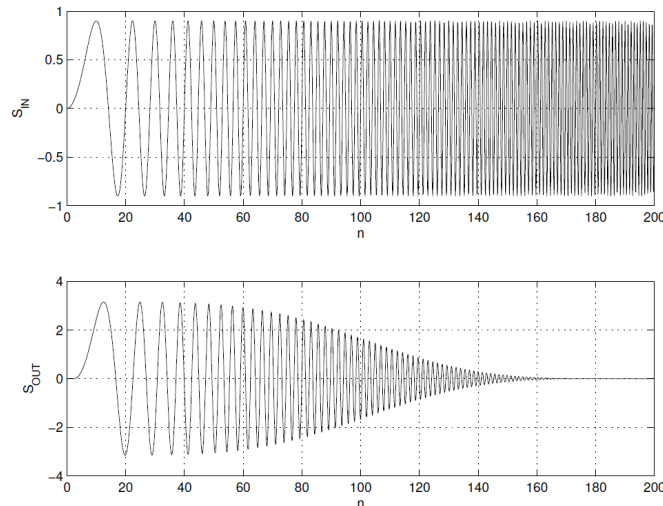
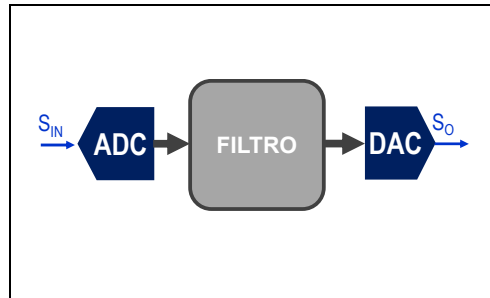


## EJERCICIOS: MODELADO DE PRECISIÓN FINITA

- Indique la resolución y el rango de números representables con los siguientes formatos numéricos (U indica codificación sin signo, S con signo en complemento a dos y  $[n,b]$  indica que se dispone de  $n$  bits de los cuales los  $b$  menos significativos son fraccionales):
  - U[8,7]
  - S[8,7]
  - U[8,8]
  - S[8,8]
  - U[10,7]
  - S[10,7]
- ¿Qué formato numérico se necesita para representar la señal  $x(n)=10\sin(2\pi n f_0/f_s)$  con una resolución de al menos 0.01?
- Codifique el número 4 y el -4 con los siguientes formatos numéricos:
  - S[4,0]
  - S[6,0]
  - S[8,0]
- Obtenga el valor entero y el código binario que se almacenaría en un registro al codificar los siguientes números con los formatos indicados:
  - 0.625 con formato S[4,3]
  - 3.25 con formato S[5,2]
- Si un registro de 8 bits tiene almacenado el número binario "10011001", ¿a qué número real corresponde si se asume que se ha codificado con los siguientes formatos numéricos?:
  - U[8,7]
  - S[8,7]
  - U[8,4]
  - S[8,4]
- Indique qué formato numérico se debe tener a las salidas de las siguientes operaciones para que no se pierda precisión y el formato esté ajustado al rango de la operación.
  - $S1=A1+A2$ , siendo el formato de  $A1$  S[8,7] y de  $A2$  S[6,3]
  - $S2=A1+A2+A3+A4+A5$ , con  $A_i$  S[8,7]
  - $S3=A1+A2+A3+A4+A5$ , con  $A_i$  U[8,7]
  - $M=A1*A2$ , con  $A1$  S[8,7] y  $A2$  S[6,5]
  - $MA=A1*A2+A3$ , con  $A_i$  S[8,7]
- Se requiere realizar la suma de 100 datos codificados con formato S[12,11]. ¿Qué formato debe tener el valor resultante para que se pueda codificar sin que se produzca desbordamiento ni pérdida de precisión?
- Se requiere realizar la operación multiplica y acumula para evaluar la suma de 500 productos ( $Y=\sum A_i \cdot B_i$ ). Si los formatos numéricos de  $A_i$  y  $B_i$  son S[8,7] y S[10,7], respectivamente, ¿cuál es el formato numérico de la salida (Y) para que se calcule sin pérdida de precisión en las operaciones intermedias?

9. Se desea filtrar digitalmente una señal. Se ha realizado un modelo del filtro y su respuesta a una entrada tipo “chirp” (señal sinusoidal cuya frecuencia de oscilación es creciente con el tiempo) se muestra en la figura. Se dispone de conversores ADC y DAC, ambos de 14 bits. ¿Qué formatos numéricos se requieren para representar las señales de entrada y salida del filtro?



10. El cálculo de la inversa de un valor  $x$ ,  $f=1/x$ , se requiere en muchos algoritmos DSP.
- Suponga que se quiere calcular la inversa de un número para valores de  $x$  limitados entre  $]0.25,1[$ . ¿Qué formatos numéricos se requieren para la entrada y salida del operador si se necesita representar sus valores con una resolución de al menos 0.01?
  - ¿Cómo se tendría que elegir el formato numérico de la salida del operador  $1/x$  si la entrada  $x$  no estuviese limitada (como en el caso anterior) y tuviera un formato  $[8,8]$  sin signo.
11. A continuación, se muestra el código Verilog de un multiplicador con salida truncada y un sencillo banco de pruebas para su verificación. Indique cómo se modela con Matlab el comportamiento de dicho multiplicador y verifique que las salidas del banco de prueba son correctas.

```

module mult_t(
input signed [7:0] A,B,
output signed [9:0] M
);

```

```

wire signed [15:0] Mc;

```

```

assign Mc = A*B;
assign M = Mc[15:6];

```

```

endmodule

```

/A	-50	56	-32	-50
/B	-97	8	6	-97
/M	75	7	-3	75

```

module tb_mult_t();
reg signed [7:0] A, B;
wire signed [9:0] M;
parameter PER = 100;

```

```

mult_t uut(.A(A),.B(B),.M(M));

```

```

initial begin

```

```

    A = 56; B = 8;

```

```

    #PER;

```

```

    A = -32; B = 6;

```

```

    #PER;

```

```

    A = -50; B = -97;

```

```

    #PER $stop;

```

```

end

```

```

endmodule

```

12. Suponga que en el ejercicio anterior las entradas tienen formato [8,7] con signo.
- ¿A qué valores reales equivalen los enteros utilizados para las entradas A y B en el banco de pruebas?
  - ¿Cuál es el formato numérico de la salida M?
  - ¿Cómo se modela directamente con Matlab (sin pasar a enteros) la operación que obtiene M, teniendo en cuenta que entran A y B con el formato [8,7]? Compruebe que ha contestado correctamente calculando los valores de M obtenidos con las entradas A y B de la sección a). Si después obtiene el equivalente entero del valor obtenido de M, éste debe ser igual al mostrado en el banco de pruebas del ejercicio anterior.
13. A continuación, se muestra el código Verilog de un circuito aritmético que realiza la operación  $MS=A*B+C$  con datos de entrada de 8 bits con signo codificados en complemento a dos y un sencillo banco de pruebas.
- ¿Está correctamente dimensionado el tamaño de la salida MS para que no se produzca desbordamiento?
  - ¿Es válido este circuito para realizar la operación  $A*B+C$  con entradas codificadas con formatos numéricos  $S[8,x]$ , donde “x” puede ser cualquier entero entre 0 y 8? Indique para qué formatos de A, B y C el circuito es válido. Escriba un sencillo banco de pruebas para comprobar que el formato elegido es el correcto.
  - Modifique el modelo Verilog del operador  $MS=A*B+C$  para que sea válido entradas codificadas con el formato  $S[8,7]$ . Compruebe que el resultado es correcto simulando el banco de pruebas y comparando el resultado con el obtenido al ejecutar el operador en Matlab.

```
module mult_add_1(
input signed [7:0] A,B,C,
output signed [15:0] MS
);

assign MS = A*B+C;
endmodule
```

/A	-50	-128	-32	-50
/B	-97	-128	6	-97
/C	123	127	-87	123
/MS	4973	16511	-279	4973

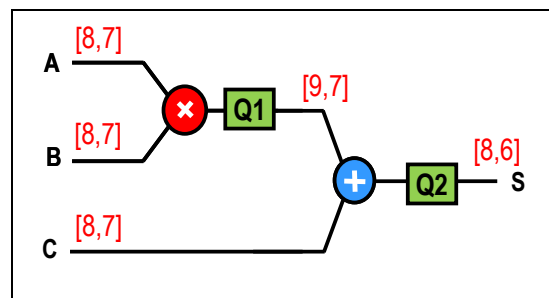
```
module tb_mult_add_1;
reg signed [7:0] A,B,C;
wire signed [15:0] MS;

parameter PER = 100;

mult_add_1 uut (.A(A),.B(B),.C(C),.MS(MS));

initial begin
    A = -128; B = -128; C = 127;
    #PER;
    A = -32; B = 6; C = -87;
    #PER;
    A = -50; B = -97; C = 123;
    #PER $stop;
end
endmodule
```

14. Realice el modelo de precisión finita del operador  $S=A*B+C$ , para los formatos numéricos con signo indicados en la figura. Q1 y Q2 indican los truncados a la salida del multiplicador y del sumador, respectivamente.
- Utilizando la función “floor” de Matlab
  - Utilizando el objeto “quantize” de Matlab
  - Utilizando Simulink



Compruebe el correcto funcionamiento del modelo para varios valores de la entrada.

15. Con este ejercicio se pretende explorar el efecto que tiene la cuantificación de los coeficientes de un filtro FIR en su respuesta en frecuencia.
- Obtenga los coeficientes de un filtro FIR de orden 40 y frecuencia de corte  $(f_s/2)/8$  con la función `fir1` ( $h=fir1(40,1/8)$ ). Visualice su respuesta en frecuencia utilizando la función `freqz` y el diagrama polos/ceros con la función `zplane`.
  - Compruebe cuáles son los valores máximo y mínimo que toman los coeficientes y elija un formato de precisión finita para representar dichos coeficientes con 16 bits.
    - ¿Cuál es el rango de valores representables con el formato elegido?
    - ¿Se está aprovechado todo el rango del formato numérico elegido al cuantificar los coeficientes?
  - Escale los coeficientes para que se aproveche completamente el rango elegido.
    - ¿Cómo cambia la respuesta en frecuencia del filtro el escalado de los coeficientes?
    - ¿Cómo afecta dicho cambio a las señales filtradas? Compruébelo filtrando una señal sinusoidal con frecuencia menor que la frecuencia de corte del filtro con el filtro original y con el que tiene los coeficientes escalados.
  - Cuantifique de nuevo los coeficientes del filtro reduciendo de uno en uno el número de bits fraccionales. Para cada cuantificación visualice la respuesta en frecuencia y el diagrama de polos y ceros.
    - ¿Cómo afecta la cuantificación de los coeficientes de un filtro en su respuesta en frecuencia?
    - ¿Por qué se produce ese efecto?
    - Indique cual es el formato numérico con el que se alcanza una atenuación de al menos 50dBs con el menor tamaño de palabra.
16. Con este ejercicio se pretende explorar cómo afecta la cuantificación de los operadores aritméticos a las señales procesadas por un filtro FIR. Para realizar este ejercicio utilice los siguientes datos:
- Coeficientes con formato [10,9]:
 
$$h=fir1(40,1/8)*8;$$

$$h=round(h*2^9)*2^{-9};$$
  - Señal de entrada sinusoidal con formato [12,11]:
 
$$Lx=1000;$$

$$n=0:Lx-1;$$

$$x=(1-2^{-13})*sin(2*pi*n*1/50);$$

$$x=floor(x*2^{11})*2^{-11};$$

A continuación, se muestra la función de Matlab `filtrafir` que obtiene la salida de un filtro FIR. También se muestra el esquema de un circuito multiplicador acumulador que se utilizará en los siguientes temas para implementar con una arquitectura secuencial un filtro FIR. El circuito multiplicador-acumulador computa en cada ciclo el producto entre un dato y un coeficiente y lo suma al valor acumulado, por tanto, computará la sentencia " $acc(i)=acc(i)+x_n(i-j+1)*h(j)$ " del código de la función `filtrafir`.

Función <code>filtrafir</code>	Circuito multiplicador-acumulador
<pre>function [y] = filtrafir(x,h) % Filtra x con un filtro FIR con coefs. h % x: vector de entrada % h: coeficientes del filtro FIR % y: salida del filtro  Lh=length(h); % Num. coeficientes xn=zeros(1,Lh-1)x; Lxn=length(xn); acc=zeros(1,Lxn);</pre>	

```

for i=Lh:Lxn
    for j=1:(Lh)
        acc(i)=acc(i)+xn(i-j+1)*h(j);
    end
end
y=acc(Lh:end);

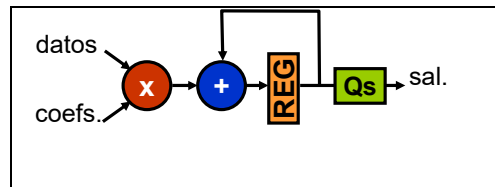
```

a) Computación sin pérdida de precisión. Que no haya pérdida de precisión en las operaciones intermedias quiere decir que el resultado al computarlo con precisión finita sea exactamente el mismo que si se hubiera computado con toda la precisión.

a.1) Filtre la señal sinusoidal indicada anteriormente con la función *filtrafir* y visualice la señal de salida ¿Qué formatos de datos se requiere a la salida del multiplicador y en el acumulador para que no haya pérdida de precisión en las operaciones intermedias del filtro?

a.2) Compruebe que ha contestado correctamente incluyendo la precisión finita en la función *filtrafir* y comparando el resultado con la función original, sin precisión finita.

b) Computación sin pérdida de precisión en las operaciones intermedias y salida truncada. Se va a suponer que una vez se obtiene un valor de la señal filtrada éste se trunca quedándonos sólo con 12 bits.



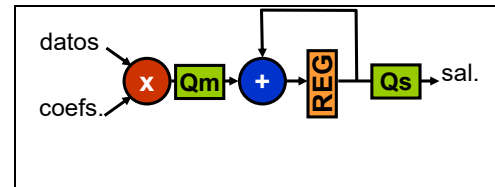
b.1) Indique el formato numérico con el que se debe representar la salida.

b.2) A partir de la función de Matlab *filtrafir* escriba otra (*filtrafir\_q1*) donde se incluya el efecto del truncado de la salida con el formato numérico indicado anteriormente.

b.3) Calcule con Matlab el error máximo cometido al truncar la salida del filtro introduciendo la misma señal a ambas funciones y restando las salidas.

b.4) ¿Cuántos bits fraccionarios se necesitan para codificar el error máximo calculado?

c) Computación con pérdida de precisión en las operaciones intermedias y salida truncada. En este caso, como en el anterior, la salida de la señal filtrada se trunca quedándonos sólo con 12 bits.

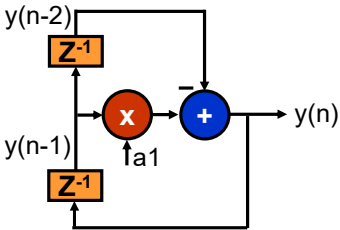


Ahora, además, el tamaño de la salida del multiplicador se limita (trunca). Al no disponer de todos los bits del multiplicador, éste introduce errores de cuantificación. Sin embargo, esto provoca que el acumulador disminuya de tamaño.

c.1) A partir de la función *filtrafir\_q1* escriba una nueva función, *filtrafir\_q2*, en la que modele el truncado de la salida del multiplicador.

c.2) Utilice Matlab para encontrar el tamaño mínimo del acumulador con el que se obtiene un error máximo como el obtenido en b.3). ¿Cuántos bits tendrá el acumulador en ese caso?

17. Una señal sinusoidal se puede generar con un filtro recursivo de segundo orden que implemente la siguiente ecuación en diferencias:  $y(n)=a1*y(n-1)-y(n-2)$  con  $y(-2)=bo$ , donde  $bo=A*\sin(2*pi*fo/fs)$  y  $a1=2*cos(2*pi*fo/fs)$ , siendo  $A$  y  $fo$  la amplitud y frecuencia de oscilación y  $fs$  la frecuencia de muestreo. A continuación, se muestra una función de Matlab que modela la ecuación anterior y el diagrama de bloques del oscilador.

Función de Matlab	Diagrama de bloques
<pre> function [y,p]=oscilador(N,a1,bo)  % [y,p]=Oscilador(N,a1,bo) % y: salida del oscilador % p: salida del producto % y(n)=a1*y(n-1)-y(n-2) % con y(-2)=bo % % N: numero de muestras  y(2)=0; y(1)=bo;  for n=(3:N)     p(n)=a1*y(n-1);     y(n)= p(n) - y(n-2); end  subplot(2,1,1),plot(y), grid title('Salida oscilador') subplot(2,1,2),plot(p),grid title('Salida del multiplicador') </pre>	

- Calcule los valores de los coeficientes  $a1$  y  $bo$  para que el oscilador genere una señal de 1 MHz con una frecuencia de muestreo de 100 MHz y compruebe que el modelo funciona correctamente. Suponga que la amplitud  $A$  es 0.98.
  - Determine el formato numérico de los coeficientes para que la frecuencia de oscilación del oscilador con precisión finita tenga un error menor a 1 Hz respecto al oscilador ideal. Tenga en cuenta que  $fo = \text{atan}(2 \cdot bo / A \cdot a1) \cdot fs / (2 \cdot \pi)$ .
  - Visualice las formas de onda a la salida del oscilador y del multiplicador. ¿Cuántos bits enteros se necesitan para cuantificar la salida del oscilador y la salida del multiplicador?
  - ¿Se puede realizar la computación del filtro recursivo con precisión finita y sin pérdida de precisión en las operaciones intermedias? Razone la respuesta.
  - A partir de la función `oscilador`, escriba una nueva función, `oscilador_q`, que incluya el efecto de truncar la salida del multiplicador a  $N_{fm}$  bits fraccionales. Utilice el modelo para determinar cuál es el tamaño mínimo del sumador con el que se consigue una señal sinusoidal con una precisión de al menos  $2^{-13}$ . Realice para ello una comparación con el modelo ideal introduciendo los coeficientes cuantificados y calcule el error como la diferencia entre la salida del modelo de precisión finita y la del ideal.
18. Supóngase que un operador genera una salida  $A_{out}$  con formato numérico [25,15] con signo. La salida de dicho operador se quiere conectar a otro operador cuya entrada  $B_{in}$  dispone de 16 bits con formato fraccionario [16,15].
- ¿Qué 16 bits de  $A_{out}$  hay que conectar al operando de entrada  $B_{in}$ ?
  - Indique cómo se modela con Matlab la obtención del operando de entrada  $B_{in}$  con formato [16,15] a partir de la salida  $A_{out}$ .