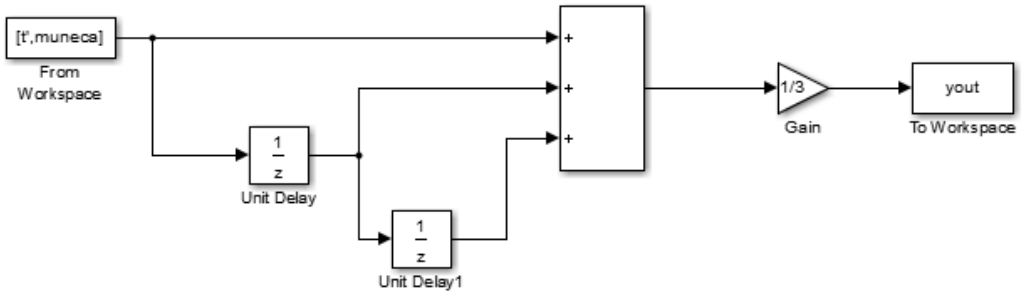
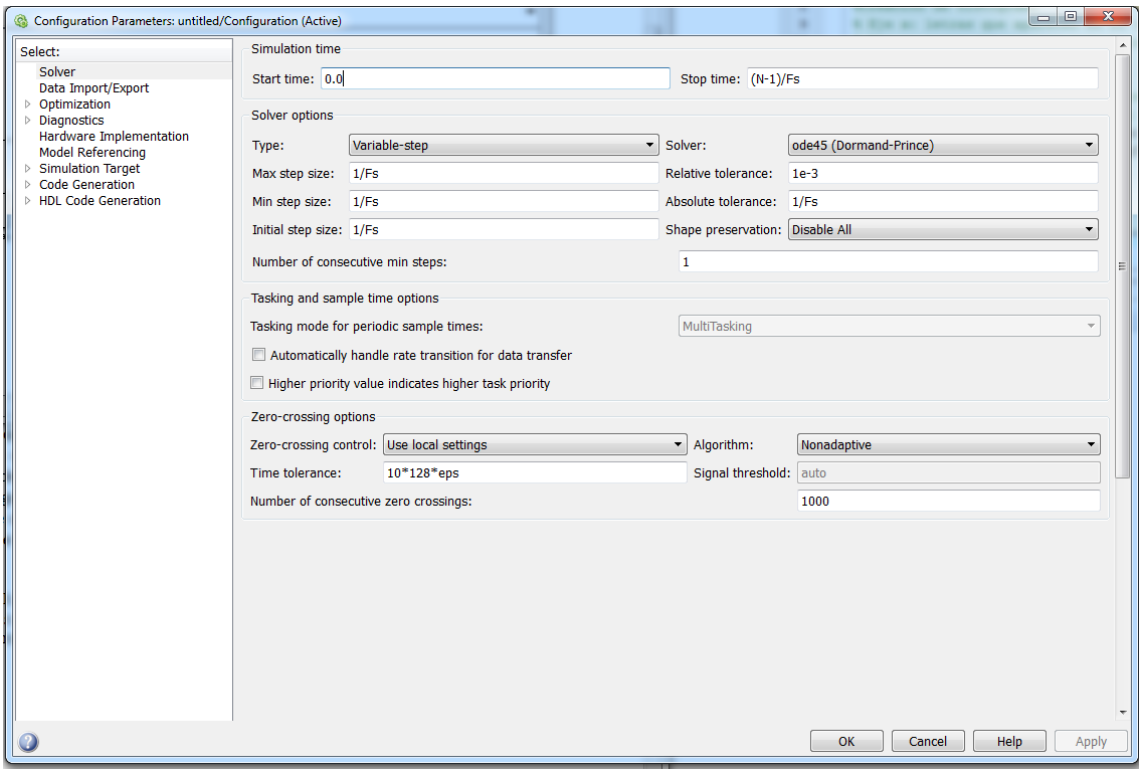
Práctica 1. Simulación de sistemas con Simulink. Cuantización de señales.

1.Filtro FIR.

El primer sistema basado en diagramas de bloques es FIR. La realización es la siguiente



Antes de ejecutar debemos de establecer los parámetros de la siguiente manera

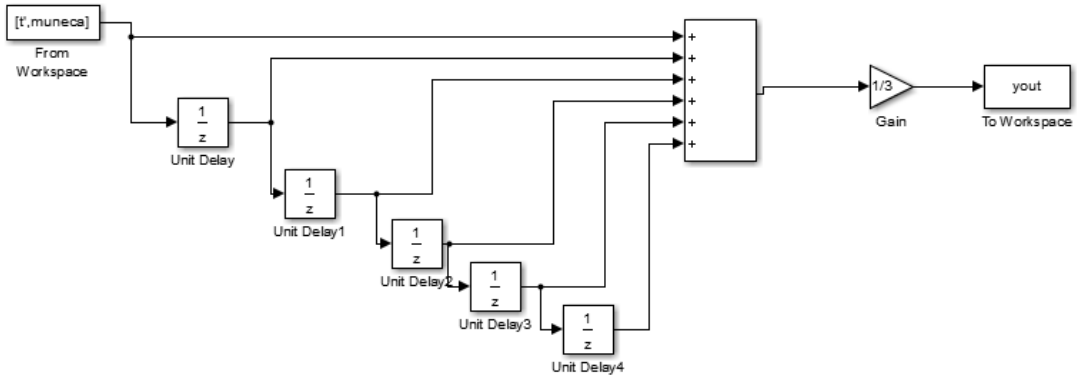


Al ejecutar el sistema nos devuelve una variable de tipo *timeseries* llamada yout. De dicha señal extraemos los datos y comprobamos de manera auditiva los efectos de la transformación efectuada.

```
>> salida1=yout.get('Data');
>> sound(salida1/max(salida1));
```

También podríamos dibujar la grafica resultante simplemente escribiendo *plot(yout)*.

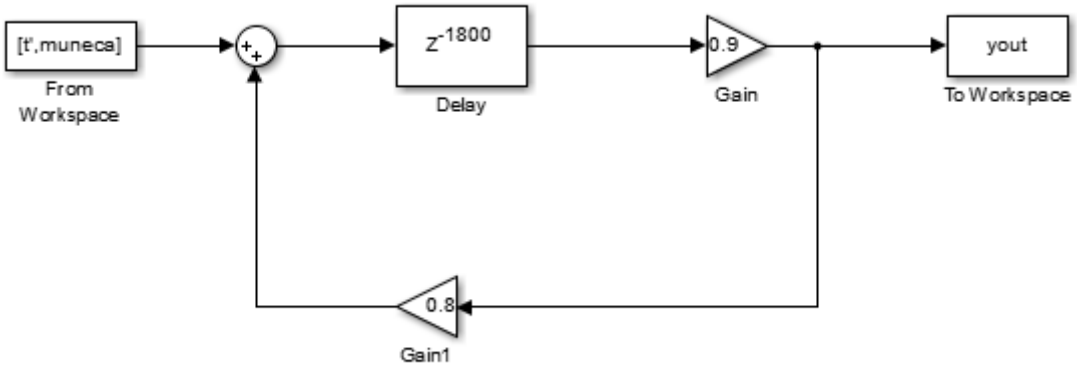
Lo siguiente a hacer es establecer más retardos a la señal



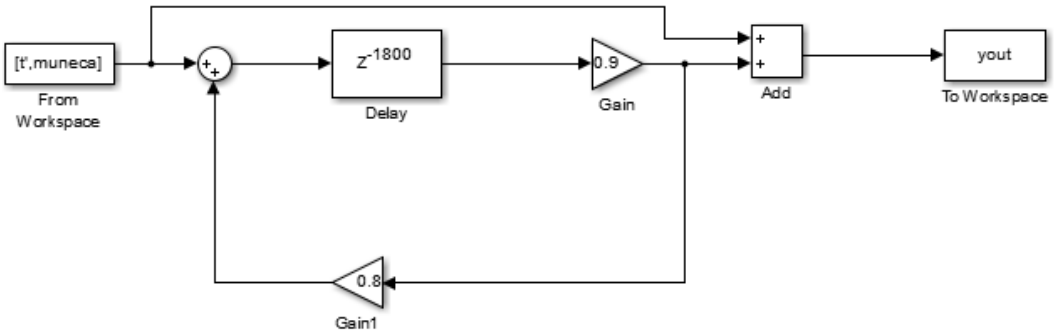

```
>> salida2=yout.get('Data');
>> sound(salida2/max(salida2))
```

Hacemos igual que antes y reproducimos el sonido resultante. Dicho sonido se nota más distorsionado, con menos nitidez que el original.

Ahora probamos a crear un efecto sobre el sonido, la reverberación, que es la reflexión que se produce al chocar las ondas sobre el material, haciendo que llegue con retardo. Para ello creamos el sistema



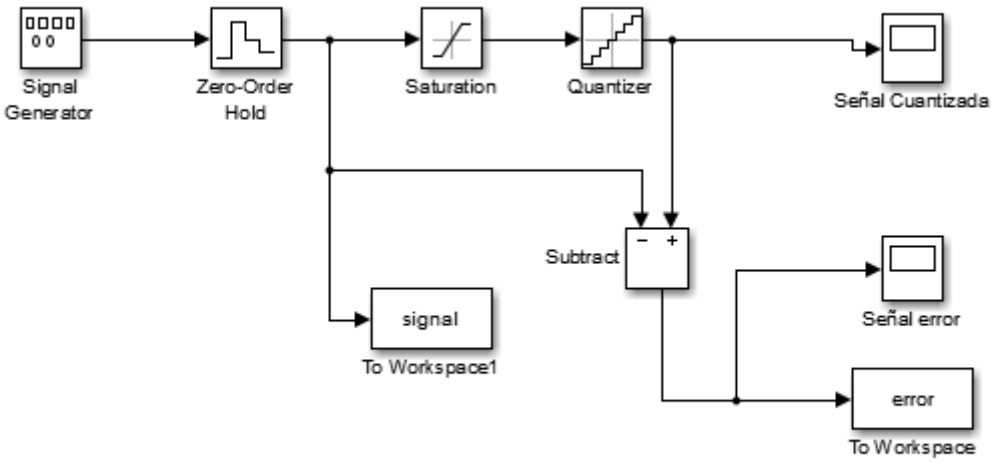
Ahora podemos comprobar el sonido como hemos hecho anteriormente y vemos que no se escucha el sonido original completo sino el sonido desplazado en tiempo. Para mostrar el efecto acoplamos la señal original y la señal reverberación. Comprobando con el sonido que existe la señal y



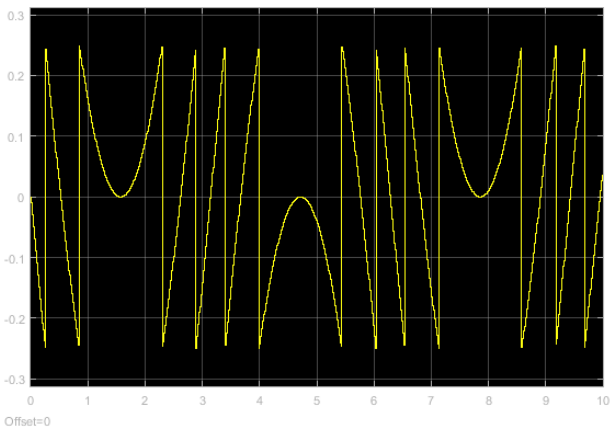
2. Representación digital de señales

Cuantización uniforme

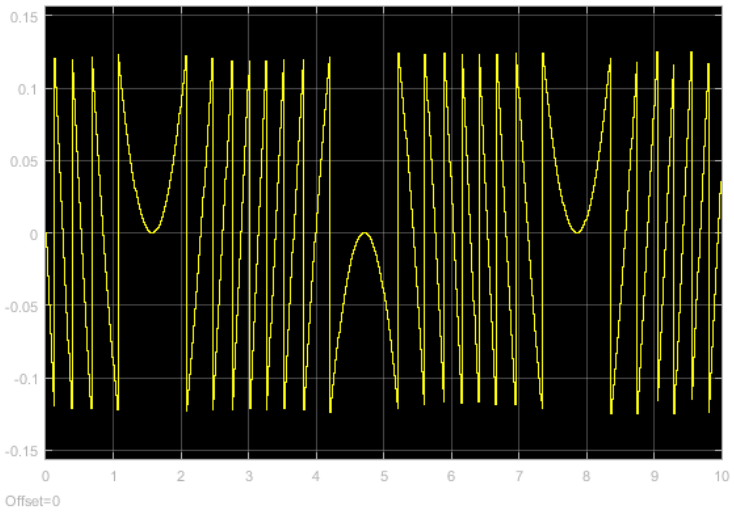
Ahora montamos un sistema con una señal sinusoidal y después la cuantizamos para obtener una función de tiempo discreto y valores discretos, es decir, una señal digital. Después calculamos el error y mediante Matlab calculamos el SNR(Signal – Noise Ratio) para una cuantización de 4 bits.



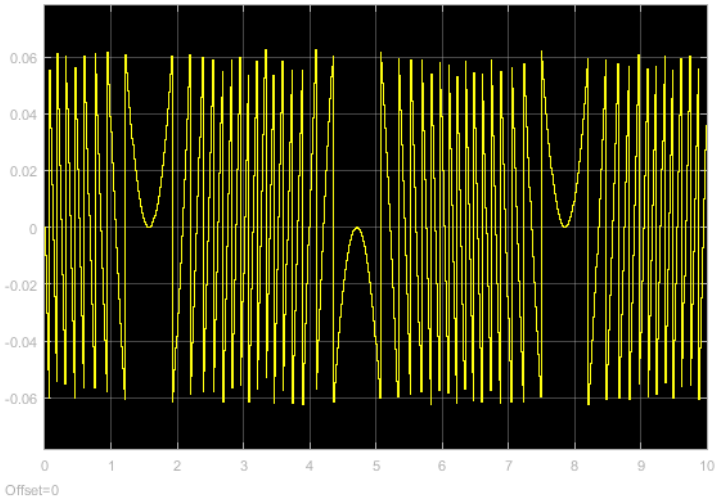
La forma de la onda error usando 2 bits para definir los posibles estados del cuanto



3 bits para nivel de cuanto



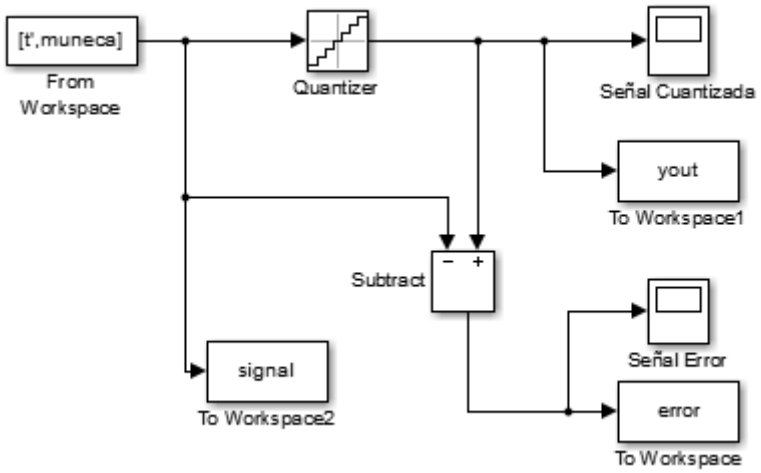
4 bits para nivel de cuanto

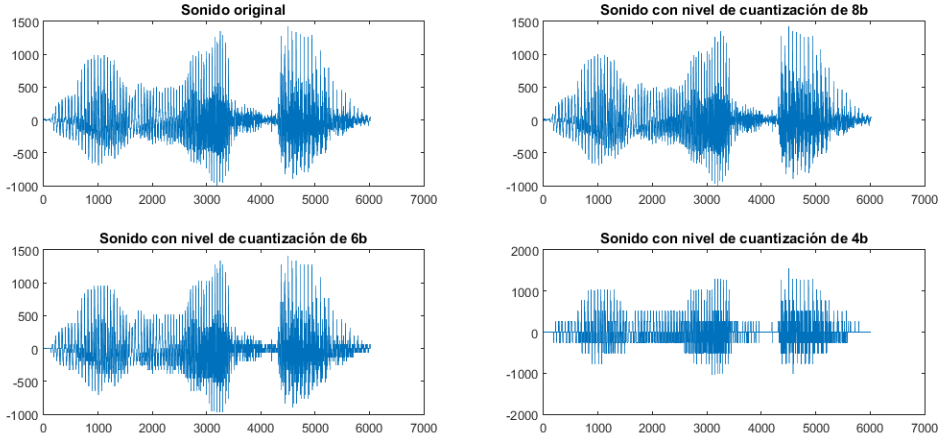


Como se puede observar hay menos error cuando usamos más bits para el cuantizador.

```
>> SNR2bits= 10*log10(sum(signal.*signal)/sum(error.*error))

SNR2bits =

   14.3674

>> SNR3bits= 10*log10(sum(signal.*signal)/sum(error.*error))

SNR3bits =

   20.1547

>> SNR4bits= 10*log10(sum(signal.*signal)/sum(error.*error))

SNR4bits =

   26.0077
```

Ahora digitalizamos la señal del sonido con cuantos de 4,6 y 8 bits. El sistema es

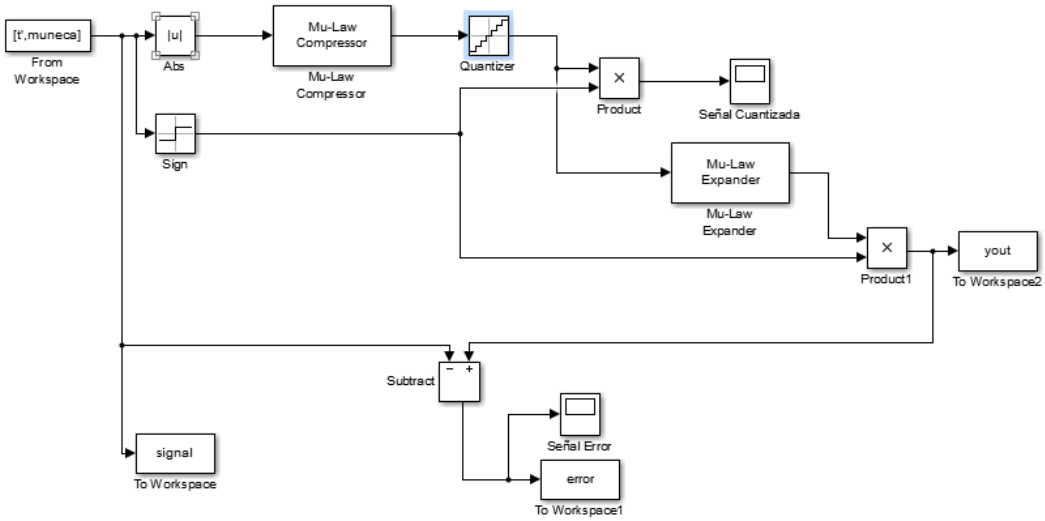




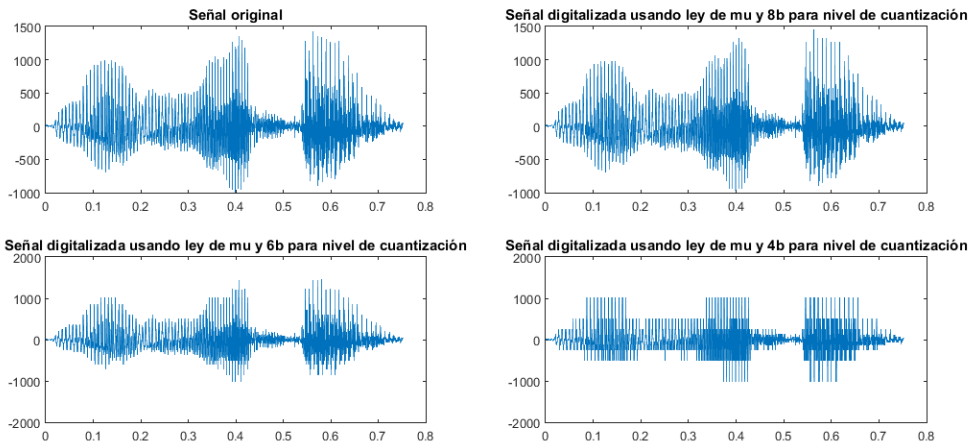

```
>> snrSonido=[snr8b snr6b snr4b]

snrSonido =

   36.0635   23.7077   12.1993
```

Cuantificación usando la ley de Mu

En este caso usamos la ley de mu para transformar la señal de tiempo discreto antes de discretizarla en los valores que puede tomar la señal, es decir, antes de usar la función cuantizadora. El sistema usado es el siguiente



Para este caso hemos seguido el procedimiento anterior y hemos usado distintos números de bits para definir los niveles de cuantización que podemos usar. Estos valores siguen siendo 8b, 6b y 4b.



Los resultados del ratio Señal-Ruído son para 8,6 y 4 bits (en dB):

```
>> snrMu

snrMu =

   37.8656   25.8379   13.8530
```

Modulación delta