Digitalización de señales y aliasing

Basto Sara, Caceres Sebastian, Parada David,
{*1803338, 1803245, 1803333*}*@unimilitar.edu.co*
Profesor: Diego Bravo

***Resumen*—En este documento se realiza la digitalización de una señal sinusoidal que varía de frecuencia en un rango especificado, implementando un módulo dac y un módulo adc por medio de dos microcontroladores. Posteriormente, estos datos se envían a una aplicación realizada en python donde se visualizan, grafican y los almacena en el disco druro del computador. Adicionalmente, se simula el proceso de muestreo y cuantificación, se estima la fidelidad del sistema de adquisición y el error de cuantificación por medio de Matlab, tomando en cuenta conceptos como frecuencia de muestreo, ancho de banda, numero de bits, entre otros. Finalmente se realiza un análisis a partir de los resultados obtenidos y se compara las señales en simulación y las señales reales.**

***Abstract*—In this document the digitization of a sinusoidal signal that varies in frequency in a specified range is performed, implementing a dac module and an adc module by means of two microcontrollers. Subsequently, these data are sent to an application made in python where they are visualized, plotted and stored on the hard disk of the computer. Additionally, the sampling and quantization process is simulated, the fidelity of the acquisition system and the quantization error are estimated using Matlab, taking into account concepts such as sampling frequency, bandwidth, number of bits, among others. Finally, an analysis is carried out from the results obtained and the signals in simulation and the real signals are compared.**

***Palabras clave*—Señal, Frecuencia, Periodo, Muestra, Ancho de banda, Antialiasing.**

I. Competencias a desarrollar

- Identificar y medir el error producido por el muestreo y la cuantificación de señales.
- Reconocer la importancia del teorema de muestreo.
- Realizar la digitalización y posterior reconstrucción de señales análogas.
- Identificar el fenómeno de aliasing.

II. Intruducción

Una señal en el dominio del tiempo es una señal análoga, la cual puede tener valores infinitos para cada valor de tiempo, por lo cual es importante si se va a trabajar con señales analógicas, entender que para poder registrar esta señal dentro de un sistema embebido es necesario realizar su digitalización, y esto conlleva tomar muestras de la señal cada determinado tiempo, por lo cual ahora se estaría trabajando con muestras y valores discretos almacenados dentro de un microcontrolador.

El presente documento corresponde a un informe de laboratorio de "Procesamiento Digital de Señales" presentado en la Universidad Militar Nueva Granada durante el periodo 2021-1.

Por otro lado se debe entender el concepto de periodo de muestreo, simplemente es el tiempo que hay entre muestra y muestra, y debe ser siempre el mismo para que toda la teoría de señales funcione adecuadamente, la frecuencia de muestreo de igual forma que en señales análogas representa el inverso del periodo, es decir cuantas muestras se estarían tomando por unidad de tiempo, sin embargo se debe tener en cuenta que si una señal tiene determinada frecuencia, para muestrearla se debe usar por lógica una frecuencia mayor a la de la misma, debido a que no tendría lógica intentar muestrear una señal cada cierto tiempo si está ya cumplió su ciclo, al hacer esto se obtiene un fenómeno que se conoce como aliasing, simplemente la señal muestreada seria indistinguible de la original.

Una vez se haya leído la señal con alguna estrategia ya sea representándola con un voltaje, campo magnético etc... se debe realizar un proceso para convertir esta variable utilizada en el valor que tenia la señal en la muestra tomada, esto se puede realizar con una ecuación dentro del microcontrolador o con alguna estrategia equivalente, una vez se tienen las muestras, es importante tener definido el periodo de muestreo usado, ya que sin el seria imposible conocer como era la señal original a partir de las muestras.

Comprendidos estos conceptos se puede explicar de manera amena lo que se planteo realizar en este desarrollo de laboratorio, como primera instancia se generó una señal con periodo variable usando un Arduino, y variando su frecuencia con un potenciómetro por medio del un ADC, está señal se emitió por su puesto con un modulo DAC, y posteriormente leída por un microcontrolador STM32F7, que es donde se realizó la digitalización de esta señal, de allí se guardó en un archivo csv para poder ser comparada con una señal simulada con Matlab y muestreada virtualmente con Matlab, por medio de funciones, para determinar el grado de error entre la señal original y la discretizada por medio de los microcontroladores, también se variaron parámetros de frecuencias de la señal original, así como de frecuencia de muestreo, para poder observar que comportamientos tenia la señal muestreada respecto a la original y como se veía afectada la frecuencia que se obtenía de las muestras obtenidas.

Por un lado se hace evidente que si se trabaja con este tipo de señales hay ventajas como el poder manejar a gusto los datos de la señal muestreada, guardarlos y estudiarlos, no obstante es claro que el método puede ser mas complejo

que simplemente procesar una señal con métodos analógicos como filtros, amplificadores y se tiene una menor velocidad de respuesta, por ello se debe tener bien entendido para que se realiza la digitalización, en este caso, para comprender como funciona este proceso y sus correspondientes fenómenos.

III. Metodología

En esta práctica se va a generar una señal analoga por medio de un conversor análogo dígital por el cual se va a generar una onda senosoidal con frecuencia la cual se puede variar desde 0.01s hasta 2s, esta señal se va a leer por medio de un ADC de otro microcontrolador el cual se le va a colocar un muestreo de 10 muestras por segundo enviando estos datos obtenidos por medio de comunicación serial a un programa en python. Este exporta los datos a matlab donde se genera una señal por medio de la función matemática y se leen los datos obtenidos, estas señales se les quita el desfase, por último hacer una correcta comparación entre estas generando calculos estadisticos como lo es el PSR MSE varianza de error y media de error.

Para está práctica se van a utilizar

- 2 STM32F746ZGYx
- Protoboard
- Computador
- 1 Potenciometro
- Software Matlab
- Software Python
- Software uKeilVision

IV. Desarrollo de los ejercicios prácticos

En la práctica se ejecutaron diversos procesos donde se realizó el respectivo análisis de los datos obtenidos.Estos procesos se representan por medio del siguiente diagrama

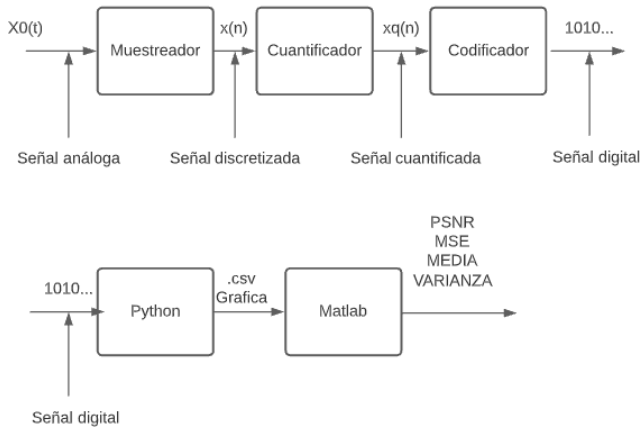

Figura 1. Diagrama de procesos

IV-A. Generacion de la señal análoga

Inicialmente, se lee la señal análoga de un potenciometro por medio del ADC un microcontrolador STM y se usa de argumento en la frecuencia de una onda seno que varía en rangos de 0.5Hz a 2 Hz generada por medio del modulo DAC de dicho microcontrolador, este proceso se realiza a 12 bits.La función que genera la señal se puede observar en la figura 2.

Adicionalmente se modifico también el código para que este también produciera un DAC desde 0.01s a 2 s por medio de un potenciometro.

```
prueba=sin(2*pi*f*t);
fun=2047*prueba+2047;
DAC->DHR12R1 = fun ;
```

Figura 2. Función del DAC

IV-B. Discretización y cuantificación de la señal

Los datos del DAC son leídos por medio del ADC de un segundo microcontrolador a una frecuencia de muestreo de 10 muestras/segundo esto se logró por medio del timer del microcontrolador, el cual posee un periodo de 1ms , a cada valor de la señal se le asignó su respectiva resolución, es decir, de 0 a 4095. Además el adc anterior también fue modificado para que su resolución fuera de 10 bits para hacer los diferentes experimentos mostrados más adelante.

```
h=TIM2->CNT;
if (TIM2->CNT >=100)


dato_adc2= ADC2->DR;
r4=dato_adc2 /1000;
r3=dato_adc2 -r4*1000;
r3=r3/100;
r2=dato_adc2 -r4*1000-r3*100;
r2=r2/10;
r1=dato_adc2 -r4*1000-r3*100-r2*10;
en[1]=numeros[r4];
en[2]=numeros[r3];
en[3]=numeros[r2];
en[4]=numeros[r1];
en[0]='\n';

TIM2->EGR=1;
}
```

Figura 3. Función del ADC y timer

IV-C. Señal digital y pyhton

Seguido a esto, las muestras se envian por medio de comunicación serial a una aplicación desarrollada en python que grafica y guarda los datos en un archivo csv, como se observa en la figura 1.

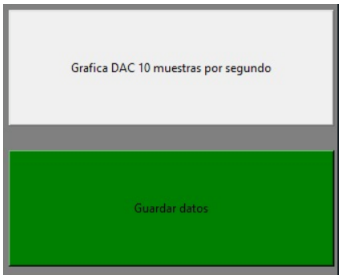

Figura 4. Interfaz gráfica

La gráfica de las señal digitalizada se realiza en tiempo real, tal y como se observa en la figura 5.Donde los límites en el eje x de la gráfica se encuentra en 20 muestras para lograr visualizar correctamente las ondas con las diversas frecuencias especificadas con anteioridad.

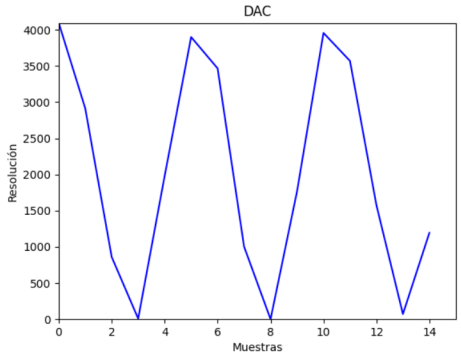

Figura 5. Grafica señal DAC

En la tabla I se logra visualizar la relación de la frecuencia de la señal análoga con la frecuencia normalizada de la señal digitalizada.

Tabla I
RELACIÓN DE FRECUENCIAS

F. SEÑAL ANALOGA(Hz)	F. SEÑAL DIGITALIZADA (Ciclos/ muestra)
0.5	1
0.5714	1.14
0.666	1.33
0.8	1.6
1	2
1.33	2.66
2	4

De esta manera, los ciclos por muestra de la onda aumentan en función del aumento de la frecuencia de la señal análoga.

Posteriormente, se realizó el decremento del periodo de la señal senoidal hasta alcanzar 0.2 segundos, es decir a una frecuencia de 5 Hz y 10 ciclos/muestra en frecuencia normalizada, esto se realizó con 1o bits.El resultado se puede observar en la siguiente imagen.

Figura 6. Grafica periodo 0.2s prueba 1

Se realizaron varias mediciones, reiniciando el microcontrolador.Obteniendo señales como las de la figura 7 y 8.

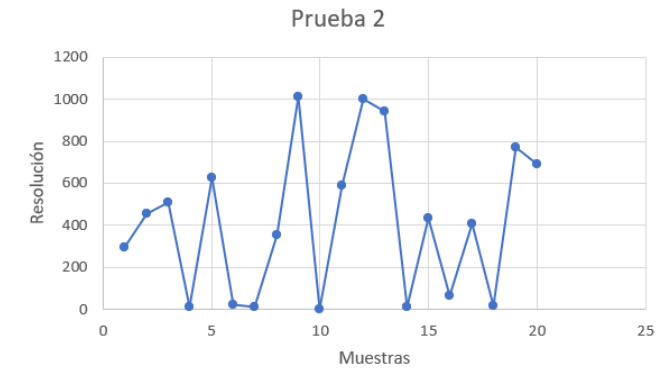

Figura 7. Grafica periodo 0.2s prueba 2

Figura 8. Grafica periodo 0.2s prueba 3

Aunque se realizaron dos pruebas mas a parte de las anteriores, no se obtuvieron señales periódiass debido a que se requieren 10 ciclos de la señal por cada muestra, y a esta frecuecia no se logra realizar el correcto muestreo de la señal análoga.

Seguido a esto, se disminuyó la frecuencia a 100 milisegundos, por lo que se obtuvo el siguiente resultado.

Figura 9. Grafica periodo 100ms

Aunque la señal tampoco es periodica, se observa mas definida la señal con periodo de 0.2 segundos que la señal de 100ms puesto que en esta última se deberían tomar 20 ciclos/muestra.En estas dos ultimas señales que no se observan periódicas, se puede ver el efecto aliasing, debido a que las señales continuas se tornan indistinguibles al realizar el muestreo digital.Para corregir esto, se debería cambiar la frecuencia de muestreo segun nyquist[2], donde;

$$FN > 2xFmax \tag{1}$$

Por lo que la frecuencia de muestreo debería ser mas de 20 muestras por segundo.

IV-D. MATLAB

En Matlab se realizó un código el cual simula las señales análogas periódicas obtenidas desde el generador de señales. Para ello:

```
%----------------------------------------------------------------------------
%1) Realice la simulación del proceso de muestreo de la señal análoga
%----------------------------------------------------------------------------
% Señal de tiempo continuo Nbits/Tiempito/CSV/escribir consola/Graficas
nbits=10;
tiempito=-0.1432;
t=0:0.001:2;%Dos segundos de tiempo señal analoga
f=1/2;   %Periodo Varible de 0.01segundos(100Hz) a 2 segundos(0.5Hz)
xt=((2^nbits-1)/2)+((2^nbits-1)/2)*sin(2*pi*f*t); %Señal analoga
% Discretización de la señal análoga
Ts=1/10; % Periodo de muestreo 10 muestras/s 0.1segundos
Fs=1/Ts;% Frecuencia de muestreo 10 muestras/s
nTs=tiempito:Ts:1.9+tiempito;%Si es 2 toma 21 datos, si es 1.9 toma 20
xnTs=((2^nbits-1)/2)+((2^nbits-1)/2)*sin(2*pi*f*nTs);
xnTs=round(xnTs);
vector=1:1:20;
%Mostrando en pantalla
figure('Name','Señal análoga y señal discretizada')
% plot(nTs,xnTs);
hold on;grid on
stem(vector,xnTs);
% hold off
```

Figura 10. Simulación señal teórica.

El anterior segmento de código realiza la simulación del muestreo de una señal análoga con un periodo de muestreo definido, esta función depende de el numero de bits y de la frecuencia, se usa en comando round para redondear al valor decimal más cercano, como ocurre en el proceso de cuantificación en el microcontrolador, es importante destacar que se realiza está señal simulada para comprobar con la señal muestreada experimentalmente.

Posteriormente se realiza la lectura del archivo csv por medio de la función readtable, la cual filtra únicamente los valores numéricos separados por comas.

```
%----------------------------------------------------------------------------
%%%%%%%%%%%%%%%%Lectura archivo csv%%%%%%%%%%%%%%%%
%----------------------------------------------------------------------------
M=table2array(readtable('10bits periodo 2.csv'));
eje_x = M(:,1);%Guardamos numero de muestra en un vector
eje_y = M(:,2);%Guardamos las muestras voltaje digitalizadas en un vector
eje_y1=transpose(eje_y);%Hacemos la transpuesta filas a col
figure('Name','Señal STM')
stem(eje_x,eje_y1);%Graficamos Señal Digitalizada y Cuantificada
```

Figura 11. Lectura del archivo csv.

Se implementan dos vectores para guardar los datos, el vector eje_x almacena el número de muestra, y en el vector eje_y almacena el valor de la muestra, se le realiza la transpuesta para que todos los datos queden sobre la misma fila y pueda ser comparado con el vector xnTs, cuyo contenido es la muestra teórica, adicionalmente se gráfica, para poder observar la señal muestreada.

Es importante resaltar que para realizar las operación PSNR y MSE los vectores deben cumplir dos condiciones, la primera es que ambos tengan las mismas dimensiones, y la segunda es que estos vectores deben contener datos sobre una misma muestra y en fase, es decir ambos comienzan el muestreo de las señales en un mismo instante de tiempo (t para la señal), entre mayor sea la diferencia de tiempo entre los tiempos de inicio de muestreo, mayor será el error para el cálculo del PSNR, MSE y demás errores que se calculen por medio de estos dos muestreos.

Por ello se debe implementar un algoritmo o método que corrija las fases entre las señales para que estén en fase, se determinó que era más sencillo modificar la fase de la señal simulada, de manera que se tomaba el primer dato obtenido en el archivo csv y se calcula el tiempo en el cual la función seno debe tener para tomar este valor de manera gráfica, esto se debe a que se obtienen dos valores cuando la señal se encuentra decreciente y cuando esta creciente. De manera que esto se determino con los datos obtenidos el archvio csv y se determino el tiempo el cual iniciar a muestrear los datos.

Posteriormente por medio de la función en Matlab “psnr()”, se calcula el psnr el cual recibe como parámetros un vector a evaluar es decir el vector de datos experimentales y una referencia, es decir el vector de datos teóricos. Para el calculo del MSE se implementó la función “immse()”, esta recibe los mismos parámetros y calcula el error cuadrático medio entre dos matrices, una matriz a evaluar y una referencia respectivamente.

```
%----------------------------------------------------------------------------
%%%%%%%%%%%%%%%%Hallar PSNR entre SIMULADA Y DIGITALIZADA ya en fase%
%----------------------------------------------------------------------------
%Hallar el PSNR entre muestra y simulacion de la muestra
%fuente: https://la.mathworks.com/help/images/ref/psnr.html
[peaksnr, snr] = psnr(eje_y1, xnTs)
fprintf('\n The Peak-SNR value is %0.4f', peaksnr);

%----------------------------------------------------------------------------
%%%%%%%%%%%%%%%%Hallar Error Cuadratico Medio MSE%%%%%%%%%%%%%%%%
%----------------------------------------------------------------------------
%fuente: https://la.mathworks.com/help/images/ref/immse.html
MSE = immse(eje_y1, xnTs);
fprintf('\n The mean-squared error is %0.4f\n', min(MSE));
```

Figura 12. PSNR MSE

El error de cuantificación se determinó por medio de un algoritmo que se encarga de crear un vector con los niveles de cuantificación y le adiciona el valor de medio delta entre niveles de cuantificación, representando la incertidumbre que hay al cuantificar señales, por medio de un for se leen los valores de la señal muestreada y se comparan con los valores de la señal teórica simulada, la diferencia entre ellos es este delta que representa el error de cuantificación.

```
%-------------------------------------------------------------------------
%%%%%%%%%%%%%%%%%%Hallar Error De Cuantificacion %%
%-------------------------------------------------------------------------
%cuantificando la señal de tiempo discreto
ran=max(eje_y1)-min(eje_y1);%rango de la señal
nnivs=(2^nbits)-1;%numero de niveles de cuantificacion
delta=ran/nnivs;%altura de nivel
nive=min(eje_y1):delta:max(eje_y1);
nivem=nive+delta/2;
[~,N]=size(nTs);
xnq=zeros(1,N);
for i=1:N
j=1;
while eje_y1(i)>nivem(j)
j=j+1;
end
xnq(i)=nive(j);
end
err=eje_y1-xnq;
figure('Name','Error de cuantificacion')
stem(vector,err);
```

Figura 13. PSNR MSE

Finalmente por medio de las funciones mean() y var(), se determinan los valores de la media y de la varianza en el error de cuantificación.

```
%-------------------------------------------------------------------------
%%%%%%%%%%%%%%%%%%Media y Varianza %%%%%%%%%%%%%%%%%%
%-------------------------------------------------------------------------
%Media fuente:https://la.mathworks.com/help/matlab/ref/mean.html?searchHighlight=mean&s_tid=srchtitle
%Varianza fuente:https://la.mathworks.com/help/matlab/ref/var.html
Media = mean(err);
fprintf('La media del error es: %0.4f\n', Media);
Varianza=var(err);
fprintf('La varianza del error es: %0.4f\n', Varianza);
```

Figura 14. Calculo varianza y media

Se realizaron las gráficas de la señal digitalizada por la stm, la señal simulada por matlab y el error de cuantificación para un periodo de un segundo a 10 bits.

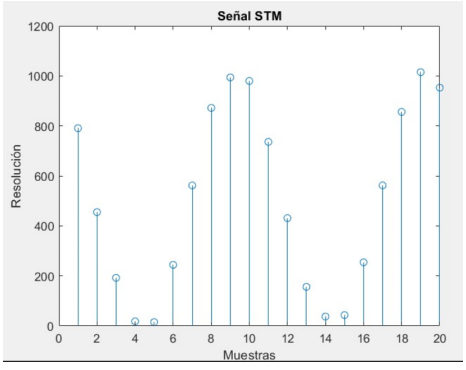

Figura 15. Gráfica de 10 bits stm

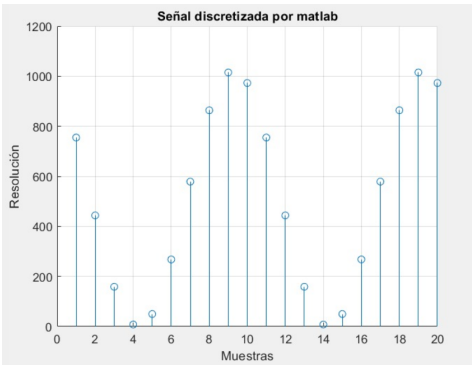

Figura 16. Gráfica de 10 bits matlab

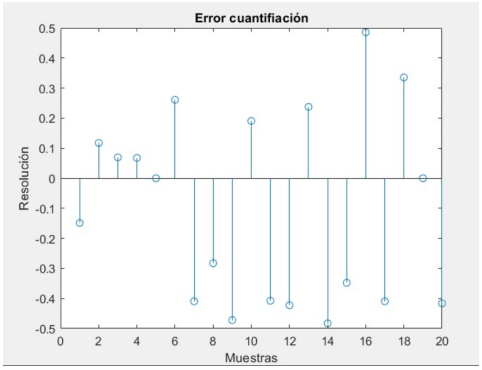

Figura 17. Error de cuantificación 10 bits

Se realizaron las gráficas de la señal digitalizada por la stm, la señal simulada por matlab y el error de cuantificación para un periodo de un segundo a 12 bits.

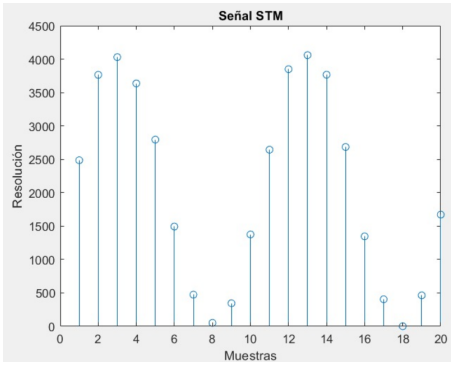

Figura 18. Error de cuantificación 10 bits

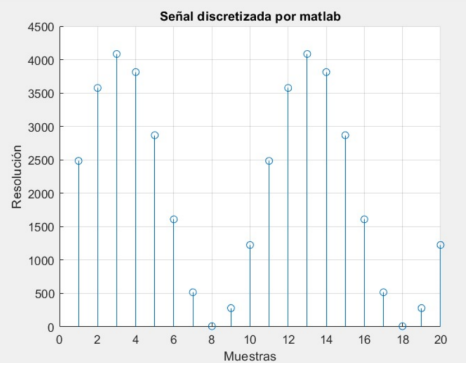

Figura 19. Error de cuantificación 10 bits

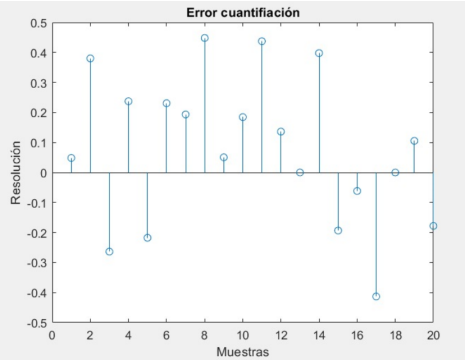

Figura 20. Error de cuantificación 10 bits

Tabla II
ESTIMACIÓN FIDELIDAD Y ERROR 12 BITS

12 BITS				
PERIODO (S)	PSNR(db)	MSE	MEDIA	VARIANZA
0.5	29.23	20018.25	−0,0137	0.0594
0.75	16.31	392104.9	0.0363	0.0615
1	27.69	28507.15	0.0762	0.0608
1.25	25.37	48679.05	-0.0274	0.0813
1.5	33.09	8221.4	0.0691	0.0636
1.75	24.03	66188.4	0.0016	0.0644
2	34.1347	6472.05	0.0511	0.0441

Tabla III
ESTIMACIÓN FIDELIDAD Y ERROR 10 BITS

10 BITS				
PERIODO (S)	PSNR(db)	MSE	MEDIA	VARIANZA
0.5	38.33	2457.7	0.0229	0.0717
0.75	27.02	33285.35	0.0238	0.0685
1	46.43	381.400	-0.1018	0.0978
1.25	25.20	50627.75	0.0832	0.0494
1.5	35.66	4549.4	-0.1051	0.0541
1.75	24.977	53305.8	-0.0245	0.0914
2	25.5467	46756.5	0.0179	0.0833

Al observar la tabla II Y III se puede evidenciar que la fidelidad de estas señales es alta debido a que el PSR es mayor a 20 a excepción de la señal de 12 bits periodo de 0.75s esto puede deberse a un error a la hora de la trasmición de datos en el microcontrolador, la media del error establece que el error entre la señal es bajo de igual forma la varianza entre los error es bajo.

Al comparar el error de cuantización de 10 bits contra los obtenidos en los 12 bits se puede observar que el error en 10 bits es mayor que en el de 12 bits.esto se debe al rango de valores que puede tomar el voltaje en estos dos casos en 12 bits al poseer 4095 valores puede tener un rango mayor por lo tanto se acercará más a la señal real.

En la tabla II y III se puede evidenciar que la variación de los errores entre las muestras obtenidas es pequeña por lo tanto se puede decir que el error es casi uniforme para cada muestra.

V. CONCLUSIONES

Al comparar las gráficas de señal simulada vs señal experimental, se observó que entre mayor es la frecuencia de la señal, mayor es el error, y esto se debe a que como habrá mayor numero de datos por unidad de tiempo, será más difícil el muestreo correcto de estos cuando se digitalizan.

Se observó que al realizar el muestreo con un numero mayor de bits el error de cuantificación disminuía debido que se tiene una mejor resolución, disminuyendo la banda de incertidumbre entre cada nivel de cuantificación.

Para hacer una correcta comparación de 2 señales estas deben estar en el mismo rango de tiempo sino el dato calculado será incorrecto y no dará nada concluyente.

Los datos obtenidos PSNR como se observan en las tablas son superiores de 24Db de manera que se puede decir que la señal generada es de alta fidelidad según la norma ISO 12640-1:1997[4].

REFERENCIAS

[1] MAYNAR, Francisco Melis. Agregación temporal y solapamiento o aliasing. Estadística española, 1992, no 130, p. 309-346.

[2]LANDAU, H. J. Sampling, data transmission, and the Nyquist rate. Proceedings of the IEEE, 1967, vol. 55, no 10, p. 1701-1706.

[3]s.Luis Pastor, et al. Cuantificación del Error en las Mediciones Debido a la Frecuencia de Muestreo. Computación y Sistemas, 2004, vol. 8, no 2, p. 86-105.

[4] KUMAR, Pankaj, et al. Study of Various ISO Standards Related to Gravure Printing Presses. International Journal of Science, Engineering and Computer Technology, 2016, vol. 6, no 2, p. 141-148.