascada
           x_n_c_c=[];
           y_n_c_c=[];
           [x_n_c,y_n_c]=casc_iir(x_n,y_n);% Calculamos la realización
                                           % en CASCADA
           q=size(y_n_c);
             f=q(1);c=q(2);

              for i=1:f
                    for j=1:c
                       x(j)=x_n_c(i,j);
                       y(j)=y_n_c(i,j);
                    end
                 [xc,yc]=cu_coef(x,y);
                 x_n_c_c=[x_n_c_c;xc];
                 y_n_c_c=[y_n_c_c;yc];
              end
              h_c=re_cas(x_n_c_c,y_n_c_c);
              re_fi_an(h_c,f_muest);
              kiko=menu('  ','CONTINUAR');
              close
        end

        if (estruc==3) % Para realización en paralelo
           x_n_c_p=[];
           y_n_c_p=[];
           [x_n_p,y_n_p,cte]=parl_iir(x_n,y_n);%Calculamos la realización
                                               %en paralelo
           q=size(y_n_p);
             f=q(1);c=q(2);

              for i=1:f
                    for j=1:c
                       x(j)=x_n_p(i,j);
                       y(j)=y_n_p(i,j);
                    end
                 [xc,yc]=cu_coef(x,y);
                 x_n_c_p=[x_n_c_p;xc];
```

```
                  y_n_c_p=[y_n_c_p;yc];
                end
              h_p=re_par(x_n_c_p,y_n_c_p,cte);
              re_fi_an(h_p,f_muest);
              kiko=menu('   ','CONTINUAR');
              close
            end

          end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%     fin de filtros analógicos en el TMDS3200051    %%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%***************************************************************************

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%  PROGRAMA DE SIMULACION CON SEÑALES DIGITALIZADAS EN EL TMDS3200051 %%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if eleccion==7
  clc
  men_1=('   *** SIMULACIËN CON SEÐALES DIGITALIZADAS EN EL TMDS3200051 *** ');
    tip_sen=-1;
    while ((tip_sen<1)|(tip_sen>4));
       clc
       disp(men_1);
         tip_sen=menu('TIPO DE SEÐAL A TRATAR:','SEÐAL DE ECG.','SEÐAL DE VOZ.','OTRA
SEÐAL GUARDADA EN DISCO','ULTIMA SEÐAL TRATADA.');
    end

    tip_fil=-1;
    while ((tip_fil<1)|(tip_fil>10))
       clc
       disp(men_1);
            tip_fil=menu('FILTRO A UTILIZAR:','FIR RECTANGULAR.','FIR HAMMING','FIR
HANNING','FIR BARTLETT','FIR BLACKMAN','BUTTERWORTH.','CHEBYSHEV DIRECTO.','CHEBYSHEV
INVERSO.','ELIPTICO.','FILTRO ESPECIFICADO ANTERIORMENTE.');
    end

    clc
    f_mu=-1;
    f_m=-1;
      while ((f_m<1)|(f_m>11));
        clc
        disp(men_1);
          f_m=menu('Entra la FRECUENCIA de MUESTREO:','15 KHz','14 KHz','13 KHz','12
KHz','11 KHz','10 KHz','9 KHz','8 KHz','7 KHz','6 KHz','Otras...');
      end

    if (f_m==1)
       f_muest=15000;
    end
    if (f_m==2)
       f_muest=14000;
    end
    if (f_m==3)
       f_muest=13000;
    end
    if (f_m==4)
       f_muest=12000;
    end
    if (f_m==5)
       f_muest=11000;
    end
    if (f_m==6)
       f_muest=10000;
    end
    if (f_m==7)
       f_muest=9000;
    end
    if (f_m==8)
       f_muest=8000;
    end
    if (f_m==9)
       f_muest=7000;
    end
    if (f_m==10)
       f_muest=6000;
    end
```

```
    if (f_m==11)
       f_muest=1111;
    end

    if (f_muest==1111)
     clc
     f_mu=-1;
       while ((f_mu<1)|(f_mu>10));
         clc
         disp(men_1);
         f_mu=menu('Entra la FRECUENCIA de MUESTREO:','5 KHz','4 KHz','3 KHz','2 KHz','1
KHz','800 Hz','600 Hz','400 Hz','200 Hz','100 Hz');
       end
    end

    if (f_mu==1)
       f_muest=5000;
    end
    if (f_mu==2)
       f_muest=4000;
    end
    if (f_mu==3)
       f_muest=3000;
    end
    if (f_mu==4)
       f_muest=2000;
    end
    if (f_mu==5)
       f_muest=1000;
    end
    if (f_mu==6)
       f_muest=800;
    end
    if (f_mu==7)
       f_muest=600;
    end
    if (f_mu==8)
       f_muest=400;
    end
    if (f_mu==9)
       f_muest=200;
    end
    if (f_mu==10)
       f_muest=100;
    end

       % Leemos la señal guardada en el disco
    if(tip_sen==1)
      load ecgt.mat  %% SEÑAL DE ECG
      senal=ecgt;
      ecgt=[];
    end
    if(tip_sen==2)
      load vozt.mat  %% SEÑAL DE VOZ
      senal=vozt;
      vozt=[];
    end
    if(tip_sen==3)
      load otra_st.mat %% OTRA SEÑAL A TRATAR
      senal=otra_st;
      otra_st=[];
    end
    if(tip_sen==4)
      load ult_sent.mat %% ULTIMA SEÑAL TRATADA
      senal=ult_sent;
      ult_sent=[];
    end

          %%% Leemos el filtro del disco

    if (tip_fil==1) % Filtro FIR Rectangular
       load fir_rect.mat;
    end
    if (tip_fil==2) % Filtro FIR Hamming
       load fir_hamm.mat;
    end
    if (tip_fil==3) % Filtro FIR Hanning
       load fir_hann.mat;
```

```
    end
    if (tip_fil==4) % Filtro FIR Bartlett
       load fir_bart.mat;
    end
    if (tip_fil==5) % Filtro FIR Blackman
       load fir_blac.mat;
    end
    if (tip_fil==6) % Filtro BUTTERWORTH
       load butter.mat
    end
    if (tip_fil==7) % Filtro CHEBYSHEV DIRECTO
       load ched.mat
    end
    if (tip_fil==8) % Filtro CHEBYSHEV INVERSO
       load chei.mat
    end
    if (tip_fil==9) % Filtro ELIPTICO.
       load elip.mat
    end
    if (tip_fil==10) % Filtro ESPECIFICADO ANTERIORMENTE
       load fil_ep.mat
    end
                                  %%%%%%%%%%%%%%%%%%%%%%%%%%
       if ((tip_fil>0)&(tip_fil<6)) % Elección estructura FIR
          estruc=-1                 %%%%%%%%%%%%%%%%%%%%%%%%%%
          while ((estruc<1)|(estruc>2));
            clc
            disp(men_1);
                        estruc=menu('Elige tipo de ESTRUCTURA FIR:','Realizaci³n
DIRECTA','Realizaci³n en CASCADA');
          end

          if (estruc==1) %Estructura FIR directa
             [x_n_d,y_n_d]=cu_coef(x_n,y_n);
             delta=[4096 zeros(1,199)];
             h_d=filter(x_n_d,y_n_d,delta);
             h_d=round(h_d);
             h_d=h_d/4096;
             ult_sent=re_se_d(h_d,f_muest,senal);
             save ult_sent.mat ult_sent;
             ult_sent=[];
          end

          if (estruc==2) %Estructura FIR cascada
               x_n_c_c=[];
               y_n_c_c=[];
               [x_n_c]=casc_fir(x_n); %% Calculamos la realización en
               y_n_c=y_n;             %% CASCADA
                  p=size(x_n_c);
                  f=p(1);c=p(2);
                  for i=1:f
                    for j=1:c
                      x(j)=x_n_c(i,j);    %% Cuantificamos los coeficientes
                     end                  %% de los sistemas en cascada
                      [xc,yc]=cu_coef(x,y_n_c); %% obteniendo
                      x_n_c_c=[x_n_c_c;xc];     %% x_n_c_c ; y_n_c_c
                      y_n_c_c=[y_n_c_c;yc,0,0];
                  end
               delta=[4096 zeros(1,199)];
               h_c=re_cas(x_n_c_c,y_n_c_c);
               ult_sent=re_se_d(h_c,f_muest,senal);
               save ult_sent.mat ult_sent;
               ult_sent=[];
           end
       end
                                  %%%%%%%%%%%%%%%%%%%%%%%%%%
       if ((tip_fil>5)&(tip_fil<10)) % Elección estructura IIR
          estruc=-1;                 %%%%%%%%%%%%%%%%%%%%%%%%%%
          while ((estruc<1)|(estruc>3));
            clc
            disp(men_1);
                        estruc=menu('Elige tipo de ESTRUCTURA IIR:','Realizaci³n
DIRECTA','Realizaci³n en CASCADA','Realizaci³n en PARALELO');
          end

          if (estruc==1) % Estructuta IIR directa
             [x_n_d,y_n_d]=cu_coef(x_n,y_n);
             delta=[4096 zeros(1,199)];
```

```
          h_d=filter(x_n_d,y_n_d,delta);
          h_d=round(h_d);
          h_d=h_d/4096;
          ult_sent=re_se_d(h_d,f_muest,senal);
          save ult_sent.mat ult_sent;
          ult_sent=[];
       end

       if (estruc==2) % Estructura IIR cascada
          x_n_c_c=[];
          y_n_c_c=[];
          [x_n_c,y_n_c]=casc_iir(x_n,y_n);% Calculamos la realización
                                          % en CASCADA
           q=size(y_n_c);
             f=q(1);c=q(2);

              for i=1:f
                    for j=1:c
                       x(j)=x_n_c(i,j);
                       y(j)=y_n_c(i,j);
                    end
                 [xc,yc]=cu_coef(x,y);
                 x_n_c_c=[x_n_c_c;xc];
                 y_n_c_c=[y_n_c_c;yc];
              end
          h_c=re_cas(x_n_c_c,y_n_c_c);
          ult_sent=re_se_d(h_c,f_muest,senal);
          save ult_sent.mat ult_sent;
          ult_sent=[];
        end

       if (estruc==3) % Estructura IIR paralela
          x_n_c_p=[];
          y_n_c_p=[];
          [x_n_p,y_n_p,cte]=parl_iir(x_n,y_n);%Calculamos la realización
          cte=cu_cte(cte);                     %en paralelo
          q=size(y_n_p);
            f=q(1);c=q(2);

             for i=1:f
                   for j=1:c
                      x(j)=x_n_p(i,j);
                      y(j)=y_n_p(i,j);
                   end
                [xc,yc]=cu_coef(x,y);
                x_n_c_p=[x_n_c_p;xc];
                y_n_c_p=[y_n_c_p;yc];
             end
          h_p=re_par(x_n_c_p,y_n_c_p,cte);
          ult_sent=re_se_d(h_p,f_muest,senal);
          save ult_sent.mat ult_sent;
          ult_sent=[];
       end
    end

    if (tip_fil==10)
       delta=[4096 zeros(1,199)];
       [x_n_d,y_n_d]=cu_coef(x_n,y_n);
       h_d=filter(x_n_d,y_n_d,delta);
       h_d=round(h_d);
       h_d=h_d/4096;
       ult_sent=re_se_d(h_d,f_muest,senal);
       save ult_sent.mat ult_sent;
       ult_sent=[];
    end


end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%    Fin de simulación con señales digitalizadas en el TMDS3200051     %%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%********************************************************************************

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%        PROGRAMACIÓN DEL DSP TMDS3200051          %%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if eleccion==8
```

```
   men_1=('            *** PROGRAMACIÈN DEL DSP TMDS3200051 *** ');

        apli=-1;
        while ((apli<1)|(apli>4));
          clc
          disp(men_1);
            apli=menu('Elige FUNCIÈN A PROGRAMAR:','CONVERSORES A/D Y D/A.','FILTRO FIR
DISEDADO.','FILTRO IIR DISEDADO.','FILTRO ESPECIFICADO.');
        end

   if (apli==1) %%% Programacion A/D y D/A

    clc
    f_mu=-1;
    f_m=-1;
      while ((f_m<1)|(f_m>11));
        clc
        disp(men_1);
           f_m=menu('Entra la FRECUENCIA de MUESTREO:','15 KHz','14 KHz','13 KHz','12
KHz','11 KHz','10 KHz','9 KHz','8 KHz','7 KHz','6 KHz','Otras...');
      end

    if (f_m==1)
       R_PRD=(' 01 h');A=(' 8');
       FI=(' 420 Hz');FS=(' 7500 Hz');FM=(' 15000 Hz');
    end
    if (f_m==2)
       R_PRD=(' 01 h');A=(' 9');
       FI=(' 400 Hz');FS=(' 7000 Hz');FM=(' 14000 Hz');
    end
    if (f_m==3)
       R_PRD=(' 01 h');A=(' 10');
       FI=(' 370 Hz');FS=(' 6500 Hz');FM=(' 13000 Hz');
    end
    if (f_m==4)
       R_PRD=(' 01 h');A=(' 10');
       FI=(' 340 Hz');FS=(' 6000 Hz');FM=(' 12000 Hz');
    end
    if (f_m==5)
       R_PRD=(' 01 h');A=(' 11');
       FI=(' 310 Hz');FS=(' 5500 Hz');FM=(' 11000 Hz');
    end
    if (f_m==6)
       R_PRD=(' 01 h');A=(' 12');
       FI=(' 280 Hz');FS=(' 5000 Hz');FM=(' 10000 Hz');
    end
    if (f_m==7)
       R_PRD=(' 01 h');A=(' 14');
       FI=(' 250 Hz');FS=(' 4500 Hz');FM=(' 9000 Hz');
    end
    if (f_m==8)
       R_PRD=(' 01 h');A=(' 16');
       FI=(' 230 Hz');FS=(' 4000 Hz');FM=(' 8000 Hz');
    end
    if (f_m==9)
       R_PRD=(' 01 h');A=(' 18');
       FI=(' 200 Hz');FS=(' 3500 Hz');FM=(' 7000 Hz');
    end
    if (f_m==10)
       R_PRD=(' 01 h');A=(' 21');
       FI=(' 170 Hz');FS=(' 3000 Hz');FM=(' 6000 Hz');
    end
    if (f_m==11)
       f_muest=1111;
    end

    if (f_muest==1111)
     clc
     f_mu=-1;
       while ((f_mu<1)|(f_mu>10));
         clc
         disp(men_1);
          f_mu=menu('Entra la FRECUENCIA de MUESTREO:','5 KHz','4 KHz','3 KHz','2 KHz','1
KHz','800 Hz','600 Hz','400 Hz','200 Hz','100 Hz');
       end
    end

    if (f_mu==1)
```

```
      R_PRD=(' 01 h');A=(' 25');
      FI=(' 140 Hz');FS=(' 2500 Hz');FM=(' 5000 Hz');
   end
   if (f_mu==2)
      R_PRD=(' 01 h');A=(' 31');
      FI=(' 115 Hz');FS=(' 2000 Hz');FM=(' 4000 Hz');
   end
   if (f_mu==3)
      R_PRD=(' 14 h');A=(' 4');
      FI=(' 85 Hz');FS=(' 1500 Hz');FM=(' 3000 Hz');
   end
   if (f_mu==4)
      R_PRD=(' 14 h');A=(' 6');
      FI=(' 55 Hz');FS=(' 1000 Hz');FM=(' 2000 Hz');
   end
   if (f_mu==5)
      R_PRD=(' 14 h');A=(' 12');
      FI=(' 30 Hz');FS=(' 500 Hz');FM=(' 1000 Hz');
   end
   if (f_mu==6)
      R_PRD=(' 14 h');A=(' 15');
      FI=(' 25 Hz');FS=(' 400 Hz');FM=(' 800 Hz');
   end
   if (f_mu==7)
      R_PRD=(' 14 h');A=(' 20');
      FI=(' 15 Hz');FS=(' 300 Hz');FM=(' 600 Hz');
   end
   if (f_mu==8)
      R_PRD=(' 14 h');A=(' 30');
      FI=(' 10 Hz');FS=(' 200 Hz');FM=(' 400 Hz');
   end
   if (f_mu==9)
      R_PRD=(' 50 h');A=(' 16');
      FI=(' 6 Hz');FS=(' 100 Hz');FM=(' 200 Hz');
   end
   if (f_mu==10)
      R_PRD=(' 50 h');A=(' 31');
      FI=(' 3 Hz');FS=(' 50 Hz');FM=(' 100 Hz');
   end


    clc
    if ((f_m>0)|(f_m<14))
      clc;
      disp(men_1);
      disp(' ');
      disp('PROGRAMACIËN CONVERSORES A/D y D/A');
      disp(' ');
      fprintf(1,'%s%s\n\n','* El valor de la frecuencia de muestreo vale:',FM);
      fprintf(1,'%s\n\n','* El valor del registro R_TCR vale: 20 h');
      fprintf(1,'%s%s\n\n','* El valor del registro R_PRD vale:',R_PRD);
      fprintf(1,'%s%s\n\n','* El valor de los registros TA y RA vale:',A);
      fprintf(1,'%s\n\n','* El valor de los registros TB y RB vale: 41');
      fprintf(1,'%s%s\n\n','* La frecuencia inferior de entrada vale:',FI);
      fprintf(1,'%s%s\n\n','* La frecuencia superior de entrada vale:',FS);
      disp(' ');
      kiko=menu('  ','CONTINUAR');

    end
  end  % fin de programación A/D y D/A


  if (apli==2) %%% Programación filtro FIR

   tip_fil=-1;
   while ((tip_fil<1)|(tip_fil>5))
      clc
      disp(men_1);
         tip_fil=menu('Filtro FIR a programar:','FIR RECTANGULAR.','FIR HAMMING','FIR
HANNING','FIR BARTLETT','FIR BLACKMAN');
   end

         %%% Leemos el filtro del disco

   if (tip_fil==1) % Filtro FIR Rectangular
      load fir_rect.mat;
   end
```

```
    if (tip_fil==2) % Filtro FIR Hamming
       load fir_hamm.mat;
    end
    if (tip_fil==3) % Filtro FIR Hanning
       load fir_hann.mat;
    end
    if (tip_fil==4) % Filtro FIR Bartlett
       load fir_bart.mat;
    end
    if (tip_fil==5) % Filtro FIR Blackman
       load fir_blac.mat;
    end
          estruc=-1
          while ((estruc<1)|(estruc>2));
            clc
            disp(men_1);
                          estruc=menu('Elige tipo de ESTRUCTURA FIR:','Realizaci³n
DIRECTA','Realizaci³n en CASCADA');
          end

          if (estruc==1) %Estructura FIR directa
             [x_n_d,y_n_d]=cu_coef(x_n,y_n);
             [xp,yp]=pro_coef(x_n_d,y_n_d);
             clc;
             disp(men_1);
             more on
             disp('Los coeficientes de x[n-k] para el DSP son:'),disp(xp)
             disp(' ');
             disp('Los coeficientes de y[n-k] para el DSP son:'),disp(yp)
             more off
             kiko=menu(' ','CONTINUAR');
          end

          if (estruc==2) %Estructura FIR cascada
               x_n_c_c=[];
               y_n_c_c=[];
               [x_n_c]=casc_fir(x_n); %% Calculamos la realización en
               y_n_c=y_n;             %% CASCADA
                  p=size(x_n_c);
                  f=p(1);c=p(2);
                  for i=1:f
                    for j=1:c
                      x(j)=x_n_c(i,j);    %% Cuantificamos los coeficientes
                    end                   %% de los sistemas en cascada
                      [xc,yc]=cu_coef(x,y_n_c); %% obteniendo
                      x_n_c_c=[x_n_c_c;xc];     %% x_n_c_c ; y_n_c_c
                      y_n_c_c=[y_n_c_c;yc,0,0];
                  end
                  clc;
                  disp(men_1);
                  for i=1:f
                    for j=1:c
                      xc(j)=x_n_c_c(i,j);
                      yc(j)=y_n_c_c(i,j);
                    end
                      [xp,yp]=pro_coef(xc,yc);
                      more on
                      fprintf(1,'%s%d\n','** Filtro N°:',i);
                      disp('Los coeficientes de x[n-k] para el DSP son:'),disp(xp)
                      disp('Los coeficientes de y[n-k] para el DSP son:'),disp(yp)
                      disp('   ');
                      kiko=menu('   ','Siguiente filtro');
                  end
                      more off
                      kiko=menu('  ','CONTINUAR');
          end
   end % fin de programacion FIR


   if (apli==3) %%% Programación filtro IIR

    tip_fil=-1;
    while ((tip_fil<1)|(tip_fil>4))
       clc
       disp(men_1);
                  tip_fil=menu('Filtro IIR a programar:','BUTTERWORTH','CHEBYSHEV
DIRECTO.','CHEBYSHEV INVERSO.','ELIPTICO.','FILTRO ESPECIFICADO ANTERIORMENTE.');
    end
```

```
          %%% Leemos el filtro del disco

    if (tip_fil==1) % Filtro BUTTERWORTH
       load butter.mat
    end
    if (tip_fil==2) % Filtro CHEBYSHEV DIRECTO
       load ched.mat
    end
    if (tip_fil==3) % Filtro CHEBYSHEV INVERSO
       load chei.mat
    end
    if (tip_fil==4) % Filtro ELIPTICO.
       load elip.mat
    end

          estruc=-1;
          while ((estruc<1)|(estruc>3));
            clc
            disp(men_1);
                          estruc=menu('Elige tipo de ESTRUCTURA IIR:','Realizaci%n
DIRECTA','Realizaci%n en CASCADA','Realizaci%n en PARALELO');
          end

          if (estruc==1) % Estructuta IIR directa
             [x_n_d,y_n_d]=cu_coef(x_n,y_n);
             [xp,yp]=pro_coef(x_n_d,y_n_d);
             clc;
             disp(men_1);
             more on
             disp('Los coeficientes de x[n-k] para el DSP son:'),disp(xp)
             disp(' ');
             disp('Los coeficientes de y[n-k] para el DSP son:'),disp(yp)
             more off
             kiko=menu(' ','CONTINUAR');
          end

          if (estruc==2) % Estructura IIR cascada
             x_n_c_c=[];
             y_n_c_c=[];
             [x_n_c,y_n_c]=casc_iir(x_n,y_n);% Calculamos la realización
                                             % en CASCADA
              q=size(y_n_c);
                f=q(1);c=q(2);

                 for i=1:f
                       for j=1:c
                          x(j)=x_n_c(i,j);
                          y(j)=y_n_c(i,j);
                       end
                    [xc,yc]=cu_coef(x,y);
                    x_n_c_c=[x_n_c_c;xc];
                    y_n_c_c=[y_n_c_c;yc];
                 end
                 clc;
                 disp(men_1);

                   for i=1:f
                     for j=1:c
                       xc(j)=x_n_c_c(i,j);
                       yc(j)=y_n_c_c(i,j);
                     end
                      [xp,yp]=pro_coef(xc,yc);
                      more on
                      fprintf(1,'%s%d\n','*Filtro N°:',i);
                      disp('Los coeficientes de x[n-k] para el DSP son:'),disp(xp)
                      disp('    ');
                      disp('Los coeficientes de y[n-k] para el DSP son:'),disp(yp)
                      kiko=menu('    ','Siguiente filtro.');
                   end
                   more off
                   kiko=menu('    ','CONTINUAR.');
           end

          if (estruc==3) % Estructura IIR paralela
             x_n_c_p=[];
             y_n_c_p=[];
             [x_n_p,y_n_p,cte]=parl_iir(x_n,y_n);%Calculamos la realización
             cte=cu_cte(cte);                    %en paralelo
```

```
            q=size(y_n_p);
              f=q(1);c=q(2);

              for i=1:f
                    for j=1:c
                       x(j)=x_n_p(i,j);
                       y(j)=y_n_p(i,j);
                    end
                 [xc,yc]=cu_coef(x,y);
                 x_n_c_p=[x_n_c_p;xc];
                 y_n_c_p=[y_n_c_p;yc];
              end
              clc;
              disp(men_1);
              disp('El valor de la constante en el DSP vale:'),disp(cte);

              for i=1:f
                 for j=1:c
                   xc(j)=x_n_c_p(i,j);
                   yc(j)=y_n_c_p(i,j);
                 end
                 [xp,yp]=pro_coef(xc,yc);
                 more on
                 fprintf(1,'%s%d\n','*Filtro Nº:',i);
                 disp('Los coeficientes de x[n-k] para el DSP son:'),disp(xp);
                 disp('   ');
                 disp('Los coeficientes de y[n-k] para el DSP son:'),disp(yp);
                 kiko=menu('    ','Siguiente filtro.');
              end
               more off
               kiko=menu('    ','CONTINUAR');
          end
   end

   if (apli==4)% Programacion filtro especificado
       load fil_ep.mat % Leemos filtro especificado del disco.
       [xc,yc]=cu_coef(x_n,y_n);
       [xp,yp]=pro_coef(xc,yc);
       clc;
       disp(men_1);
       more on
       disp('Los coeficientes de x[n-k] para el DSP son:'),disp(xp)
       disp(' ');
       disp('Los coeficientes de y[n-k] para el DSP son:'),disp(yp)
       more off
       kiko=menu(' ','CONTINUAR');
   end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%            Fin de programación del DSP TMDS3200051          %%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%****************************************************************************
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%            AYUDA PARA USAR EL PROGRAMA          %%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if eleccion==9
 ayuda;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%            Fin de ayuda para usar el programa               %%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

   if eleccion==10 %%%%%%  FIN DEL PROGRAMA   %%%%%%%%%

       salir=1;

   end

end  %FIN DE BUCLE DEL PROGRAMA PRINCIPAL

clear % borramos las variable de memoria.
clc;  % borramos la pantalla de texto.
clg;  % borramos la pantalla de grßficos
close ; % cerramos todas las ventanas
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

LISTADO DE CASC_FIR.M

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%    Función que realiza la función de transferencia del filtro FIR    %%%%
%%%    en la realización en cascada                                       %%%%
%%%   parametro de entrada:                                               %%%%
%%%       Coeficientes del filtro x_n                                     %%%%
%%%   paramatro de salida:                                                %%%%
%%%       matrices de realización en cascada x_n_c                        %%%%
%%%     cada fila de esta matriz contiene los coeficientes de los         %%%%
%%%     sistemas en cascada                                               %%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%                  function [x_n_c]=casc_fir(x_n)                       %%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                  function [x_n_c]=casc_fir(x_n)
x_n=rot90(x_n);
x_n=rot90(x_n); %% Ponemos la función en potencias crecientes

rai_num=roots(x_n); %%% Calculamos los ceros de la función

cte_num=x_n(1); % Calculamos el factor de escala que desaparece al calcular
                % las raices de los polinomios.

con_num=length(rai_num);

x_n_c=[];

j=1;
    while (j<=con_num)
        if (imag(rai_num(j))==0)
          term1=[0,poly([rai_num(j)])];
          x_n_c=[x_n_c;term1]; %% descomponemos en términos el numerador
          j=j+1;               %% cunado no tiene raices complejas
        else                   %% produce términos de orden 1
          x_n_c=[x_n_c;poly([rai_num(j),rai_num(j+1)])];
          j=j+2;               %% descomponemos en términos el numerador
        end                    %% cuando tiene raices complejas
    end                        %% produce términos de orden 2

lon_num=size(x_n_c);
col_num=lon_num(2);

for i=1:col_num
    x_n_c(1,i)=x_n_c(1,i)*cte_num;  %% Introducimos la cte del numerador
end                                 %% que se perdió al hallar las raices

x_n_c=rot90(x_n_c);
x_n_c=rot90(x_n_c); %% ponemos en potencias decrecientes
                    %% la matriz de coeficientes de x[n] en cascada
```

LISTADO DE CASC_IIR.M

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%    Función que realiza la función de transferencia del filtro  IIR   %%%%
%%%    en la realización en cascada                                       %%%%
%%%   parametro de entrada:                                               %%%%
%%%       Coeficientes del filtro x_n,y_n                                 %%%%
%%%   paramatro de salida:                                                %%%%
%%%       matrices de realización en cascada x_n_c,y_n_c                  %%%%
%%%     cada fila de esta matriz contiene los coeficientes de los         %%%%
%%%     sistemas en cascada                                               %%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%                  function [x_n_c,y_n_c]=casc_iir(x_n,y_n)             %%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                  function [x_n_c,y_n_c]=casc_iir(x_n,y_n)
x_n=rot90(x_n);
x_n=rot90(x_n);
y_n=rot90(y_n); %% Ponemos la función en potencias crecientes
y_n=rot90(y_n);

rai_num=roots(x_n);
rai_den=roots(y_n); %%% Calculamos los polos y los ceros de la función

cte_num=x_n(1); % Calculamos el factor de escala que desaparece al calcular
```

```
cte_den=y_n(1); % las raices de los polinomios.

con_num=length(rai_num);
con_den=length(rai_den);

x_n_c=[];
y_n_c=[];
j=1;k=1;
    while (j<=con_num)
        if (imag(rai_num(j))==0)
          term1=[0,poly([rai_num(j)])];
          x_n_c=[x_n_c;term1]; %% descomponemos en términos el numerador
          j=j+1;               %% cunado no tiene raices complejas
        else                   %% produce términos de orden 1
          x_n_c=[x_n_c;poly([rai_num(j),rai_num(j+1)])];
          j=j+2;              %% descomponemos en términos el numerador
        end                   %% cuando tiene raices complejas
    end                       %% produce términos de orden 2

    while (k<=con_den)
        if (imag(rai_den(k))==0)
          term2=[0,poly([rai_den(k)])];
          y_n_c=[y_n_c;term2]; %% descomponemos en términos el denominador
          k=k+1;               %% cuando no tiene raices complejas
        else                   %% produce términos de orden 1
          y_n_c=[y_n_c;poly([rai_den(k),rai_den(k+1)])];
          k=k+2;               %% descomponemos en términos el denominador
        end                    %% cuando tiene raices complejas
    end                        %% produce términos de orden 2

lon_num=size(x_n_c);
lon_den=size(y_n_c);
col_num=lon_num(2);
col_den=lon_den(2);

for i=1:col_num
    x_n_c(1,i)=x_n_c(1,i)*cte_num;  %% Introducimos la cte del numerador
end                                 %% que se perdió al hallar las raices

for i=1:col_den
    y_n_c(1,i)=y_n_c(1,i)*cte_den;  %% Introducimos la cte del denominador
end                                 %% que se perdió al hallar las raices


x_n_c=rot90(x_n_c);
x_n_c=rot90(x_n_c); %% ponemos en potencias decrecientes
                    %% la matriz de coeficientes de x[n] en cascada


y_n_c=rot90(y_n_c);
y_n_c=rot90(y_n_c); %% ponemos en potencias decrecientes
                    %% la matriz de coeficientes de y[n] en cascada
```

LISTADO DE COEF_ENT.M

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%    Función que devuelve los coeficientes del filtro tras pedirlos    %%%%
%%%    dependiendo del parámetro de entrada:                              %%%%
%%%    opcion: parametro de entrada                                       %%%%
%%%       1.- Ecuación en diferencias                                     %%%%
%%%       2.- Función de transferencia H(Z)                               %%%%
%%%       3.- diagrama de polos y ceros.                                  %%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%               [x_n,y_n]=coef_ent(opcion)                             %%%%
%%%               x_n=coeficientes de x[n]                               %%%%
%%%               y_n=coeficientes de y[n]                               %%%%
%%%               opcion=parámetro entrada a la función                  %%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                  function [x_n,y_n]=coef_ent(opcion)

%%% Suponemos que la opción de entrada solo es 1,2 ó 3.
%%%%%%%%%%%%%%%      opción (1) ecuación en diferencias        %%%%%%%%%%%%%%%
  % pide los coeficientes de la ecuación en diferencias
  if opcion==1
```

```
      disp(sprintf('\n\n\t ECUACION EN DIFERENCIAS '));
      x_n=input('Entra Vector Coeficientes [x(n),x(n-1),..,x(n-N)]: ');
      y_n=input('Entra Vector Coeficientes [y(n),y(n-1),..,y(n-M)]: ');
  end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%% opción (2) Funcion de transferencia H(Z)  %%%%%%%%%%%%%%%%%%%%
  % pide los coeficientes de la función H(z)
  if opcion==2
      disp(sprintf('\n\n\t FUNCION DE TRANSFERENCIA H(Z)=N(Z)/D(Z)'));
      x_n=input('Entra Vector Coef. Numerador [a0,a1*z-1,..,aN*z-N]: ');
      y_n=input('Entra Vector Coef. Denominador [b0,b1*z-1,..,aM*z-M]: ');
  end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%       opción (3) diagrama polos-ceros     %%%%%%%%%%%%%%%%%%%%
  % pide el lugar geomÚtrico de los polos y de los ceros
  if opcion==3
      disp(sprintf('\n\n\t DIAGRAMA POLOS-CEROS'));
      op_cord=-1;
      while ((op_cord<1)|(op_cord>2))
        op_cord=menu('Elige tipo de coordenadas','Rectangulares.','Polares.');
      end
          if op_cord==1  %%%%% Entrada en coordenadas rectangulares
            disp(sprintf('\n\n\t ai=real , bi=imaginario  '));
            polos=input('Entra polos[a1+i*b1;..;aN+i*bN]: ');
            ceros=input('Entra ceros[a1+i*b1;..;aM+i*bM]: ');
            polos=conj(polos');
            ceros=conj(ceros');
          end

          if op_cord==2  %%%%% Entra en coordenadas polares
             disp(sprintf('\n\n\t mi=modulo , fi=fase(grados)  '));
             polos_p=input('Entra polos[m1,f1;..;mN,fN]: ');
             ceros_p=input('Entra ceros[m1,f1;..;mM,fM]: ');

              %%%%% Cambiamos a coordenadas rectangulares
              dimension=size(polos_p);
              filas=dimension(1);
              for j=1:filas
                 fase=(-1*polos_p(j,2)*pi)/180; %pasa grados a radianes
                 polos(j)=(polos_p(j,1)*cos(fase))+i*(polos_p(j,1)*sin(fase));
              end

              dimension=size(ceros_p);
              filas=dimension(1);
              for j=1:filas;
                 fase=(-1*ceros_p(j,2)*pi)/180; % pasa grados a radianes
                 ceros(j)=(ceros_p(j,1)*cos(fase))+i*(ceros_p(j,1)*sin(fase));
              end

              polos=polos';
              ceros=ceros';
          end
                                %%% Calcula los coeficientes del filtro
           y_n=poly(polos);
           x_n=poly(ceros);

      ultimo_y=length(y_n);
      ultimo_x=length(x_n);      %%%%  Corrije los grados
                                 %%%%  del polinomio generado
    if y_n(ultimo_y)==0          %%%%  con la función poly.
       x_n=[0 x_n];
      y_n=y_n(1:ultimo_y-1);
    end

    if x_n(ultimo_x)==0
       y_n=[0 y_n];
       x_n=x_n(1:ultimo_x-1);
    end

    delta=[1 zeros(size(1:199))];
    h=filter(x_n,y_n,delta);        %%% Hacemos que una especificaci*nen polos
    factor=max(abs(fft(h)));    %%% y ceros tenga ganancia unidad
    y_n=y_n*factor;
 end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

LISTADO DE CON_AD.M

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%   CONVIERTE LAS FRECUENCIAS ANALOGICAS A FRECUENCIAS DIGITALES NORM.    %%
%%                          Y VICEVERSA                                    %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%   Las frecuencias normalizadas de salida son 1= PI rad. que corresponde %
%%   a la frecuencia de muestreo medios (fm/2)                              %
%%       los parámetros de entrada son:                                     %
%%         * f_muest     :frecuencia de muestreo en (Hz).                   %
%%         * f_ent       :frecuencia entrada.                               %
%%         * ad_da       :1 conversión A/D , 0 conversión D/A               %
%%       los parámetros de salida son:                                      %
%%         * frecuencia :frecuencia convertida.                             %
%%      function  frecuencia=con_ad(f_muest,f_ent,ad_da)                    %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

          function  frecuencia=con_ad(f_muest,f_ent,ad_da)

%%%%  conversión A/D

if (ad_da==1)

        frecuencia=((2*f_ent)/f_muest)*pi;

%%%% conversión D/A

else

        frecuencia=(f_ent*(f_muest/2))*pi;

end
```

LISTADO DE CU_COEF.M

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%  FUNCION QUE CUANTIFICA LOS COEFICIENTES DE UN FILTRO Y NOS DA LOS  %%%
%%%          COEFICIENTES QUE TENDRA EN EL DSP TMDS3200051               %%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%  Los parámetros de entrada son:                                      %%%
%%%       x_n,y_n  : coeficientes del filtro ideal                       %%%
%%%  Parámetros de salida:                                               %%%
%%%       x_n_c,y_n_c : coeficientes en el filtro real del DSP           %%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%                function [x_n_c,y_n_c]=cu_coef(x_n,y_n)              %%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

                function [x_n_c,y_n_c]=cu_coef(x_n,y_n)
x_ba=abs(x_n);
y_ba=abs(y_n);            %%% Normalizamos los coeficientes
x_max=max(x_ba);
y_max=max(y_ba);

   if x_max > y_max
        max_coef=x_max;
   else
        max_coef=y_max;
   end

   if ((max_coef<1)&(max_coef>0))
        max_coef=1;
   end

if (max_coef==0)
     x_n_c=[0,0,0];
     y_n_c=[0];
else
     x_n_c_c=x_n/max_coef;
     y_n_c_c=y_n/max_coef;

                        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

                        %%%%  Cuantificamos los coeficientes    %%%%%%%%

for i=1:length(x_n_c_c)
 valor=x_n_c_c(i);
```

```
   if (valor>0)
     x_n_c_s(i)=0;
   else                %%% Guardamos un vector de signos x_n_c_s para los
     x_n_c_s(i)=1;     %%% coeficientes x_n_c_c.
   end
end
x_n_c_c=abs(x_n_c_c); %%% Tomamos de x_n_c_c el valor absoluto

for i=1:length(y_n_c_c)
 valor=y_n_c_c(i);
   if (valor>0)
     y_n_c_s(i)=0;
   else                %%% Guardamos un vector de signos y_n_c_s para los
     y_n_c_s(i)=1;     %%% coeficientes y_n_c_c
   end
end
y_n_c_c=abs(y_n_c_c); %%% Tomamos de y_n_c_c el valor absoluto

     for i=1:length(x_n_c_c) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
         j=1;                %%% Cuantificamos x_n_c_c obteniendo x_n_c %%%%
         den=1/2;            %%% en valor absoluto.                     %%%%
         k2=1;          %%% Más tarde colocamos el signo correspondiente %%%
       while (j<13)  %% 14 bits ( 1 signo 13 valor)
           k1=k2;
           k2=den;
           p=x_n_c_c(i);
         if (k1>p)&(p>=k2)
           di_1=abs(p-k1);
           di_2=abs(p-k2);
              if (di_1>di_2)
                 x_n_c(i)=k2;
              else
                 x_n_c(i)=k1;
              end
         end
           den=den/2;
           j=j+1;
       end
         if x_n_c_c(i)==1
           x_n_c(i)=1;
         end
         if x_n_c_c(i)<(1/4096)
           x_n_c(i)=0;
         end
     end

    for j=1:length(x_n_c)
       valor=x_n_c_s(j);          %% ponemos el signo correspondiente
          if (valor==1);
            x_n_c(j)=x_n_c(j)*(-1);
          end
    end                                 %%% fin para x_n_c %%%
                              %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


     for i=1:length(y_n_c_c) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
         j=1;                %%% Cuantificamos y_n_c_c obteniendo y_n_c %%%%
         den=1/2;            %%% en valor absoluto.                     %%%%
         k2=1;          %%% Más tarde colocamos el signo correspondiente %%%
       while (j<13) %%% 14 bit del cuantificador ( 1 signo 13 valor)
           k1=k2;
           k2=den;
           p=y_n_c_c(i);
         if (k1>p)&(p>=k2)
           di_1=abs(p-k1);
           di_2=abs(p-k2);
              if (di_1>di_2)
                 y_n_c(i)=k2;
              else
                 y_n_c(i)=k1;
              end
         end
           den=den/2;
           j=j+1;
       end
         if y_n_c_c(i)==1
           y_n_c(i)=1;
         end
```

```
        if y_n_c_c(i)<(1/4096)
           y_n_c(i)=0;
        end
    end

   for j=1:length(y_n_c)
      valor=y_n_c_s(j);       %% ponemos el signo correspondiente
         if (valor==1);
           y_n_c(j)=y_n_c(j)*(-1);
         end                         %% fin para y_n_c %%%
   end                     %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
end
```

LISTADO DE CU_CTE.M

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%         Función que cuantifica las constantes para el DSP         %%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%  Parámetro entrada : cte                                            %%%%
%%%  Parámetro salida  : cte_c                                          %%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%   function  cte_c=cu_cte(cte)    %%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                     function  cte_c=cu_cte(cte)

         if (cte>0)
           c_s=1;
         else
           c_s=0;
         end
         cte=abs(cte);

         j=1;                    %%% Cuantificamos cte obteniendo cte_c
         den=1/2;
         k2=1;
      while (j<13)   %% 14 bits ( 1 signo 13 valor)
           k1=k2;
           k2=den;
           p=cte;
          if (k1>p)&(p>=k2)
              di_1=abs(p-k1);
              di_2=abs(p-k2);
                 if (di_1>di_2)
                    cte_c=k2;
                 else
                    cte_c=k1;
                 end
          end
              den=den/2;
              j=j+1;
      end
           if (cte>=1)
              cte_c=round(cte);
           end
           if cte<(1/4096)
              cte_c=0;
           end

     if c_s==0
       cte_c=cte_c*(-1);
     end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

LISTADO DE CUAN_SEN.M

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%    Función que devuelve los cuantifica una señal con una precisión  %%%%
%%%    de 14 bits igual que el DSP                                      %%%%
%%%   parametro de entrada:                                             %%%%
%%%       Señal sin cuantificar sen_sc                                  %%%%
%%%   paramatro de salida:                                              %%%%
%%%       Señal cuantificada con 14 bits de precisión  sen_c            %%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%              function sen_c=cuan_sen(sen_sc);                       %%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
                function sen_c=cuan_sen(sen_sc);
x_max=max(sen_sc);  % normalizamos a 2 elevado (14-1)
x_min=min(sen_sc);
    if (abs(x_max)>abs(x_min))
        factor=8191/abs(x_max);
    else
        factor=8191/abs(x_min);
    end

sen_sc=sen_sc*factor;

                %cuantificamos el valor de las muestras
sen_c=round(sen_sc);
```

LISTADO DE ELIP_ES.M

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%  FUNCION QUE CALCULA LOS COEFICIENTES DE UN FILTRO ELIPTICO          %%%
%%                  Rizado en ambas bandas                               %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%      Se dan especificacioenes del filtro y luego da              %%%%%
%%%     como salida los coeficientes del filtro y el orden de este   %%%%%
%%%  parámetros de entrada:                                          %%%%%
%%%      w_paso :frecuencia de la banda de paso (0-1) 1=fm/2         %%%%%
%%%      w_aten :frecuencia de la banda atenuada(0-1) 1=fm/2         %%%%%
%%%      a_paso :atenuacion en la banda de paso                      %%%%%
%%%      a_aten :atenuacion en la banda atenuada                     %%%%%
%%%      tipo   :tipo de filtro 1 FLP, 2 FHP, 3 FBP ,4 FRB           %%%%%
%%% w_paso y w_aten dos valores [w1 w2] para paso banda o rehazo b.  %%%%%
%%%  parámetros de salida:                                           %%%%%
%%%      orden  :orden del filtro calculado                          %%%%%
%%%      x_n,y_n :Coeficientes del filtro                            %%%%%
%% function  [x_n,y_n,n]=elip_es(w_paso,w_aten,a_paso,a_aten,tipo)    %%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    function [x_n,y_n,n]=elip_es(w_paso,w_aten,a_paso,a_aten,tipo)

[n,wn]=ellipord(w_paso,w_aten,a_paso,a_aten); %calculamos el orden del filtro

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
      %%%  cálculo de un Elíptico PASO BAJO

if tipo==1

   [x_n,y_n]=ellip(n,a_paso,a_aten,wn);

end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
      %%%  cálculo de un Elíptico PASO ALTO

if tipo==2

   [x_n,y_n]=ellip(n,a_paso,a_aten,wn,'high');

end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
      %%%  cálculo de un Elíptico PASO BANDA

if tipo==3

   [x_n,y_n]=ellip(n,a_paso,a_aten,wn);

end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
      %%%  cálculo de un Elíptico RECHAZO BANDA

if tipo==4

   [x_n,y_n]=ellip(n,a_paso,a_aten,wn,'stop');

end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

LISTADO DE ESTAB_F.M

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%        FUNCION ENCARGA DE ESTUDIAR LA ESTABILIDAD DEL FILTRO     %%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%     Toma como parámetros de entrada los coeficientes del filtro  %%%%%%%%
%%%     un indicador para decir si se quiere representar el diagrama %%%%%%%%
%%%     de polos y ceros y como salida devuelve otro indicador para  %%%%%%%%
%%%     decirnos si el filtro es o no estable                        %%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%  PARAMETROS DE ENTRADA:                                          %%%%%%%%
%%%   x_n,y_n:Coeficientes del filtro                                 %%%%%%%%
%%%   dibujar:1 para dibujar,0 para no dibujar.                       %%%%%%%%
%%%  PARAMETROS DE SALIDA:                                           %%%%%%%%
%%%   estable:1 filtro estable, 0 filtro inestable.                   %%%%%%%%
%%%          function estable=estabili(x_n,y_n,dibujar)               %%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
            function estable=estabili(x_n,y_n,dibujar)

    if (length(x_n)>length(y_n))
       y_n=[y_n zeros(1,length(x_n)-length(y_n))];   % Obtenemos una función
    end                                              % en potencia de zn a partir
                                                     % de una de z-n
    if (length(y_n)>length(x_n))
       x_n=[x_n zeros(1,length(y_n)-length(x_n))];
    end


    if length(x_n)==1       % cuando tengamos un polo y un cero
         ceros=[0];
         polos=[0];
    else
        ceros=roots(x_n);   % calculamos los polos y los ceros
        polos=roots(y_n);
    end

 tetap=angle(polos);rhop=abs(polos); % polos y ceros en forma polar
 tetac=angle(ceros);rhoc=abs(ceros);

  estable=1;

    %% comprobamos los polos que esten fuera del ciculo unidad
    %% o en su frontera (supondremos filtros causales)

  m_p=rhop;
  m_c=rhoc;

    for j=1:length(rhop)
        if (rhop(j)>=1)
          for k=1:length(rhoc)
            if (round(m_p(j))==round(m_c(k)))&(round(tetap(j))==round(tetac(k)))
                  m_p(j)=-1; m_c(k)=-10; %valores para eliminar polo y cero
            end
          end
        end
    end

    for j=1:length(rhop)
         if(m_p(j)>=1)
            estable=0;
            j=length(rhop);
         end
    end

    if (dibujar==1) % si la opción dibujar =1 dibuja diagrama polos-ceros
        t=0:1:100;                          % circulo unidad H(jÛ)
        circ_unidad=exp(i*((2*pi)/50)*t);
        tetac_u=angle(circ_unidad);rhoc_u=abs(circ_unidad);
        circ_un_t=1.1*exp(i*((2*pi)/50)*t);
        tetac_t=angle(circ_un_t);rhoc_t=abs(circ_un_t);

        clg
        subplot(111);
        polar(tetac_t,rhoc_t,'i',tetac,rhoc,'ow',tetap,rhop,'xg',tetac_u,rhoc_u,'.g');
        title('POLOS (x) Y CEROS (O) DE H(Z)');

          if (estable==1)
              xlabel('filtro  ESTABLE ');
          else
              xlabel('Ojo  ***** filtro INESTABLE *******');
          end

    end
```

LISTADO DE F_BUT_ES.M

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%  FUNCION QUE CALCULA LOS COEFICIENTES DE UN FILTRO BUTTERWORTH      %%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%      Se dan especificacioenes del filtro y luego da                  %%%%%
%%%     como salida los coeficientes del filtro y el orden de este      %%%%%
%%%  parámetros de entrada:                                              %%%%%
%%%      w_paso :frecuencia de la banda de paso (0-1) 1=fm/2             %%%%%
%%%      w_aten :frecuencia de la banda atenuada(0-1) 1=fm/2             %%%%%
%%%      a_paso :atenuacion en la banda de paso                          %%%%%
%%%      a_aten :atenuacion en la banda atenuada                         %%%%%
%%%      tipo   :tipo de filtro 1 FLP, 2 FHP, 3 FBP ,4 FRB               %%%%%
%%% w_paso y w_aten dos valores [w1 w2] para paso banda o rehazo b.      %%%%%
%%%  parámetros de salida:                                               %%%%%
%%%      orden  :orden del filtro calculado                              %%%%%
%%%      x_n,y_n :Coeficientes del filtro                                %%%%%
%%% function  [x_n,y_n,n]=f_but_es(w_paso,w_aten,a_paso,a_aten,tipo)    %%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    function [x_n,y_n,n]=f_but_es(w_paso,w_aten,a_paso,a_aten,tipo)

[n,wn]=buttord(w_paso,w_aten,a_paso,a_aten); %calculamos el orden del filtro

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
      %%%  cálculo de un Butterworth PASO BAJO

if tipo==1

   [x_n,y_n]=butter(n,wn);

end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
      %%%  cálculo de un Butterworth PASO ALTO

if tipo==2

   [x_n,y_n]=butter(n,wn,'high');

end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
      %%%  cálculo de un Butterworth PASO BANDA

if tipo==3

   [x_n,y_n]=butter(n,wn);

end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
      %%%  cálculo de un Butterworth RECHAZO BANDA

if tipo==4

   [x_n,y_n]=butter(n,wn,'stop');

end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

LISTADO DE F_FIR.M

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%       FUNCION QUE CALCULA LOS COEFICIENTES DE UN FILTRO FIR         %%%%%
%%%%                  MEDIANTE EL METODO DE LAS VENTANAS                 %%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Se da el orden del filtro la ventana y las frecuencias de corte     %%%%%
%%%     como salida los coeficientes del filtro                         %%%%%
%%%  parámetros de entrada:                                              %%%%%
%%%   w_paso :frecuencia de la banda de paso (0-1) 1=fm/2                %%%%%
%%%   tipo   :tipo de filtro 1 FLP, 2 FHP, 3 FBP ,4 FRB                  %%%%%
%%% win : ventana 1 Rectan.,2 Hamming,3 Hanning ,4 Bartlett, 5 Blackman %%%%
%%%   n      :orden del filtro                                           %%%%%
%%%       w_paso dos valores [w1 w2] para paso banda o rehazo banda     %%%%%
%%%  parámetros de salida:                                               %%%%%
%%%      x_n,y_n :Coeficientes del filtro                                %%%%%
%%% function  [x_n,y_n]=f_fir1(w_paso,n,tipo,win)                       %%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    function [x_n,y_n]=f_fir1(w_paso,n,tipo,win)

y_n=[1];
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
      %%%  cálculo de un FIR PASO BAJO

if tipo==1

   if win==1     % Ventana Rectangular
     x_n = fir1(n,w_paso,boxcar(n+1));
   end

   if win==2     % Ventana Hamming
     x_n =fir1(n,w_paso); % por defecto ventana Hamming
   end

   if win==3     % Ventana Hanning
     x_n =fir1(n,w_paso,hanning(n+1));
   end

   if win==4     % Ventana Bartlett
     x_n =fir1(n,w_paso,bartlett(n+1));
   end

   if win==5     % Ventana Blackman
     x_n =fir1(n,w_paso,blackman(n+1));
   end

end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
      %%%  cálculo de un FIR PASO ALTO

if tipo==2

   if win==1     % Ventana Rectangular
     x_n = fir1(n,w_paso,'high',boxcar(n+1));
   end

   if win==2     % Ventana Hamming
     x_n =fir1(n,w_paso,'high'); % por defecto ventana Hamming
   end

   if win==3     % Ventana Hanning
     x_n =fir1(n,w_paso,'high',hanning(n+1));
   end

   if win==4     % Ventana Bartlett
     x_n =fir1(n,w_paso,'high',bartlett(n+1));
   end

   if win==5     % Ventana Blackman
     x_n =fir1(n,w_paso,'high',blackman(n+1));
   end

end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
      %%%  cálculo de un FIR PASO BANDA

if tipo==3

   if win==1     % Ventana Rectangular
     x_n = fir1(n,w_paso,boxcar(n+1));
   end

   if win==2     % Ventana Hamming
     x_n =fir1(n,w_paso); % por defecto ventana Hamming
   end

   if win==3     % Ventana Hanning
     x_n =fir1(n,w_paso,hanning(n+1));
   end

   if win==4     % Ventana Bartlett
     x_n =fir1(n,w_paso,bartlett(n+1));
   end

   if win==5     % Ventana Blackman
     x_n =fir1(n,w_paso,blackman(n+1));
   end
```

```
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
      %%%  cálculo de un FIR RECHAZO BANDA

if tipo==4

   if win==1    % Ventana Rectangular
     x_n = fir1(n,w_paso,'stop',boxcar(n+1));
   end

   if win==2    % Ventana Hamming
     x_n =fir1(n,w_paso,'stop'); % por defecto ventana Hamming
   end

   if win==3    % Ventana Hanning
     x_n =fir1(n,w_paso,'stop',hanning(n+1));
   end

   if win==4    % Ventana Bartlett
     x_n =fir1(n,w_paso,'stop',bartlett(n+1));
   end

   if win==5    % Ventana Blackman
     x_n =fir1(n,w_paso,'stop',blackman(n+1));
   end

end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

LISTADO DE MENU_FIR.M

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%              FUNCION QUE PIDE LOS PARAMETROS DEL FIR                  %%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%    Da como salida el orden,las frecuencia normalizadas de corte   y  %%%%
%%    el tipo de filtro los parámetros de salida son:                   %%%%
%%       *  n      :orden                                               %%%
%%       *  wc     :frecuencias de corte normalizada (0 hasta 1=pi)      %%
%%       *  tipo   : tipo 1 LP,2 HP, 3BP  y 4RB                         %%
%%%                                                                     %%
%%%         function [n,wc,tipo]=menu_fir                               %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
            function [n,wc,tipo,error] = menu_fir

error=0;
mens='*** DISEÐO DE FILTROS DIGITALES *** ';
tipo = -1;
 while ((tipo<1)|(tipo>4))
   clc
   disp(mens);
       tipo  =  menu('ELIGE  TIPO  DE  FILTRO  FIR','PASO  BAJO.','PASO  ALTO.','PASO
BANDA.','RECHAZO BANDA.');

 end
   n =input('Entra el orden del filtro: ');
   clc
      if ((tipo==2)|(tipo==4))
        p=(n/2)-floor(n/2);   % Arreglamos el orden para filtro
         if (p~=0)            % paso alto y rechazo banda debido a las
            n=n+1;            % restricciones de las funciones de Matlab
         end
      end


      if tipo == 3
       esp_f=-1;
       while((esp_f<1)|(esp_f>2))
         clc
         disp(mens);
                        esp_f  =  menu('ELIGE  TIPO  ESPECIFICACIËN','Especificaci³n
ANALËGICA','Especificaci³n DIGITAL');
       end

          if (esp_f==1)
             f_m = input ('Entra frecuencia de muestreo (Hz): ');
              wc(1)=input ('Entra frecuencia de corte inferior del filtro 0-fm/2 (Hz):
');
```

```
            wc(2)=input ('Entra frecuencia de corte superior del filtro 0-fm/2 (Hz):
');
            wc(1)=(wc(1)*2)/f_m;
            wc(2)=(wc(2)*2)/f_m;
          end

          if (esp_f==2)
            wc(1)=input ('Entra frecuencia normalizada de corte inferior del filtro 0-
pi (rad): ');
            wc(2)=input ('Entra frecuencia normalizada de corte superior del filtro 0-
pi (rad): ');
            wc(1)=wc(1)/pi;
            wc(2)=wc(2)/pi;
          end
      end

      if tipo == 4
       esp_f=-1;
       while((esp_f<1)|(esp_f>2))
         clc
         disp(mens);
                        esp_f   =   menu('ELIGE   TIPO   ESPECIFICACIÈN','Especificaci³n
ANALÈGICA','Especificaci³n DIGITAL');
       end
        if (esp_f==1)
            f_m = input ('Entra frecuencia de muestreo en (Hz): ');
            wc(1)=input ('Entra frecuencia corte inferior del filtro 0-fm/2 (Hz): ');
            wc(2)=input ('Entra frecuencia corte superior del filtro 0-fm/2 (Hz): ');
            wc(1)=(wc(1)*2)/f_m;
            wc(2)=(wc(2)*2)/f_m;
        end

        if (esp_f==2)
            wc(1)=input ('Entra frecuencia normalizada corte inferior del filtro 0-pi
(rad): ');
            wc(2)=input ('Entra frecuencia normalizada corte superior del filtro 0-pi
(rad): ');
          wc(1)=wc(1)/pi;
          wc(2)=wc(2)/pi;
         end
      end

      if ((tipo==1)|(tipo==2))
       esp_f=-1;
       while((esp_f<1)|(esp_f>2))
         clc
         disp(mens);
                        esp_f   =   menu('ELIGE   TIPO   ESPECIFICACION','Especificaci³n
ANALOGICA','Especificaci³n DIGITAL');
       end
            if (esp_f==1)
              f_m= input ('Entra frecuencia de muestreo en (Hz): ');
              wc = input ('Entra la frecuencia de corte del filtro 0-fm/2 (Hz): ');
              wc=(2*wc)/f_m;
            end

            if (esp_f==2)
                wc = input ('Entra la frecuencia normalizada de corte del filtro 0-pi
(rad): ');
              wc=wc/pi;
            end
      end

if ((n./round(n)~=1)|(n<1))
   error=1;
end

if (length(wc)==2)
  if ((wc(1)>1)|(wc(1)<0)|(wc(2)>1)|(wc(2)<0)|(wc(1)>wc(2)))
      error=1;
  end
end

if (length(wc)==1)
   if ((wc>1)|(wc<0))
      error =1;
   end
end
```

LISTADO DE MENU_IIR.M

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%              FUNCION QUE PIDE LOS PARAMETROS DEL IIR                    %%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%    Da como salida las frecuencia normalizadas ,atenuacion y              %%%%
%%    el tipo de filtro los parámetros de salida son:                       %%%%
%%       *  w_paso    :frecuencia de paso  normalizada                      %%%
%%       *  w_aten    :frecuencias atenuada normalizada (0 hasta 1=pi)       %%
%%       *  a_paso    :atenuacion en la banda de paso                        %%
%%       *  a_aten    :atenuacion en la banda atenuada                       %%
%%       *  tipo      :tipo 1 LP,2 HP, 3BP  y 4RB                            %%
%%%                                                                         %%
%%%  function [w_paso,w_aten,a_paso,a_aten,tipo]=menu_iir                   %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
     function [w_paso,w_aten,a_paso,a_aten,tipo,error]=menu_iir

error=0;
mens='*** DISEÑO DE FILTROS DIGITALES ***';
tipo = -1;
 while ((tipo<1)|(tipo>4))
   clc
   disp (mens);
       tipo  =  menu('ELIGE  TIPO  DE  FILTRO  IIR','PASO  BAJO.','PASO  ALTO.','PASO
BANDA.','RECHAZO BANDA.');
 end
 clc
     if (tipo == 3) % paso banda
         tip_esp=-1;
         while((tip_esp<1)|(tip_esp>2))
           clc
           disp(mens)
                                tip_esp=menu('ELIGE  TIPO  ESPECIFICACIÓN','Especificaci³n
ANALÓGICA','Especificaci³n DIGITAL');
         end
         if (tip_esp==1)
           f_m=input('Entra frecuencia de muestreo (Hz): ');
             w_paso(1)=input('Entra frecuencia de paso inferior del filtro 0-fm/2 (Hz):
');
             w_paso(2)=input('Entra frecuencia de paso superior del filtro 0-fm/2 (Hz):
');
             w_aten(1)=input('Entra frecuencia de atenuaci³n inferior del filtro 0-fm/2
(Hz): ');
              w_aten(2)=input('Entra frecuencia de atenuaci³n supeior del filtro 0-fm/2
(Hz): ');
           w_paso=(w_paso*2)/f_m;
           w_aten=(w_aten*2)/f_m;
         end
         if (tip_esp==2)
            w_paso(1)=input('Entra frecuencia normalizada de paso inferior del filtro 0-
pi (rad): ');
            w_paso(2)=input('Entra frecuencia normalizada de paso superior del filtro 0-
pi (rad): ');
           w_paso=w_paso/pi;
              w_aten(1)=input('Entra frecuencia normalizada de atenuaci³n inferior del
filtro 0-pi (rad): ');
               w_aten(2)=input('Entra frecuencia normalizada de atenuaci³n superior del
filtro 0-pi (rad): ');
           w_aten=w_aten/pi;
         end
      end

      if (tipo == 4) % rechazo banda
         tip_esp=-1;
         while((tip_esp<1)|(tip_esp>2))
           clc
           disp(mens)
                                tip_esp=menu('ELIGE  TIPO  ESPECIFICACIÓN','Especificaci³n
ANALÓGICA','Especificaci³n DIGITAL');
         end
         if(tip_esp==1)
           f_m=input('Entra la frecuencia de muestreo (Hz): ');
            w_aten(1)=input ('Entra frecuencia de paso inferior del filtro 0-fm/2 (Hz):
');
            w_aten(2)=input ('Entra frecuencia de paso superior del filtro 0-fm/2 (Hz):
');
```

```
            w_paso(1)=input ('Entra frecuencia de atenuaci³n inferior del filtro 0-fm/2
(Hz): ');
            w_paso(2)=input ('Entra frecuencia de atenuaci³n superior del filtro 0-fm/2
(Hz): ');
          w_aten=((2*w_aten)/f_m);
          w_paso=((2*w_paso)/f_m);
        end
        if(tip_esp==2)
           w_aten(1)=input ('Entra frecuencia normalizada de paso inferior 0-pi (rad):
');
           w_aten(2)=input ('Entra frecuencia normalizada de paso superior 0-pi (rad):
');
          w_aten=w_aten/pi;
            w_paso(1)=input('Entra frecuencia normalizada de atenuaci³n inferior 0-pi
(rad): ');
            w_paso(2)=input('Entra frecuencia normalizada de atenuaci³n superior 0-pi
(rad): ');
          w_paso=w_paso/pi;
        end
    end

    if (tipo==1) %paso bajo
       tip_esp=-1;
       while((tip_esp<1)|(tip_esp>2))
         clc
         disp(mens)
                              tip_esp=menu('ELIGE  TIPO  ESPECIFICACIËN','Especificaci³n
ANALËGICA','Especificaci³n DIGITAL');
        end
        if (tip_esp==1)
          f_m=input('Entra la frecuencia de muestreo (Hz): ');
          w_paso=input ('Entra la frecuencia de paso del filtro 0-fm/2 (Hz): ');
          w_aten=input ('Entra la frecuencia de atenuaci³n del filtro 0-fm/2 (Hz): ');
          w_paso=((2*w_paso)/f_m);
          w_aten=((2*w_aten)/f_m);
        end
        if (tip_esp==2)
          w_paso=input ('Entra la frecuencia normalizada de paso del filtro 0-pi (rad):
');
          w_paso=w_paso/pi;
           w_aten=input ('Entra la frecuencia normalizada de atenuaci³n del filtro 0-pi
(rad): ');
          w_aten=w_aten/pi;
        end
    end

    if (tipo==2) % paso alto
       tip_esp=-1;
       while((tip_esp<1)|(tip_esp>2))
         clc
         disp(mens)
                              tip_esp=menu('ELIGE  TIPO  ESPECIFICACIËN','Especificaci³n
ANALËGICA','Especificaci³n DIGITAL');
        end
        if(tip_esp==1)
          f_m=input('Entra la frecuencia de muestreo (Hz): ');
          w_aten=input ('Entra la frecuencia de paso del filtro 0-fm/2 (Hz): ');
          w_paso=input ('Entra la frecuencia de atenuaci³n del filtro 0-fm/2 (Hz): ');
          w_aten=((2*w_aten)/f_m);
          w_paso=((2*w_paso)/f_m);
        end
        if(tip_esp==2)
          w_aten=input ('Entra la frecuencia normalizada de paso del filtro 0-pi (rad):
');
          w_aten=w_aten/pi;
           w_paso=input ('Entra la frecuencia normalizada de atenuaci³n de 0-pi (rad):
');
          w_paso=w_paso/pi;
        end
    end

  a_paso = input ('Valor de atenuaci³n en la banda de paso [>0dB](dB): ');
  a_aten = input ('Valor de atenuaci³n en la banda atenuada [>0dB](dB): ');

if ((a_aten<=0)|(a_paso<=0))
   error=1;
end
```

```
if (length(w_paso)==2)
       if((w_paso(1)>1)|(w_paso(1)<0)|(w_paso(2)>1)|(w_paso(2)<0)|(w_paso(1)>w_paso(2)))
         error=1;
       end
       if((w_aten(1)>1)|(w_aten(1)<0)|(w_aten(2)>1)|(w_aten(2)<0)|(w_aten(1)>w_aten(2)))
         error=1;
       end
       if ((w_aten(1)>w_paso(1))|(w_aten(2)<w_paso(2)))
         error=1;
       end
end

if (length(w_paso)==1)
       if ((w_paso<0)|(w_paso>1)|(w_aten<0)|(w_aten>1)|(w_paso>w_aten))
          error=1;
       end
end


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

LISTADO DE PARL_IIR.M

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%    Función que realiza la función de transferencia del filtro IIR   %%%%
%%%    en la realización en paralelo                                    %%%%
%%%   parametro de entrada:                                             %%%%
%%%       Coeficientes del filtro x_n,y_n                               %%%%
%%%   paramatro de salida:                                              %%%%
%%%       matriz de realización en paralelo x_n_p,y_n_p,const           %%%%
%%%   cada fila de la matriz da un sistema en paralelo                  %%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%    function [x_n_p,y_n_p,const]=parl_iir(x_n,y_n)                   %%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
       function [x_n_p,y_n_p,const]=parl_iir(x_n,y_n)
x_n=rot90(x_n);
x_n=rot90(x_n);  %% ponemos en potencias crecientes la función H(z)
y_n=rot90(y_n);
y_n=rot90(y_n);

[resid,polos,const]=residue(x_n,y_n); %% Cálculo de residuos,polos y cte.
const=sum(const);
num_polos=length(polos);
x_n_p=[];
y_n_p=[];
c=1;

   while (c<=num_polos)
     k=imag(polos(c));

     if k==0
        term1=[0,0,resid(c)];      %% Cálculo de los términos cuando los
        x_n_p=[x_n_p;term1];       %% polos son reales (los residuos también
        term2=[0,poly([polos(c)])];%% son reales)
        y_n_p=[y_n_p;term2];
        c=c+1;
     else
              %% Cálculo de los términos cuando los polos son complejos
        term3=[0,resid(c)+resid(c+1),-(resid(c)*polos(c+1)+resid(c+1)*polos(c))];
        x_n_p=[x_n_p;term3];
        y_n_p=[y_n_p;poly([polos(c),polos(c+1)])];
        c=c+2;
     end
   end
x_n_p=rot90(x_n_p);
x_n_p=rot90(x_n_p);  %% ponemos los términos en potencias decrecientes
y_n_p=rot90(y_n_p);
y_n_p=rot90(y_n_p);
```

LISTADO DE PRESENT.M

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  PANTALLA DE PRESENTACION DEL PROGRAMA DE ANALISIS DE FILTROS DIGITALES  %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
 clc
 clg
 load f_pres.mat ; % La grafica estß en la variable H_M
 f=-pi:(2*pi)/200:pi-(2*pi/200);
```

```
subplot(111)
plot(f,H_M,'y');
xlabel('Frecuencia digital');
ylabel('Amplitud en dB');
title('Programa de simulaci³n  <<DISFILT>>')
text(-2,14,'AUTOR TFC:Rafael Socas GutiÚrrez.');
text(-2,6,'TUTOR:D. Juan RuÝz Alzola.')
```

LISTADO DE PRG_COEF.M

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%      FUNCION QUE PROGRAMA LOS COEFICIENTES EN EL DSP      %%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%   Valores de entrada:                                                  %%%%%
%%%%       * x_n_c     * y_n_c                                            %%%%%
%%%%   Valores de salida:                                                   %%%%%
%%%%       * x_prog    *y_prog                                            %%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%          function [x_prog,y_prog]=pro_coef(x_n_c,y_n_c)              %%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
             function [x_prog,y_prog]=pro_coef(x_n_c,y_n_c)

y_prog=[];        % Programación de los coeficientes AN

  if (abs(1/y_n_c(1))==1)
     y_prog(1,1)=0;
  else                    % Valor de A0 programado
     y_prog(1,1)=1;
  end

  y_prog(1,2)=abs(1/y_n_c(1));

  if ((1/y_n_c(1))>0)
     y_prog(1,3)=0;
  else
     y_prog(1,3)=1;
  end

for i=2:length(y_n_c)

    if ((abs(y_n_c(i))>.75)&(abs(y_n_c(i))<=1))
        if (y_n_c(i)>0)
           y_prog(i,1)=4;y_prog(i,2)=0;y_prog(i,3)=0;
        else
           y_prog(i,1)=3;y_prog(i,2)=0;y_prog(i,3)=0;
        end
    end

    if ((y_n_c(i)>0)&(y_n_c(i)<.75))
           y_prog(i,1)=2;
    end

    if ((y_n_c(i)<0)&(y_n_c(i)>-.75))
           y_prog(i,1)=1;
    end

    if (y_n_c(i)==0)
           y_prog(i,1)=0;y_prog(i,2)=0;y_prog(i,3)=0;
    end

    if ((abs(y_n_c(i))<.75)&(abs(y_n_c(i))>.375))
          y_prog(i,2)=16380;y_prog(i,3)=1;
    end
    if ((abs(y_n_c(i))<.375)&(abs(y_n_c(i))>.1875))
          y_prog(i,2)=8188;y_prog(i,3)=2;
    end
    if ((abs(y_n_c(i))<.1875)&(abs(y_n_c(i))>.09375))
          y_prog(i,2)=4092;y_prog(i,3)=3;
    end
    if ((abs(y_n_c(i))<.09375)&(abs(y_n_c(i))>.046875))
          y_prog(i,2)=2044;y_prog(i,3)=4;
    end
    if ((abs(y_n_c(i))<.046875)&(abs(y_n_c(i))>.0234375))
          y_prog(i,2)=1022;y_prog(i,3)=5;
    end
    if ((abs(y_n_c(i))<.0234375)&(abs(y_n_c(i))>.01171875))
          y_prog(i,2)=508;y_prog(i,3)=6;
```

```
    end
    if ((abs(y_n_c(i))<.01171875)&(abs(y_n_c(i))>.005859375))
        y_prog(i,2)=252;y_prog(i,3)=7;
    end
    if ((abs(y_n_c(i))<.005859375)&(abs(y_n_c(i))>.002929687))
        y_prog(i,2)=124;y_prog(i,3)=8;
    end
    if ((abs(y_n_c(i))<.002929687)&(abs(y_n_c(i))>.00146484375))
        y_prog(i,2)=60;y_prog(i,3)=9;
    end
    if ((abs(y_n_c(i))<.00146484375)&(abs(y_n_c(i))>.000732421875))
        y_prog(i,2)=28;y_prog(i,3)=10;
    end
    if ((abs(y_n_c(i))<.000732421875)&(abs(y_n_c(i))>.0003662109))
        y_prog(i,2)=12;y_prog(i,3)=11;
    end
    if ((abs(y_n_c(i))<.0003662109)&(abs(y_n_c(i))>.00012207))
        y_prog(i,2)=4;y_prog(i,3)=12;
    end
    if (abs(y_n_c(i))<.00012207)
        y_prog(i,1)=0;y_prog(i,2)=0;y_prog(i,3)=0;
    end

end

x_prog=[]; % Programamos coeficientes BN

for i=1:length(x_n_c)

    if ((abs(x_n_c(i))>.75)&(abs(x_n_c(i))<=1))
        if (x_n_c(i)>0)
            x_prog(i,1)=4;x_prog(i,2)=0;x_prog(i,3)=0;
        else
            x_prog(i,1)=3;x_prog(i,2)=0;x_prog(i,3)=0;
        end
    end

    if ((x_n_c(i)>0)&(x_n_c(i)<.75))
        x_prog(i,1)=2;
    end

    if ((x_n_c(i)<0)&(x_n_c(i)>-.75))
        x_prog(i,1)=1;
    end

    if (x_n_c(i)==0)
        x_prog(i,1)=0;x_prog(i,2)=0;x_prog(i,3)=0;
    end

    if ((abs(x_n_c(i))<.75)&(abs(x_n_c(i))>.375))
        x_prog(i,2)=16380;x_prog(i,3)=1;
    end
    if ((abs(x_n_c(i))<.375)&(abs(x_n_c(i))>.1875))
        x_prog(i,2)=8188;x_prog(i,3)=2;
    end
    if ((abs(x_n_c(i))<.1875)&(abs(x_n_c(i))>.09375))
        x_prog(i,2)=4092;x_prog(i,3)=3;
    end
    if ((abs(x_n_c(i))<.09375)&(abs(x_n_c(i))>.046875))
        x_prog(i,2)=2044;x_prog(i,3)=4;
    end
    if ((abs(x_n_c(i))<.046875)&(abs(x_n_c(i))>.0234375))
        x_prog(i,2)=1022;x_prog(i,3)=5;
    end
    if ((abs(x_n_c(i))<.0234375)&(abs(x_n_c(i))>.01171875))
        x_prog(i,2)=508;x_prog(i,3)=6;
    end
    if ((abs(x_n_c(i))<.01171875)&(abs(x_n_c(i))>.005859375))
        x_prog(i,2)=252;x_prog(i,3)=7;
    end
    if ((abs(x_n_c(i))<.005859375)&(abs(x_n_c(i))>.002929687))
        x_prog(i,2)=124;x_prog(i,3)=8;
    end
    if ((abs(x_n_c(i))<.002929687)&(abs(x_n_c(i))>.00146484375))
        x_prog(i,2)=60;x_prog(i,3)=9;
    end
    if ((abs(x_n_c(i))<.00146484375)&(abs(x_n_c(i))>.000732421875))
        x_prog(i,2)=28;x_prog(i,3)=10;
```

```
    end
    if ((abs(x_n_c(i))<.000732421875)&(abs(x_n_c(i))>.0003662109))
        x_prog(i,2)=12;x_prog(i,3)=11;
    end
    if ((abs(x_n_c(i))<.0003662109)&(abs(x_n_c(i))>.00012207))
        x_prog(i,2)=4;x_prog(i,3)=12;
    end
    if (abs(x_n_c(i))<.00012207)
        x_prog(i,1)=0;x_prog(i,2)=0;x_prog(i,3)=0;
    end

end
```

LISTADO DE RE_C_S.M

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%     FUNCION QUE DA LA SALIDA DE LA REALIZACION EN CASCADA          %%%%%
%%%%  PARA EL DSP TMDS3200051 CON LOS COEFICIENTES YA CUANTIFICADOS     %%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Parámetros de entrada:                                                   %%
%%%    * x_n_c_c: matriz de coeficientes de x_n de los sistemas en cascada%%
%%%        cada sistema esta en una fila.                                   %%
%%%    * y_n_c_c: matriz de coeficientes de y_n de los sistemas en cascada%%
%%%        cada sistema está en una fila.                                   %%
%%%    * se_ent: señal de entrada al DSP.                                   %%
%%% Parámetros de salida:                                                    %%
%%%    * se_sal: señal de salida del DSP.                                   %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%         function se_sal=re_c_s(x_n_c_c,y_n_c_c,se_ent)              %%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
            function se_sal=re_c_s(x_n_c_c,y_n_c_c,se_ent)

l_x=size(x_n_c_c);
l_y=size(y_n_c_c);
l_x=l_x(1);
l_y=l_y(1);
                          % rellenamos las matrices de coeficientes
   if (l_x>l_y)           % con (1,0,0) factor unitario en la cascada
      for i=(l_y+1):l_x   % para luego no tener problemas al referenciarlas
         y_n_c_c(i,1)=1;
         y_n_c_c(i,2)=0;
         y_n_c_c(i,3)=0;
       end
   end

   if (l_y>l_x)
      for i=(l_x+1):l_y
         x_n_c_c(i,1)=1;
         x_n_c_c(i,2)=0;
         x_n_c_c(i,3)=0;
       end
   end

h1=se_ent;
cont=size(x_n_c_c);
cont=cont(1);

      for i=1:cont
         for j=1:3
           x(j)=x_n_c_c(i,j);  %% calculamos cada sistema
           y(j)=y_n_c_c(i,j);  %% de la cascada
         end
         xa=abs(x);
         ya=abs(y);
         xa=max(xa); %% Cuando los coeficientes de un sistema son cero
         ya=max(ya); %% no lo ponemos.

          if ((xa>0)&(ya>0))
            h2=filter(x,y,h1); %% calculamos la respuesta de cada sistema
            h2=round(h2);%% redondeamos ya que el DSP redondea al dividir
            h1=h2;       %% ,porque divide desplazando bits a la derecha
           end
      end
se_sal=h2/4096;
```

LISTADO DE RE_CAS.M

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%     FUNCION QUE DA LA RESPUESTA DE LA REALIZACION EN CASCADA      %%%%%
%%%%  PARA EL DSP TMDS3200051 CON LOS COEFICIENTES YA CUANTIFICADOS    %%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Parámetros de entrada:                                                  %%
%%%    * x_n_c_c: matriz de coeficientes de x_n de los sistemas en cascada%%
%%%      cada sistema esta en una fila.                                     %%
%%%    * y_n_c_c: matriz de coeficientes de y_n de los sistemas en cascada%%
%%%      cada sistema está en una fila.                                     %%
%%% Parámetros de salida:                                                   %%
%%%    * h_c: respuesta al impulso del sistema en cascada                 %%
%%%   Da una respuesta de longitud 200                                    %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%            function h_c=re_cas(x_n_c_c,y_n_c_c)                     %%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                function h_c=re_cas(x_n_c_c,y_n_c_c)

l_x=size(x_n_c_c);
l_y=size(y_n_c_c);
l_x=l_x(1);
l_y=l_y(1);
                          % rellenamos las matrices de coeficientes
   if (l_x>l_y)           % con (1,0,0) factor unitario en la cascada
      for i=(l_y+1):l_x   % para luego no tener problemas al referenciarlas
         y_n_c_c(i,1)=1;
         y_n_c_c(i,2)=0;
         y_n_c_c(i,3)=0;
       end
   end

   if (l_y>l_x)
      for i=(l_x+1):l_y
         x_n_c_c(i,1)=1;
         x_n_c_c(i,2)=0;
         x_n_c_c(i,3)=0;
       end
   end

delta=[4096 zeros(1,199)];%% generamos la delta del DSP
h1=delta;
cont=size(x_n_c_c);
cont=cont(1);

      for i=1:cont
         for j=1:3
           x(j)=x_n_c_c(i,j);  %% calculamos cada sistema
           y(j)=y_n_c_c(i,j);  %% de la cascada
         end
         xa=abs(x);
         ya=abs(y);
         xa=max(xa); %% Cuando los coeficientes de un sistema son cero
         ya=max(ya); %% no lo ponemos.

          if ((xa>0)&(ya>0))
            h2=filter(x,y,h1); %% calculamos la respuesta de cada sistema
            h2=round(h2);%% redondeamos ya que el DSP redondea al dividir
            h1=h2;       %% ,porque divide desplazando bits a la derecha
          end
      end
h_c=h2/4096;
```

LISTADO DE RE_CO_FI.M

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%    FUNCION QUE REPRESENTA LAS CARACTERISTICAS DE LAS DIFERENTES     %%%%
%%%%           REALIZACIONES . DIRECTA , CASCADA Y PARALELO              %%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%   Los parámetros de entrada son:                                     %%%%
%%%%   * hi :respuesta al impulso ideal                                   %%%%
%%%%   * hd :respuesta al impulso real con realización directa            %%%%
%%%%   * hc :respuesta al impulso real con realización en cascada.        %%%%
%%%%   * hp :respuesta al impulso real con realización en paralelo.       %%%%
%%%%   * tipo :tipo de filtro 1 FIR, 0 IIR ,2 OTRO                        %%%%
%%%%     Supondremos que las h[n] tiene 200 valores                       %%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%            function re_co_fi(hi,hd,hc,hp,tipo)                      %%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
                function re_co_fi(hi,hd,hc,hp,tipo)

HI=fft(hi);
HD=fft(hd);    % Calculamos y truncamos las TF de las diferentes h[n]
HC=fft(hc);
HI=(HI(1:100));
HD=(HD(1:100));
HC=(HC(1:100));
M_I=abs(HI);
M_I_D=20*(log(M_I)/log(10));
F_I=angle(HI);                  %% Hallamos los módulos y las fases
M_D=abs(HD);                    %% de las TFs.
M_D_D=20*(log(M_D)/log(10));
F_D=angle(HD);
M_C=abs(HC);
M_C_D=20*(log(M_C)/log(10));
F_C=angle(HC);

   if tipo==0                %%% Cuando sea un filtro IIR calculamos
      HP=fft(hp);            %%% la debida a la realización en paralelo
      HP=(HP(1:100));
      M_P=abs(HP);
      M_P_D=20*(log(M_P)/log(10));
      F_P=angle(HP);
   end

f_norm=0:3.14/100:3.14-(3.14/100); %% vector de frecuencias digitales

clg
  if tipo==0      %%% Representamos los parámetros para IIR o FIR respectivamente
     subplot(223)
     plot(f_norm,M_I,'-w',f_norm,M_D,'--r',f_norm,M_C,'--g',f_norm,M_P,'--b');
     xlabel('Frecuencia digital (rad)');
     ylabel('Módulo lineal');
     title('MODULO DEL FILTRO');
  end
  if tipo==1
     subplot(223)
     plot(f_norm,M_I,'-w',f_norm,M_D,'--r',f_norm,M_C,'--g');
     xlabel('Frecuencia digital (rad)');
     ylabel('Módulo lineal');
     title('MODULO DEL FILTRO');
  end
  if tipo==2
     subplot(223)
     plot(f_norm,M_I,'-w',f_norm,M_D,'--r');
     xlabel('Frecuencia digital (rad)');
     ylabel('Módulo lineal');
     title('MODULO DEL FILTRO');
  end

  if tipo==0
     subplot(224)
     plot(f_norm,F_I,'-w',f_norm,F_D,'--r',f_norm,F_C,'--g',f_norm,F_P,'--b');
     xlabel('Frecuencia digital (rad)');
     ylabel('Fase (rad)');
     title('FASE DEL FILTRO');
  end
  if tipo==1
     subplot(224)
     plot(f_norm,F_I,'-w',f_norm,F_D,'--r',f_norm,F_C,'--g');
     xlabel('Frecuencia digital (rad)');
     ylabel('Fase (rad)');
     title('FASE DEL FILTRO');
  end
  if tipo==2
     subplot(224)
     plot(f_norm,F_I,'-w',f_norm,F_D,'--r');
     xlabel('Frecuencia digital (rad)');
     ylabel('Fase (rad)');
     title('FASE DEL FILTRO');
  end

  if tipo==0
     subplot(211)
     plot(f_norm,M_I_D,'-w',f_norm,M_D_D,'--r',f_norm,M_C_D,'--g',f_norm,M_P_D,'--b');
     xlabel('Frecuencia digital (rad)');
```

```
    ylabel('M³dulo en dB');
    title('blanco--IDEAL  rojo--DIRECTA  verde--CASCADA  azul--PARALELO');
       if (max(M_D)>10)
          kiko=menu('Realizaci³n DIRECTA *NO DEBE IMPLEMENTARSE*','CONTINUAR');
       end
       if (max(M_C)>10)
          kiko=menu('Realizaci³n CASCADA *NO DEBE IMPLEMENTARSE*','CONTINUAR');
       end
       if (max(M_P)>10)
          kiko=menu('Realizaci³n PARALELO *NO DEBE IMPLEMENTARSE*','CONTINUAR');
       end

 end
 if tipo==1
    subplot(211)
    plot(f_norm,M_I_D,'-w',f_norm,M_D_D,'--r',f_norm,M_C_D,'--g');
    xlabel('Frecuencia digital (rad)');
    ylabel('M³dulo en dB');
    title('blanco--IDEAL  rojo--DIRECTA  verde--CASCADA');
       if (max(M_D)>10)
          kiko=menu('Realizaci³n DIRECTA *NO DEBE IMPLEMENTARSE*','CONTINUAR');
       end
       if (max(M_C)>10)
          kiko=menu('Realizaci³n CASCADA *NO DEBE IMPLEMENTARSE*','CONTINUAR');
       end
 end

 if tipo==2
    subplot(211)
    plot(f_norm,M_I_D,'-w',f_norm,M_D_D,'--r');
    xlabel('Frecuencia digital (rad)');
    ylabel('M³dulo en dB');
    title('blanco--IDEAL  rojo--DIRECTA ');
    if (max(M_D)>10)
          kiko=menu('Realizaci³n DIRECTA *NO DEBE IMPLEMENTARSE*','CONTINUAR');
    end

 end


 if tipo==0
    kiko=menu('  ','CONTINUAR');
    close

    subplot(223) %forma directa
    plot(f_norm,M_I,'-w',f_norm,M_D,'--r');
    xlabel('Frecuencia digital (rad)');
    ylabel('M³dulo lineal');
    title('MODULO DEL FILTRO');
    subplot(224)
    plot(f_norm,F_I,'-w',f_norm,F_D,'--r');
    xlabel('Frecuencia digital (rad)');
    ylabel('Fase (rad)');
    title('FASE DEL FILTRO');
    subplot(211)
    plot(f_norm,M_I_D,'-w',f_norm,M_D_D,'--r');
    xlabel('Frecuencia digital (rad)');
    ylabel('M³dulo en dB');
    title('blanco--IDEAL  rojo--DIRECTA ');
    kiko=menu('  ','CONTINUAR');
    close

    subplot(223) %forma cascada
    plot(f_norm,M_I,'-w',f_norm,M_C,'--g');
    xlabel('Frecuencia digital (rad)');
    ylabel('M³dulo lineal');
    title('MODULO DEL FILTRO');
    subplot(224)
    plot(f_norm,F_I,'-w',f_norm,F_C,'--g');
    xlabel('Frecuencia digital (rad)');
    ylabel('Fase (rad)');
    title('FASE DEL FILTRO');
    subplot(211)
    plot(f_norm,M_I_D,'-w',f_norm,M_C_D,'--g');
    xlabel('Frecuencia digital (rad)');
    ylabel('M³dulo en dB');
    title('blanco--IDEAL  verde--CASCADA');
    kiko=menu('  ','CONTINUAR');
```

```
    close

    subplot(223) %forma paralelo
    plot(f_norm,M_I,'-w',f_norm,M_P,'--b');
    xlabel('Frecuencia digital (rad)');
    ylabel('Mºdulo lineal');
    title('MODULO DEL FILTRO');
    subplot(224)
    plot(f_norm,F_I,'-w',f_norm,F_P,'--b');
    xlabel('Frecuencia digital (rad)');
    ylabel('Fase (rad)');
    title('FASE DEL FILTRO');
    subplot(211)
    plot(f_norm,M_I_D,'-w',f_norm,M_P_D,'--b');
    xlabel('Frecuencia digital (rad)');
    ylabel('Mºdulo en dB');
    title('blanco--IDEAL azul--PARALELO');

end

if (tipo==1)
    kiko=menu('  ','CONTINUAR');
    close

    subplot(223) %forma directa
    plot(f_norm,M_I,'-w',f_norm,M_D,'--r');
    xlabel('Frecuencia digital (rad)');
    ylabel('Mºdulo lineal');
    title('MODULO DEL FILTRO');
    subplot(224)
    plot(f_norm,F_I,'-w',f_norm,F_D,'--r');
    xlabel('Frecuencia digital (rad)');
    ylabel('Fase (rad)');
    title('FASE DEL FILTRO');
    subplot(211)
    plot(f_norm,M_I_D,'-w',f_norm,M_D_D,'--r');
    xlabel('Frecuencia digital (rad)');
    ylabel('Mºdulo en dB');
    title('blanco--IDEAL  rojo--DIRECTA');
    kiko=menu('  ','CONTINUAR');
    close

    subplot(223) %forma cascada
    plot(f_norm,M_I,'-w',f_norm,M_C,'--g');
    xlabel('Frecuencia digital (rad)');
    ylabel('Mºdulo lineal');
    title('MODULO DEL FILTRO');
    subplot(224)
    plot(f_norm,F_I,'-w',f_norm,F_C,'--g');
    xlabel('Frecuencia digital (rad)');
    ylabel('Fase (rad)');
    title('FASE DEL FILTRO');
    subplot(211)
    plot(f_norm,M_I_D,'-w',f_norm,M_C_D,'--g');
    xlabel('Frecuencia digital (rad)');
    ylabel('Mºdulo en dB');
    title('blanco--IDEAL verde--CASCADA');

end
```

LISTADO DE RE_FI_A.M

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%        representación del MODULO,RETARDO DE GRUPO                          %%%
%%                  de un filtro ANALOGICO                                    %%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%       function  re_fi_a(b,a,fm)                                           %%%
%%%%%     fm=frecuencia de muestreo                                           %%%
%%%%%     a=vector de coeficientes y[n]                                       %%%
%%%%%     b=vector de coeficientes x[n]                                       %%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
         function  re_fi_a(b,a,fm)

 delta_t=[1 zeros(1,199)];          % generamos la delta de dirac
 h_t=filter(b,a,delta_t);           % cálculo respuesta al impulso
 frecuencia=0:fm/200:fm/2-(fm/200);    % vector de frecuencia
 H_w=fft(h_t);                      % transformada de fourier de h(t);
```

```
H_w=(H_w(1:100));                    % truncamos y cogemos el espectro positivo
[Ret_grupo]=grpdelay(b,a,100);       % cálculo de retardo de grupo
Ret_grupo=Ret_grupo/fm;              %Pasamos retardo de grupo a segundos
H_modulo=abs(H_w);                   % módulo de H(W)
H_modulo_log=20*(log(H_modulo)/log(10));  % módulo en dB


clg
subplot(222);plot(frecuencia,Ret_grupo,'g');    % representa retardo de grupo
xlabel('Frecuencia (Hz)');ylabel('Segundos');
title('RETARDO DE GRUPO');grid;

subplot(212);plot(frecuencia,H_modulo_log,'g'); % representamos el módulo en dB
xlabel('Frecuencia (Hz)');ylabel('Mºdulo en dB (Voltaje)');
title('MËDULO DE H(W)');grid;


subplot(221);plot(frecuencia,H_modulo,'g');         % módulo lineal
xlabel('Frecuencia (Hz)');ylabel('Mºdulo lineal ');
title('MËDULO DE H(W)');grid;
```

LISTADO DE RE_FI_AN.M

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%              representación del MODULO,FASE                              %%%
%%                 de un filtro ANALOGICO                                   %%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%       function  re_fi_an(h_t,fm)                                        %%%
%%%%%     fm=frecuencia de muestreo                                         %%%
%%%%%     h_n=respuesta del filtro                                          %%%
%%%%%  nota: Suponemos que h_t tiene 200 muestras                           %%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
          function  re_fi_an(h_t,fm)

 frecuencia=0:fm/200:fm/2-(fm/200);    % vector de frecuencia
 H_w=fft(h_t);                        % transformada de fourier de h(t);
 H_w=(H_w(1:100));                     % truncamos y cogemos el espectro positivo
 H_modulo=abs(H_w);                    % módulo de H(W)
 H_modulo_log=20*(log(H_modulo)/log(10));  % módulo en dB
 H_fase=angle(H_w);                    % Calculamos la fase

 clg
 subplot(222);plot(frecuencia,H_fase,'g');    % representa la fase
 xlabel('Frecuencia (Hz)');ylabel('Radianes');
 title('FASE');grid;

 subplot(212);plot(frecuencia,H_modulo_log,'g'); % representamos el módulo en dB
 xlabel('Frecuencia (Hz)');ylabel('Mºdulo en dB (Voltaje)');
 title('MODULO DE H(W)');grid;


 subplot(221);plot(frecuencia,H_modulo,'g');         % módulo lineal
 xlabel('Frecuencia (Hz)');ylabel('Mºdulo lineal ');
 title('MODULO DE H(W)');grid;
```

LISTADO DE RE_FILT.M

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%        representación del MODULO,RETARDO DE GRUPO Y DIAGRAMA            %%%
%%              DE POLOS Y CEROS de un filtro DIGITAL                      %%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%       function  re_filt(b,a)                                            %%%
%%%%%     a=vector de coeficientes y[n]                                     %%%
%%%%%     b=vector de coeficientes x[n]                                     %%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
          function  re_filt(b,a)

 delta_n=[1 zeros(1,199)];          % generamos la delta de dirac
 h_n=filter(b,a,delta_n);           % cálculo respuesta al impulso
 f_norm=0:3.14/100:3.14-(3.14/100);  % vector de frecuencia normalizada
 H_w=fft(h_n);                    % transformada de fourier de h(n);
 H_w=(H_w(1:100));                   % truncamos y cogemos el espectro positivo
 H_fase=angle(H_w);      % Calculamos la fase
[Ret_grupo]=grpdelay(b,a,100);      % cálculo de retardo de grupo
 H_modulo=abs(H_w);                  % módulo de H(Û)
```

```
H_modulo_log=20*(log(H_modulo)/log(10));  % módulo en dB

  if (length(b)>length(a))
     a=[a zeros(1,length(b)-length(a))];    % Obtenemos una función
  end                                       % en potencia de zn a partir
                                             % de una de z-n
  if (length(a)>length(b))
     b=[b zeros(1,length(a)-length(b))];
  end

   if (length(b)==1)
      ceros=[0];
      polos=[0];
   else
      ceros=roots(b);    % calculamos los polos y los ceros
      polos=roots(a);
   end

tetap=angle(polos);rhop=abs(polos); % polos y ceros en forma polar
tetac=angle(ceros);rhoc=abs(ceros);


t=0:1:100;                             % circulo unidad H(jÛ)
circ_unidad=exp(i*((2*pi)/50)*t);
circ_t=1.1*exp(i*((2*pi)/50)*t);
tetac_u=angle(circ_unidad);rhoc_u=abs(circ_unidad);
teta_t=angle(circ_t); rho_t=abs(circ_t);

clg
subplot(212);plot(f_norm,H_modulo_log,'g'); % representamos el módulo en dB
xlabel('Frecuencia digital (rad)');ylabel('Módulo en dB (Voltaje)');
title('MODULO DEL FILTRO');grid;

subplot(211);plot(f_norm,H_modulo,'g');          % módulo lineal
xlabel('Frecuencia digital (rad)');ylabel('Módulo lineal ');
title('MODULO DEL FILTRO');grid;

kiko= menu(' ','CONTINUAR');
close

subplot(211);plot(f_norm,H_fase,'g');    % representa la fase
xlabel('Frecuencia digital (rad)');ylabel('Radianes');
title('FASE');grid;

subplot(212);plot(f_norm,Ret_grupo,'g');    % representa retardo de grupo
xlabel('Frecuencia digital (rad)');ylabel('Muestras');
title('RETARDO DE GRUPO');grid;

kiko= menu(' ','CONTINUAR');
close

subplot(111);            % representamos polos
polar(tetap,rhop,'*y');title('Diagrama de Polos');

kiko= menu('  ','CONTINUAR');
close

subplot(111);     % representamos ceros
polar(tetac,rhoc,'Oy');title('Diagrama de Ceros');

kiko= menu('    ','CONTINUAR');
close
```

LISTADO DE RE_P_R.M

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%     FUNCION QUE DA LA SALIDA DE LA REALIZACION EN PARALELO        %%%%%
%%%%  PARA EL DSP TMDS3200051 CON LOS COEFICIENTES YA CUANTIFICADOS     %%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Parámetros de entrada:                                                %%
%%%     * x_n_c_p: matriz de coeficientes de x_n de los sistemas en parale.%%
%%%         cada sistema esta en una fila.                                %%
%%%     * y_n_c_p: matriz de coeficientes de y_n de los sistemas en parale.%%
%%%         cada sistema está en una fila.                                %%
%%%     * cont: Constante de las etapas en paralelo                       %%
%%%     * se_ent :Señal de entrada.                                       %%
```

```
%%% Parámetros de salida:                                                    %%
%%%    * se_sal: Señal de salida.                                            %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%     function se_sal=re_p_r(x_n_c_p,y_n_c_p,cont,se_ent)            %%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
         function se_sal=re_p_r(x_n_c_p,y_n_c_p,cont,se_ent)

l_x=size(x_n_c_p);
l_x=l_x(1);
k=length(se_ent);
h_p=zeros(1:k);

      for i=1:l_x
            for j=1:3
              x(j)=x_n_c_p(i,j);
              y(j)=y_n_c_p(i,j);  % Calculamos los coeficientes de los
            end                   % sistemas en paralelo
          h_p=h_p+filter(x,y,se_ent); % calculamos las salidas independientes
          h_p=round(h_p); % Redondeamos igual que lo hace el DSP
      end

hcont=se_ent.*cont; % Respuesta debido al término constante
h_p=hcont+h_p; % respuesta total del sistema
h_p=round(h_p);
h_p=h_p/4096;
se_sal=h_p;  % Obtenemos la salida que tendría el DSP

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

LISTADO DE RE_PAR.M

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%     FUNCION QUE DA LA RESPUESTA DE LA REALIZACION EN PARALELO     %%%%%
%%%%  PARA EL DSP TMDS3200051 CON LOS COEFICIENTES YA CUANTIFICADOS    %%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Parámetros de entrada:                                                  %%
%%%    * x_n_c_p: matriz de coeficientes de x_n de los sistemas en parale.%%
%%%        cada sistema esta en una fila.                                   %%
%%%    * y_n_c_p: matriz de coeficientes de y_n de los sistemas en parale.%%
%%%        cada sistema está en una fila.                                   %%
%%%    * cont: Constante de las etapas en paralelo                          %%
%%% Parámetros de salida:                                                    %%
%%%    * h_p: respuesta al impulso del sistema en paralelo                  %%
%%%   Da una respuesta de longitud 200                                      %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%            function h_p=re_par(x_n_c_p,y_n_c_p,cont)              %%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                function h_p=re_par(x_n_c_p,y_n_c_p,cont)

l_x=size(x_n_c_p);
l_x=l_x(1);
delta=[4096 zeros(1,199)];%% generamos la delta del DSP
h_p=zeros(size(1:200));

      for i=1:l_x
            for j=1:3
              x(j)=x_n_c_p(i,j);
              y(j)=y_n_c_p(i,j);  % Calculamos los coeficientes de los
            end                   % sistemas en paralelo
          h_p=h_p+filter(x,y,delta); % calculamos las respuestas globales
          h_p=round(h_p); % Redondeamos igual que lo hace el DSP
      end

hcont=delta.*cont; % Respuesta debido al término constante
h_p=hcont+h_p; % respuesta total del sistema
h_p=round(h_p);
h_p=h_p/4096; % Obtenemos la respuesta que tendría el DSP

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

LISTADO DE RE_SE_D.M

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%   Función que representa la señal de salida de un filtros dado y los  %%%
%%   parámetros del filtro por donde pasa la señal ..                     %%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%   Los parámetros de entrada son:                                       %%%
```

```
%%     * h:respuesta al impulso del filtro.                                  %%%
%%     * sen_ent :señal de entrada.                                          %%%
%%     * f_muest :frecuencia de muestreo                                     %%%
%%   Devuelve la señal de salida - sen_sal                                   %%%
%%             function  sen_sal=re_se_d (h,f_muest,sen_ent)                 %%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
             function  sen_sal=re_se_d (h,f_muest,sen_ent)

sen_sal=conv(h,sen_ent);   % Calculamos la señal de salida
sen_sal=sen_sal(1:length(sen_ent));
H_W=fft(h); % Calculamos H(W)
h_n=[];
H_F=angle(H_W);
H_F=H_F(1:100);% Calculamos la fase de H(W)
H_W=abs(H_W); % Calculamos módulo de H(W)
H_W=H_W(1:100); % Cogemos parte positiva del espectro
H_W_D=20*(log(H_W)/log(10)); %modulo en dB de Voltaje
frc=f_muest/2;
f_n=0:frc/100:frc-(frc/100); % Vector de frecuencias

n=length(sen_ent);
S_E=fft(sen_ent);
S_E=abs(S_E);
S_E=S_E(1:n/2);  %% Densidad Espectral de energia de la señal de entrada
DEE_E=S_E.*S_E;
S_E=[];

S_S=fft(sen_sal);
S_S=abs(S_S);
S_S=S_S(1:n/2);  %% Densidad Espectral de energia de la señal de salida
DEE_S=S_S.*S_S;
S_S=[];

     % Ahora representamos las señales
     clg;
     subplot(221);
     plot(f_n,H_W,'g');grid % Dibujamos módulo lineal del filtro
     xlabel('Frecuencia (Hz)');ylabel('Módulo Lineal');
     title('MODULO DEL FILTRO');
     subplot(222);
     plot(f_n,H_F,'g');grid % Dibujamos la fase del filtro
     xlabel('Frecuencia (Hz)');ylabel('Radianes');
     title('FASE');
     subplot(212)
     plot(f_n,H_W_D,'g');grid % Dibujamos módulo en dB.
     xlabel('Frecuencia (Hz)');ylabel('Módulo en (dB)');
     title('MODULO DEL FILTRO');
     kiko=menu('  ','CONTINUAR');
     close

     clg;  % Señal de entrada y salida del filtro
     k=n;
     k1=0;
     k2=n/f_muest;
   opst=1;
   while (opst~=3)
     t=k1:(k2-k1)/k:k2-((k2-k1)/k); % Generamos vector de tiempos
     clg;
     subplot(211);
     plot(t,sen_ent(k1*f_muest+1:k2*f_muest),'g');grid ;
     xlabel('Segundos');ylabel('Amplitud');title('SEÑAL DE ENTRADA AL FILTRO');
     subplot(212);
     plot(t,sen_sal(k1*f_muest+1:k2*f_muest),'g');grid ;
     xlabel('Segundos');ylabel('Amplitud');title('SEÑAL DE SALIDA DEL FILTRO');
     kiko=menu('   ','CONTINUAR');

     opst=-1;
     while((opst<1)|(opst>3))
       clc
       opst=menu('Entra opcion:','Zoom de la señal.','Señal inicial.','Continuar.');
     end
     close;
     if (opst==1)
       ti=input('Entra el valor de tiempo inferior 0-tm (seg): ');
       ts=input('Entra el valor de tiempo superior 0-tm (seg): ');
       k=(ts-ti)*f_muest;
       k1=ti;
       k2=(k/f_muest)+k1;
```

```
            if((ti>ts)|(ti==ts)|(ti<0)|(ts<0)|((k1*f_muest+1)>n)|((k2*f_muest)>n))
               k=n;
               k1=0;
               k2=n/f_muest;
            end
    end

    if (opst==2)
      k=n;
      k1=0;
      k2=n/f_muest;
    end
  end

    clg; % Energia de la señal de entrada y salida
    p=n;
    p1=0;
    p2=frc;
  opst=1;
  while (opst~=3)
    f_n_e=p1:(p2-p1)/(p/2):p2-((p2-p1)/(p/2)); %% Vector de frecuencias para DEE
    clg;
    subplot(211);
    l1=((p1*n)/(2*frc))+1;
    l2=((p2*n)/(2*frc));
    plot(f_n_e,DEE_E(l1:l2),'g');grid % Dibujamos DEE de la señal de entrada al filtro
    xlabel('Frecuencia (Hz)');ylabel('U. energia/Hz');
    title('DENSIDAD ESPECTRAL DE ENERGIA - SEÑAL DE ENTRADA');
    subplot(212);
    plot(f_n_e,DEE_S(l1:l2),'g');grid % Dibujamos DEE de la señal de salida al filtro
    xlabel('Frecuencia (Hz)');ylabel('U. energia/Hz');
    title('DENSIDAD ESPECTRAL DE ENERGIA - SEÑAL DE SALIDA');
    kiko=menu('    ','CONTINUAR');

    opst=-1;
    while((opst<1)|(opst>3))
      clc
      opst=menu('Entra opcion:','Zoom de la señal.','Señal inicial.','Continuar.');
    end
    close
    if (opst==1)
      fi=input('Entra el valor de frecuencia inferior 0-fm/2 (Hz): ');
      fs=input('Entra el valor de frecuencia superior 0-fm/2 (Hz): ');
      p=((fs-fi)*n)/frc;
      p1=fi;
      p2=fs;

       z1=((p1*n)/(2*frc))+1;
       z2=((p2*n)/(2*frc));

       if((fi<0)|(fs<0)|(fi>fs)|(fi==fs)|(z1>n/2)|(z2>n/2))
          p=n;
          p1=0;
          p2=frc;
       end

    end
    if (opst==2)
      p=n;
      p1=0;
      p2=frc;
    end
  end
```

LISTADO DE RE_SEN.M

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%   Función que representa la señal de salida de un filtros dado y los  %%%
%%   parámetros del filtro por donde pasa la señal ..                    %%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%   Los parámetros de entrada son:                                      %%%
%%     * x,y :coeficientes del filtro.                                   %%%
%%     * sen_ent :señal de entrada.                                      %%%
%%     * f_muest :frecuencia de muestreo                                 %%%
%%   Devuelve la señal de salida - sen_sal                               %%%
%%             function  sen_sal=re_sen (x,y,f_muest,sen_ent)            %%%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
              function  sen_sal=re_sen (x,y,f_muest,sen_ent)

sen_sal=filter(x,y,sen_ent);   % Calculamos la señal de salida
delta=[1 zeros(size(1:199))]; %  Generamos delta de Dirac
h_n=filter(x,y,delta); %  Calculamos h[n]
delta=[];
H_W=fft(h_n); % Calculamos H(W)
h_n=[];
H_W=abs(H_W); % Calculamos módulo de H(W)
H_W=H_W(1:100); % Cogemos parte positiva del espectro
H_W_D=20*(log(H_W)/log(10)); %modulo en dB de Voltaje
frc=f_muest/2;
f_n=0:frc/100:frc-(frc/100); % Vector de frecuencias
ret_grup=grpdelay(x,y,100);
ret_grup=ret_grup/f_muest; % Retardo de grupo en segundos

n=length(sen_ent);
S_E=fft(sen_ent);
S_E=abs(S_E);
S_E=S_E(1:n/2);  %% Densidad Espectral de energia de la señal de entrada
DEE_E=S_E.*S_E;
S_E=[];

S_S=fft(sen_sal);
S_S=abs(S_S);
S_S=S_S(1:n/2);  %% Densidad Espectral de energia de la setal de salida
DEE_S=S_S.*S_S;
S_S=[];

% Ahora representamos las señales
clg;
subplot(221);
plot(f_n,H_W,'g');grid % Dibujamos módulo lineal del filtro
xlabel('Frecuencia (Hz)');ylabel('M³dulo Lineal');
title('MODULO DEL FILTRO');
subplot(222);
plot(f_n,ret_grup,'g');grid % Dibujamos retardo de grupo del filtro
xlabel('Frecuencia (Hz)');ylabel('Segundos');
title('RETARDO DE GRUPO');
subplot(212)
plot(f_n,H_W_D,'g');grid % Dibujamos módulo en dB.
xlabel('Frecuencia (Hz)');ylabel('M³dulo en (dB)');
title('MODULO DEL FILTRO');
kiko=menu('  ','CONTINUAR');
close
   clg; % Señal de entrada y salida del filtro
   k=n;
   k1=0;
   k2=n/f_muest;
 opst=1;
 while (opst~=3)
   t=k1:(k2-k1)/k:k2-((k2-k1)/k); % Generamos vector de tiempos
   clg;
   subplot(211);
    plot(t,sen_ent(k1*f_muest+1:k2*f_muest),'g');grid % Dibujamos señal de entrada al
filtro
   xlabel('Segundos');ylabel('Amplitud');
   title('SEÐAL DE ENTRADA AL FILTRO');
   subplot(212);
    plot(t,sen_sal(k1*f_muest+1:k2*f_muest),'g');grid % Dibujamos señal de salida del
filtro
   xlabel('Segundos');ylabel('Amplitud');title('SEÐAL DE SALIDA DEL FILTRO');
   kiko=menu('  ','CONTINUAR');
   opst=-1;
   while((opst<1)|(opst>3))
     clc
     opst=menu('Entra opci³n:','Zoom de la setal.','Setal inicial.','Continuar.');
   end
   close
   if (opst==1)
     ti=input('Entra el valor de tiempo inferior 0-tm (seg): ');
     ts=input('Entra el valor de tiempo superior 0-tm (seg): ');
     k=(ts-ti)*f_muest;
     k1=ti;
     k2=(k/f_muest)+k1;

                if((ti>ts)|(ti==ts)|(ti<0)|(ts<0)|((k1*f_muest+1)>n)|((k2*f_muest)>n))
```