Laboratorio de DSP y FPGA

Trabajo Práctico N° 3

Grupo 2

KAMMANN, Lucas Agustín

FARALL, Facundo David

DAVIDOV, Gonzalo Joaquín

TROZZO, Nicolás Rafael

Contenido

Ejercicio A

Diseño

En este ejercicio, se propone diseñar e implementar un filtro de segundo orden digital, utilizando un filtro de respuesta impulsiva infinita (IIR). El tipo de filtro a diseñar es un pasabanda, y la Tabla 1, muestra los parámetros generales que lo describen.

f_o	Δf	Q
$4kHz$	$4kHz$	1

Tabla 1: Especificaciones de filtro pasabanda de segundo orden

Para un filtro de segundo orden, en configuración pasabanda, la función transferencia analógica, en el dominio de Laplace, se expresa de la siguiente forma

$$H(s) = \frac{\frac{s}{\omega_o}}{\left(\frac{s}{\omega_o}\right)^2 + \frac{s}{Q \cdot \omega_o} + 1}$$

Se desea llevar la expresión anterior a una forma digital como la que se muestra a continuación, ya que es óptima para su implementación en un DSP. Los parámetros se pueden encontrar en la Tabla 2.

$$H(z) = \frac{\alpha \cdot (1 + \mu \cdot z^{-1} + \sigma \cdot z^{-2})}{\frac{1}{2} - \gamma \cdot z^{-1} + \beta \cdot z^{-2}}$$

α	0,09839579390151451
β	0,30320841219697103
γ	0,7001685777752599
μ	0
σ	-1

Tabla 2: Parámetros del IIR

La Ilustración 1 y la Ilustración 2, muestran los resultados teóricos a partir de una simulación del filtro digital IIR realizada con Python.

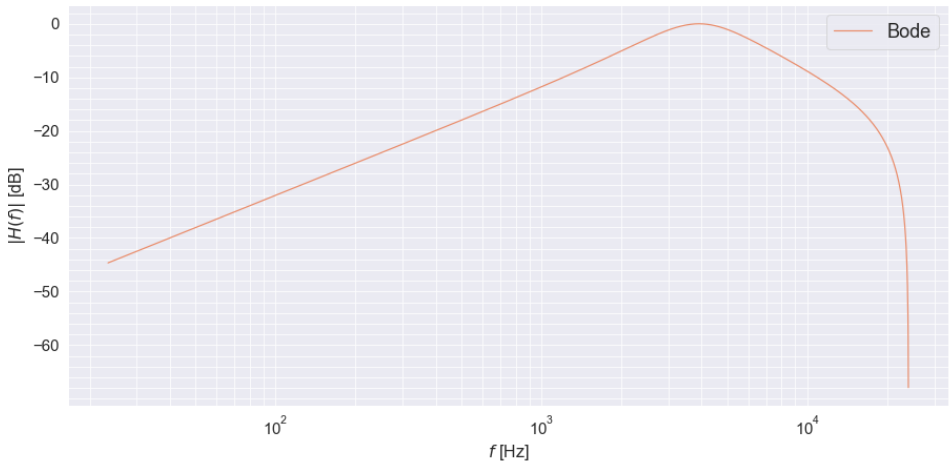

Ilustración 1: Diagrama de bode del IIR

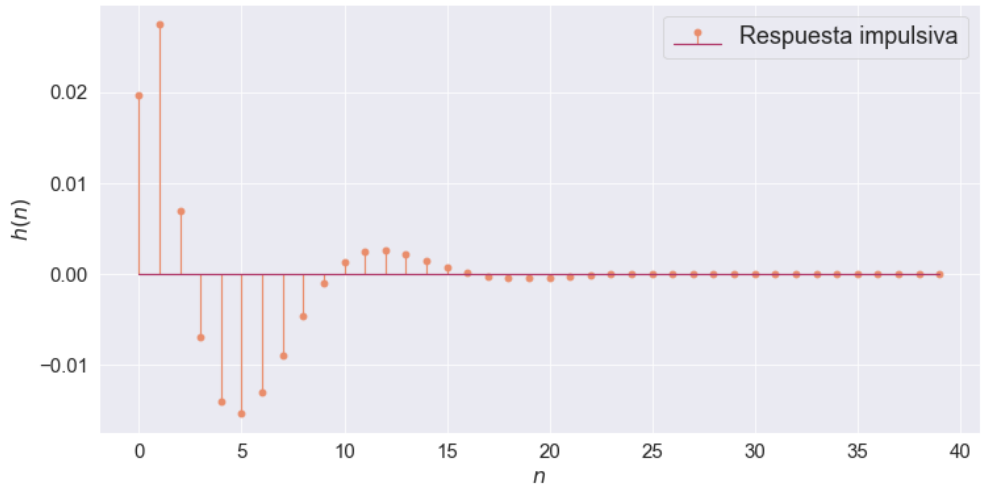

Ilustración 2: Respuesta al impulso del IIR

Implementación

En la Ilustración 3, se muestra el fragmento de código fuente correspondiente a la implementación de la subrutina **iir**. La subrutina implementa el algoritmo para ejecutar el procesamiento por muestras de una celda IIR de segundo orden. Vale destacar, que se agregó una instrucción NOP (no operation) al final de la subrutina para agregar una demora previa a guardar el contenido del acumulador A. De esta forma, nos aseguramos por software que el resultado esté listo en el momento adecuado, y no se produzcan problemas por dependencias.

```
;=====================================================================
; iir
;
; Executes a second order IIR using the direct form.
; It is expected that the first time the subroutine is called, X0 is loaded
; with the alpha coefficient. The subroutine expects that the X0 register always
; contains the alpha coefficient.
;
; @param    Y1 Input sample x(n)
; @param    R0 Pointer to the coefficients
; @param    R4 Pointer to the input samples
; @param    R5 Pointer to the output samples
; @return   X1 Output sample y(n)
;=====================================================================
iir     MPY X0,Y1,A     X:(R0)+,X0      Y:(R4)+,Y0      ; A = alfa * x(n)
        MAC X0,Y0,A     X:(R0)+,X0      Y:(R4),Y0       ; A = A + alfa * mu * x(n-1)
        MAC X0,Y0,A     X:(R0)+,X0      Y:(R5)+,Y0      ; A = A + alfa * sigma * x(n-2)
        MAC X0,Y0,A     X:(R0)+,X0      Y:(R5),Y0       ; A = A + gamma * y(n-1)
        MAC X0,Y0,A     X:(R0)+,X0      Y1,Y:(R4)       ; A = A - beta * y(n-2)
        NOP                                             ; Wait for the pipeline to finish
        MOVE            A,X1            A,Y:(R5)        ; yn = 2 * A (scaling mode is set)
        RTS
```

*Ilustración 3: Subrutina **IIR***

Resultados

Previo a la validación práctica, se realizó una simulación con el simulador provisto por Motorola para el DSP56307, utilizando el **iir_testbench**, utilizando como entrada una delta y observando en memoria la salida del filtro IIR. De esta forma, se verifica que se comporta como es esperado el filtro, por contrastación de la respuesta impulsiva.

Para la validación del filtro IIR diseñado, se compila, enlaza y programa el DSP56307 de Motorola con el código fuente **filtro.asm** que utiliza la subrutina **iir** desarrollada. Se conecta la entrada de audio de la placa de desarrollo del DSP56307 al generador, y su salida al osciloscopio, y se procede a medir la respuesta en frecuencia en un rango $f \in [1Hz, 24kHz]$. Es importante tener configurada la placa de desarrollo (por hardware) en una frecuencia de muestreo $f_s = 48kHz$.

En la Tabla 3 se muestra la contrastación entre la especificación, como objetivo buscado por el diseño teórico, y la medición como resultado práctico.

	Especificación	*Medición*	*Error porcentual*
f_o	$4kHz$	$3.879kHz$	3.025%
Δf	$4kHz$	$3.857kHz$	3.575%
Q	1	1.0057	0.57%

Tabla 3: Validación de especificaciones

Para finalizar, se ilustra a continuación la respuesta en frecuencia medida en la implementación práctica sobre el DSP56307.

Ilustración 4: Magnitud de la respuesta en frecuencia medida

El parámetro α de la Tabla 2 multiplica a todos los coeficientes, por lo que fue escalado para obtener una ganancia en banda pasante de 0dB.

Ejercicio B

Diseño

Para este ejercicio, el filtro a desarrollar debía contar con un orden de al menos 5, e implementado mediante celdas de segundo orden. Se eligió un filtro de tipo rechaza-banda, que cumpla con la plantilla presentada a continuación:

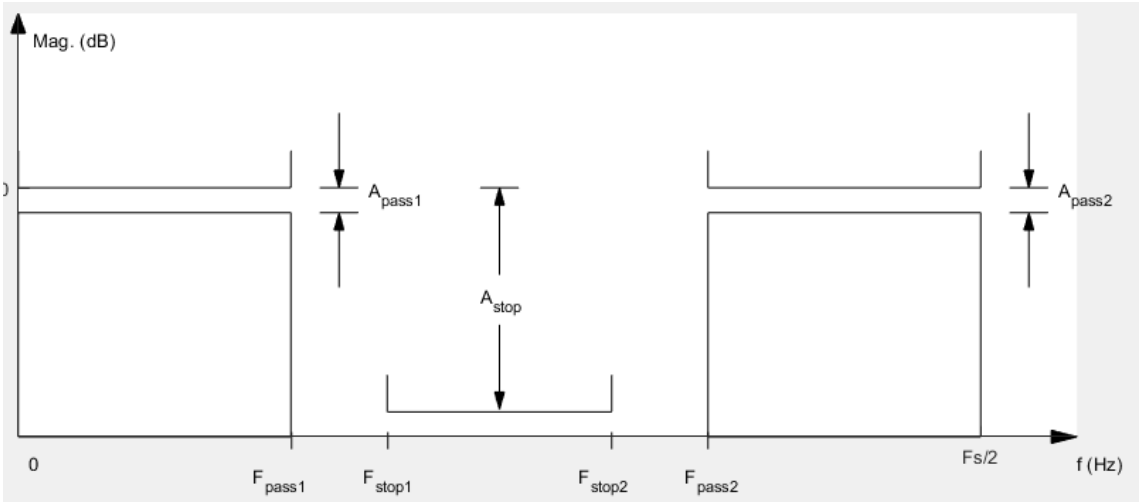

Ilustración 5: Plantilla de filtro rechaza-banda

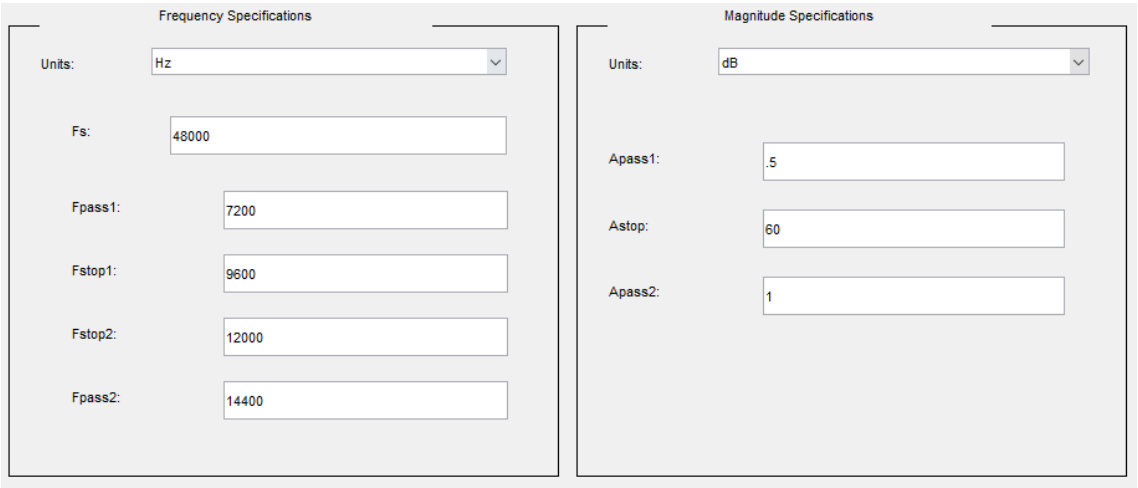

Ilustración 6: Especificaciones de la plantilla

Mediante el uso del paquete *filterDesigner* de MATLAB, se diseñó un filtro de orden 8, que cumpliera los requerimientos, y se obtuvieron los coeficientes de cada una de las etapas de segundo orden. Las mismas debían seguir el modelo de función transferencia indicado a continuación.

$$H(z) = \frac{\alpha \cdot (1 + \mu \cdot z^{-1} + \sigma \cdot z^{-2})}{\frac{1}{2} - \gamma \cdot z^{-1} + \beta \cdot z^{-2}}$$

La Tabla 4: Parámetros del IIR ilustra los valores obtenidos para los coeficientes de cada una de las 4 etapas resultantes.

$\boldsymbol{\alpha_1}$	$2.268282413482666015625/4$
$\boldsymbol{\beta_1}$	$0.47335147857666015625/2$

γ_1	0.9906291961669921875/2
μ_1	−0.49643802642822265625
σ_1	1
α_2	2.268282413482666015625/4
β_2	0.3928449153900146484375/2
γ_2	−0.599300384521484375/2
μ_2	−0.201916217803955078125
σ_2	1
α_3	0.2192783355712890625
β_3	0.8907105922698974609375/2
γ_3	1.0883333683013916015625/2
μ_3	−0.682704925537109375
σ_3	1
α_4	0.2192783355712890625
β_4	0.8733577728271484375/2
γ_4	−0.54757046699523925781 25/2
μ_4	0.0016105175018310546875
σ_4	1

Tabla 4: Parámetros del IIR

Las Ilustración 7: Respuesta en frecuencia del filtro rechaza-banda y Ilustración 8: Respuesta al impulso del filtro rechaza-banda presentan la respuesta en frecuencia y al impulso del filtro, respectivamente, obtenidas de la misma herramienta de diseño.

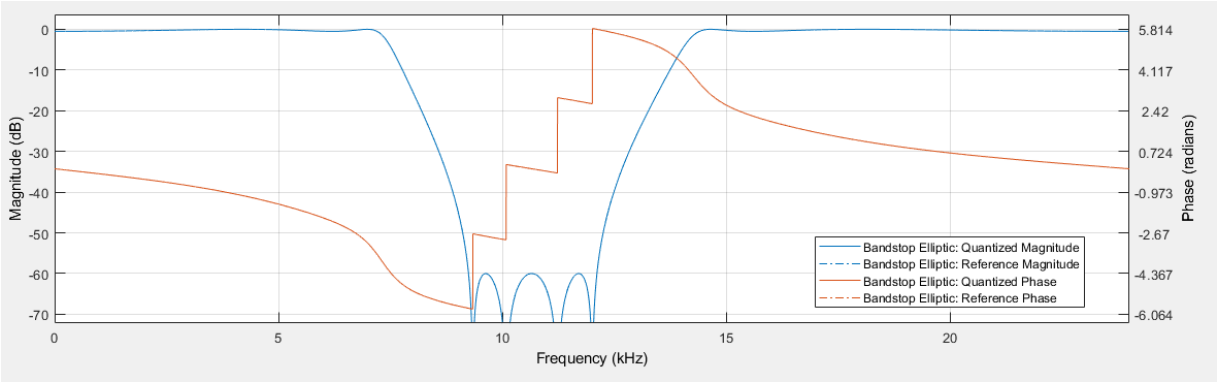

Ilustración 7: Respuesta en frecuencia del filtro rechaza-banda

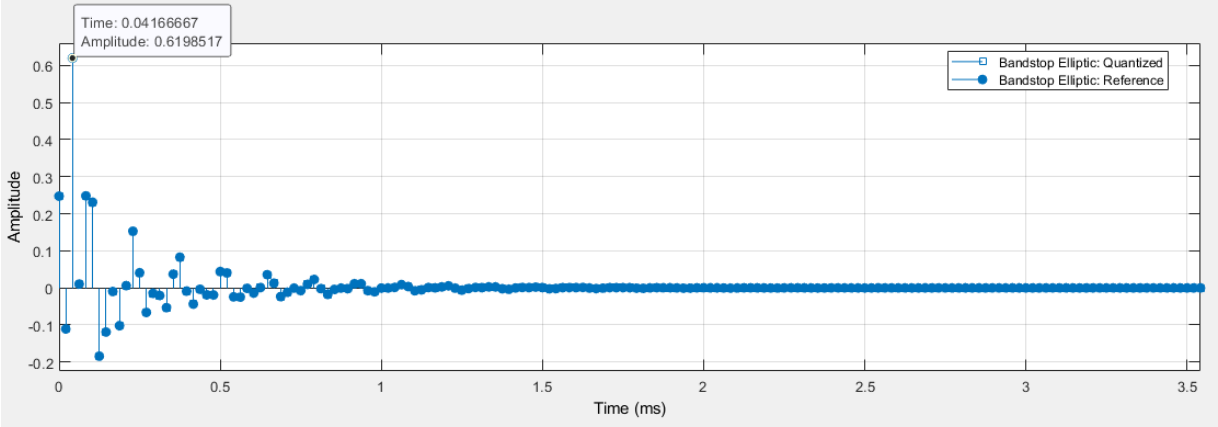

Ilustración 8: Respuesta al impulso del filtro rechaza-banda

Implementación

En la Ilustración 9: Fragmento de código que implementa el filtro IIR como cascada de secciones de segundo orden.Ilustración 3, se muestra el fragmento de código fuente correspondiente a la implementación de la macro **iir**, la cual para esta implementación no es sencillamente una subrutina, con el fin de recibir como parámetro la cantidad de celdas que el filtro diseñado deberá ciclar en cada pasada. La subrutina **iir_casc** implementa el algoritmo para ejecutar el procesamiento por muestras de una cascada de celdas IIR de segundo orden.

```
;====================================================================
; iir
;
; Executes a second order IIR using the direct form.
; It is expected that the first time the subroutine is called, X0 is loaded
; with the alpha coefficient. The subroutine expects that the X0 register always
; contains the alpha coefficient of the first second order section, and
; X1 is loaded iwth the beta coefficient of the same section.
;
; @param    Y1 Input sample x(n)
; @param    R0 Pointer to the coefficients
; @param    R4 Pointer to the previous samples to be used by the filter
; @return   Y1 Output sample y(n)
;====================================================================
iir     MACRO           nsec
iir_casc
        ORI             #$08,MR                         ; Setting scaling mode for doubling the output
        MOVE            (R4)+                           ; Point to next xbuf (buffer with previous values) entry

        DO #nsec,sectn
        MPY X0,Y1,A     X:(R0)+,X0      Y:(R4)+,Y0      ; A = alpha_i * x_i(n)
        MAC X0,Y0,A     X:(R0)+,X0      Y:(R4)+N4,Y0    ; A = A + alpha_i * sigma_i * x_i(n-2)
        MAC X0,Y0,A     X:(R0)+,X0      Y:(R4)+,Y0      ; A = A + alpha_i * mu_i * x_i(n-1)
        MAC X0,Y0,A                     Y:(R4)-N4,Y0    ; A = A + gamma_i * y_i(n-1)
        MAC -X1,Y0,A    X:(R0)+,X1      Y1,Y:(R4)+N4    ; A = A - beta_i * y_i(n-2) and save x(n)
        MOVE A,Y1        X:(R0)+,X0                      ; y_i(n) = 2 * A (scaling mode is set)
sectn                                                   ; X1 = beta_i+1 and X0 = alpha_i+1
                                                        ; Output y(n) = Y1
        MOVE            #(nsec+1)*2-1,M4                ; Filter order+1
        NOP
        MOVE                            Y1,Y:(R4)+N4    ; Save y(n)
        MOVE            #1,M4

        RTS

        ENDM
```

Ilustración 9: Fragmento de código que implementa el filtro IIR como cascada de secciones de segundo orden.

Resultados

Al igual que en el ejercicio anterior, el simulador provisto por Motorola para el DSP56307, fue utilizando en el código **iir_testbench**, siendo la entrada una delta y observando en memoria la salida del filtro IIR. De esta forma, se verifica que se comporta como es esperado el filtro, por contrastación de la respuesta impulsiva.

Para la validación del filtro IIR diseñado, se compila, enlaza y programa el DSP56307 de Motorola con el código fuente **filtro.asm** que utiliza la subrutina **iir_casc** desarrollada. Se conecta la entrada de audio de la placa de desarrollo del DSP56307 al generador, y su salida al osciloscopio, y se procede a medir la respuesta en frecuencia en un rango $f \in [1Hz, 24kHz]$. Es importante tener configurada la placa de desarrollo (por hardware) en una frecuencia de muestreo $f_s = 48kHz$.

En la Tabla 3 se muestra la contrastación entre la especificación, como objetivo buscado por el diseño teórico, y la medición como resultado práctico.

	Especificación	*Medición*	*Error porcentual*
f_{pass1}	$7.2kHz$	$7.2344kHz$	0.47%
f_{stop1}	$9.6kHz$	$9.2698kHz$	3.4%
f_{stop2}	$12kHz$	$12.059kHz$	0.5%
f_{spass}	$14.4kHz$	$14.306kHz$	0.65%

Tabla 5: Validación de especificaciones

Para finalizar, se ilustra a continuación la respuesta en frecuencia medida en la implementación práctica sobre el DSP56307.

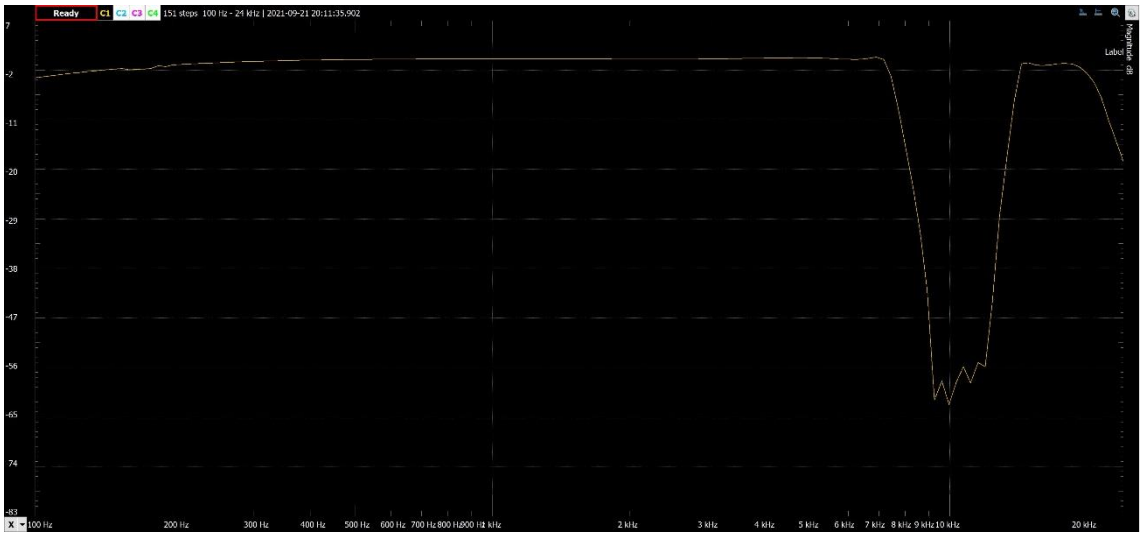

Ilustración 10: Medición de la respuesta en frecuencia del filtro rechaza-banda

Se ajustó la ganancia multiplicando una de las etapas por 2.54 para lograr una ganancia de 0dB en banda pasante.

No se cumple la plantilla en dos lugares debido a las siguientes razones:

- La atenuación en la banda de rechazo es levemente menor a 60dB. Esto se debe al piso de ruido de la medición. Se utilizó una señal de entrada de 300mV de amplitud para evitar saturaciones en las etapas intermedias, por lo que en la banda de rechazo la señal atenuada 60dB sería de $300\mu V$, lo cual está por debajo del piso de ruido de medición.
- En baja frecuencia se observa una atenuación que no debería estar, de -3dB a 100Hz. Esto se debe a la respuesta en frecuencia del DSP, se muestra en la Ilustración 11: Medición de la respuesta en frecuencia del DSP la medición de la respuesta en frecuencia de un programa que simplemente pone a la salida las mediciones de la entrada (se lo podría llamar un “caño”).

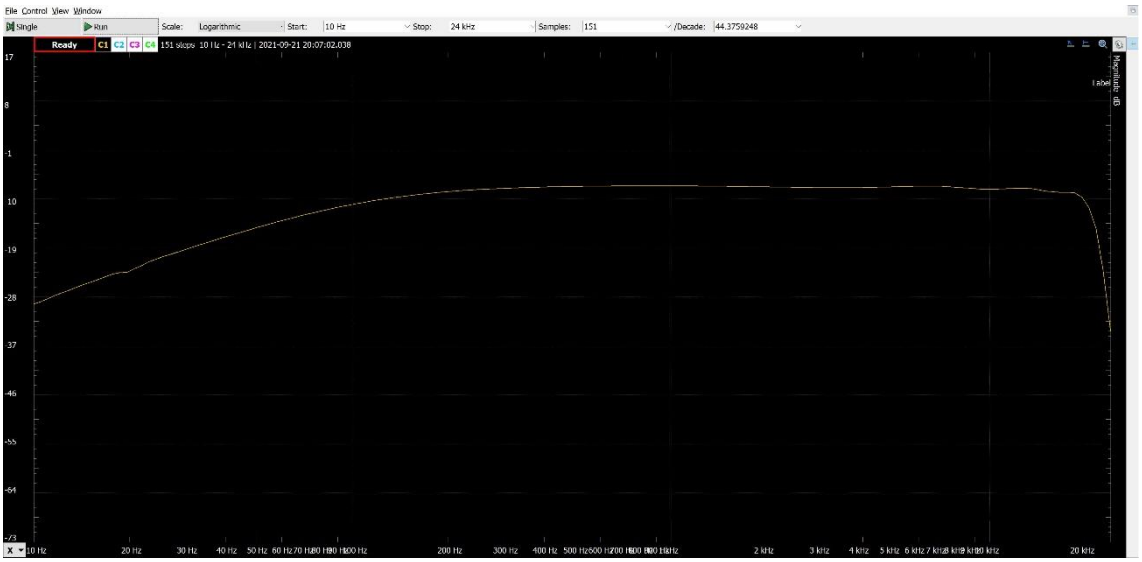

Ilustración 11: Medición de la respuesta en frecuencia del DSP

Ejercicio C

El rango dinámico del filtro se define como:

$$RD_{dB} = 20 \log\left(\frac{V_{imax}}{V_{imin}}\right)$$

Este se ve limitado por la cantidad finita de bits disponible, es decir, por la cuantización realizada en el procesamiento digital. La cuantización afecta el rango dinámico en dos puntos [1]:

- Digitalización de la señal de entrada. No es analizada ya que forma parte de la conversión AD.
- Redondeo (o truncado) del producto de multiplicaciones.

La segunda es la principal fuente de ruido, llamado “round-off noise”, se encuentra en varios puntos a lo largo del filtro, luego de cada multiplicación, y limita el rango dinámico del filtro, en particular limita la mínima amplitud de entrada.

Por lo tanto, tanto el numerador como el denominador de la expresión de RD tienen limitaciones debido a la cantidad de bits:

- V_{imax}: Debe ser tal que no haya overflow en ninguna de las operaciones intermedias.
- V_{imin}: Debe ser tal que el “round-off noise” nos permita operar. Se debe definir un SNR deseado.

En relación a la limitación del overflow para V_{imax} es que el DSP tiene bits extra en los acumuladores A y B, con lo que este efecto se ve minimizado, aunque no desaparece.

La ganancia y el Q de las etapas intermedias afectan al rango dinámico, ya que si las ganancias son muy desparejas y hay Q’s muy altos, la señal sufrirá muchos cambios innecesarios, pudiendo llegar en cualquier punto del filtro tanto al límite de overflow como al límite de SNR por hacerse muy pequeña. Por lo tanto, así como en los filtros analógicos, se deben evitar los altos Q y tener ganancias parejas.

Referencia: [1] “On the Interaction of Roundoff Noise and Dynamic Range in Digital Filters”. Jackson, 1970.

Ejercicio D

La cuantización de los coeficientes del IIR puede afectar en que, por ejemplo, los polos y los ceros de la respuesta en frecuencia cambiarán al elegir los niveles de cuantización. Se debe tener esto presente ya que, a su vez, podría afectar a la estabilidad del filtro si justo uno de estos cambios numéricos hace que un polo de la respuesta en frecuencia quede ubicado en el semiplano derecho. Otro parámetro que se verá afectado es la ganancia del filtro.

Ejercicio E

Si se quiere implementar el filtro del Ejercicio 1.b, pero ahora utilizando un FIR en lugar de un IIR se va a necesitar realizar un filtro de mayor orden. Para poder lograr esto se necesitará más memoria ya que la respuesta impulsiva del filtro será de mayor orden y, por lo tanto, habrá más cantidad de muestras de entrada.

Además, al usar un FIR se tendrá un mayor costo computacional para realizar todas las operaciones necesarias para el orden del filtro, que, cuando se implementa con un IIR, se hace uso de las conexiones en cascada de segundo orden. Por esto último se puede agregar que se necesitará un hardware más veloz para realizar las operaciones con la misma velocidad que para un IIR.