39

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE ESTUDIOS SUPERIORES

CUAUTITLAN

ESTUDIO SOBRE EL CONVERTIDOR ANALOGICO DIGITAL

T E S I S

QUE PARA OBTENER EL TITULO DE

INGENIERO MECANICO ELECTRICISTA

P R E S E N T A :

ROBERTO GONZALEZ HUERTA

ASESOR: ING. JOSE JUAN CONTRERAS ESPINOSA

CUAUTITLAN IZCALLI, EDO. DE MEX. AGOSTO, 2000

UNIVERSIDAD NACIONAL AVENOMA DE MEXICO

FACULTAD DE ESTUDIOS SUPERIORES CUAUTITLAN
UNIDAD DE LA ADMINISTRACION ESCOLAR
DEPARTAMENTO DE EXAMENES PROFESIONALES

ASUNTO: VOTOS APROBATORIOS

DR. JUAN ANTONIO MONTARAZ CRESPO
DIRECTOR DE LA FES CUAUTITLAN
P R E S E N T E

ATN: Q. Ma. del Carmen García Mijares
Jefe del Departamento de Exámenes
Profesionales de la FES Cuautitlán

Con base en el art. 28 del Reglamento General de Exámenes, nos permitimos comunicar a usted que revisamos la TESIS:
" Estudio Sobre el Convertidor Analógico Digital "

que presenta el pasante: Roberto González Huerta
con número de cuenta: 8841476 - 2 para obtener el título de :
Ingeniero Mecánico Electricista

Considerando que dicho trabajo reúne los requisitos necesarios para ser discutido en el EXAMEN PROFESIONAL correspondiente, otorgamos nuestro VOTO APROBATORIO

A T E N T A M E N T E
"POR MI RAZA HABLARA EL ESPIRITU"
Cuautitlán Izcalli, Méx. a 23 de octubre de 2006

PRESIDENTE	Ing. José Juan Contreras Espinosa	
VOCAL	Ing. Gloria Villanueva Aguilar	
SECRETARIO	Ing. Juan González Vega	
PRIMER SUPLENTE	Ing. José Luz Hernández Castillo	
SEGUNDO SUPLENTE	Ing. Pedro Guzmán Tinajero	

Dedico este trabajo a Manuel y Esperanza por darme la oportunidad de tener un futuro promisorio y ser un hombre de éxito

INDICE

INTRODUCCION

INTRODUCCIÓN

En está tesis presentamos las variables existentes en el diseño al utilizar el Circuitos Integradc Convertidor Analógico Digital CAD.

La mayoría de los libros que actualmente existen en español y su inmensa mayoría en inglés estudian los diferentes tipos de CAD, destacando solamente los principios básicos de funcionamiento, estructura y aplicación, pero solo en forma ideal. Esto no es suficiente ya que al utilizar un CAD producen comportamientos inherentes que se deben de tomar en cuenta, cuando es necesario se debe implemetar en función de un análisis matemático para poder incrementar la exactitud y precisión, de otra forma esta no se podrá mejorar. Al final de cuentas lo que busca un CAD es el leer la lectura analógica y producir un código binario que represente al valor analógico de entrada (ver en el Capitulo 1 Fig.1.1); pero, que tan exacta es la representación digital de la lectura analógica y como puede hacer para incrementar esta exactitud, el pretender contestar lo anterior es el fin de esta tesis.

Partiendo del hecho, que un CAD es un dispositivo no ideal, para realizar la implementación con un CAD, trae como consecuencia el estudio de variables que deberán ser analizadas antes y durante el diseño para determinar como el

diseñador podrá controlarlas. Por esto, expondremos errores, criterios de diseño, propiedades del semiconductor, etc.

Al mismo tiempo, se hace un estudio de las gráficas que caracterizan el comportamiento de este tipo de Circuitos Integrados. Estas gráficas serán splots reales de resultados obtenidos al ser analizado el uso del Convertidor Analógico Digital en el laboratorio.

Hablaremos también sobre Medición, Prueba y Problemas que se presentan en el diseño al utilizar el CAD. Se mencionarán algunas recomendaciones para realizar un diseño óptimo. Realizaremos la definición de los diferentes conceptos que envuelven el estudio del CAD.

Cabe hacer notar que el fabricante del CAD describirá propiedades, valores y nomenclaturas. En caso de asignar una nueva característica en su silicón diferente a las existentes en el mercado, la describe y especifica en *la hoja del componente - Data Sheet.*

El CAD por ser un dispositivo periférico, comúnmente hace interfase con una unidad de procesamiento, al ser conjugado con esta unidad, dependerá del programa que el diseñador desarrolle para la unidad de procesamiento digital. Presentaré el por que', la necesidad de un algoritmo matemático como una herramienta para tender mas al valor real analógico representado por el valor

digital de esta lectura analógica, cuando existe la interacción con una unidad de procesamiento.

Estudiaremos la transformada rápida de Fourier como herramienta necesaria para obtener conversiones más cercanas al valor real. Propondremos un algoritmo utilizando las FFT´s. Igualmente utilizaremos la probabilidad para saber como podemos eliminar eficazmente el error producido en el "*quantizer - cuantización*" buscando acercarnos más a la verdad de la lectura analógica real representada por un formato digital al ser convertido.

Haremos énfasis en la FFT como una herramienta fundamental la utilizar una unidad procesamiento digital. De igual manera dotaremos de un lenguaje característico utilizado en el mundo del CAD.

En el Capitulo 1 estudiamos los diferentes tipos de errores, como pueden afectar el funcionamiento estático del Convertidor Analógico Digital – CAD, las consideraciones que deben ser tomadas en cuenta, y gráficas de estos, además de analizar los errores más comunes.

En el Capitulo 2 estudiamos las diferentes arquitecturas así como el funcionamiento de los tres diferentes tipos de Convertidor Analógico Digital, el Convertidor Flash, Convertidor de Aproximaciones Sucesivas, así como,

Convertidor Delta Sigma $\Delta\Sigma$. Cabe señalar que algunos estudios pueden hablar de mas tipos pero estos son ramas de estos tres.

En el Capitulo 3 nos metemos de lleno en el comportamiento dinámico del convertidor analógico digital, presentamos el criterio para eliminar ventanas, y mencionamos parámetros a medir durante este análisis.

Se estudiará la importancia en determinar la frecuencia de muestreo ideal en la cual se tenga que hacer trabajar el CAD y él por qué de esta frecuencia. Se mencionará la importancia de los filtros digitales y por qué se utilizan.

En el Capitulo 4 describimos la numerología que las compañías de semiconductores utilizan para traducir las lecturas en cantidades discretas digitales, mencionamos los distintos formatos que los bits tienen y que se dan en la salida de un CAD después de la conversión analógica a digital, además presentamos dos criterios que proponemos que el ingeniero debe de considerar para escoger un CAD.

En el capitulo 5 hacemos el análisis de algoritmo de la Transformada Rápida de Fourier presentada en el Capitulo 3, lo detallamos, y presentamos de forma real esta programa.

En síntesis, esta tesis tiene como finalidad principal responder cómo poder encontrar el CAD exacto para una aplicación específica, cómo realizar el algoritmo de optimización, cómo probar, medir y buscar una solución en problemas de diseño al utilizar los CAD´s y el por qué de la necesidad de un algoritmo donde se implemente *La Transformada Rápida de Fourier TRF- Fourier Fast Transform FFT* como herramienta matemática.

Un tutorial sobre la Transformada Rápida de Fourier TRF - Fast Fourier Transform FFT

Mucha de la gente que esta empezando con el CAD no se encuentra muy familiarizado con la Transformada de Fourier, trataré de explicar de forma muy sencilla él por qué de su importancia y él por qué utilizar la Transformada Rápida de Fourier TRF – Fast Fourier Transform FFT.

La grabación digital más común de audio es llamada *Pulse Code Modulation PCM - Modulación por ancho de pulso MAP.* El PCM es el tipo de código de

modulación que los CD's y la mayoría de los archivos WAV utilizan. En un hardware PCM para grabar, un micrófono convierte una variación de presión de aire (ondas de sonido) en una variación de voltaje. Después, un CAD convierte la medida de voltaje a intervalos regulares de tiempo. Por ejemplo, en un disco compacto la grabación de audio, tiene exactamente 44,100 muestras tomadas cada segundo. Cada muestra es un valor de voltaje convertido de un entero de 16 bits. Un CD contiene dos canales de datos: uno para el oído izquierdo y otro para el oído derecho para producir la sensación estereo. En un disco compacto los dos canales son grabaciones una independiente de la otra, colocada una al lado de la otra. De hecho los datos de canal alternan de izquierda a derecha, de derecha a izquierda y así sucesivamente.

Los datos que resultan de una grabación de PCM son función del Tiempo. Regularmente sorprende mucho a la gente que millones de secuencias de enteros en una grabación de CD puedan representar con tal nitidez la música y la voz humana. La gente trata de entender cómo una cadena de números puede sonar como si fuese una orquesta completa. Cuando el oído oye más de un sonido a la vez, los diferentes sonidos son físicamente mezclados uno con otro en los oídos, como un solo patrón de presión de aire. Los oídos y el cerebro trabajan juntos para analizar la señal dentro las diferentes sensaciones auditivas. Para entender por que esto es verdad, imagina que tu puedes poner una cámara microscópica en el oído para oír la vibración del oído en cámara lenta. Supón que la cámara es muy

tan rápida que puede tomar imágenes cada 1/44,100 de segundo. También supón que las imágenes capturadas en esta cámara son tan nuevas y brillantes que tu no puedes discernir las 65,536 (64k) de distintas posiciones de la superficie del tímpano de tu oído, como si esta se moviera de atrás para adelante en respuesta a la entrada de ondas de sonido. Si tu utilizas el tímpano de tu oído mientras estas escuchando a tu mejor amigo decir tu nombre, y después tomas los resultados de la película y escribes los resultados numéricos de la posición de tu tímpano cada cuadro de la película, tu tendrás una grabación PCM. Si mas tarde puedes hacer tu oído vibrar de atrás hacia delante de acuerdo con los miles de números y los escribes, tu oirás la voz de tu amigo diciendo tu nombre exactamente como si esta fuera la primera vez, realmente no importa que sonido es, si es tu amigo, una orquesta completa, o una sinfónica, el concepto es el mismo. Cuando tu escuchas mas de una cosa a la vez, todos los distintos sonidos son mezclados físicamente juntos en tus oídos como un solo patrón de la variación de la presión. Tu cerebro y tus oídos trabajan juntos para analizar la señal dentro de un auditorio separado de sensaciones. . Literariamente esta todo en la cabeza.

Información de la Frecuencia en Función del Tiempo

El órgano que se encuentra en nuestro oído llamado cóclea nos permite detectar la tonalidad de los sonidos que nosotros oímos. La cóclea esta acústicamente acoplada al yunque por una serie de conductores llamados conductos semicirculares. Estos conductores en espiral cuentan con una superficie arrugada llena de líquido y bello pequeño. El bello por el lado del oído externo es más grande y va descendiendo conforme el espiral va hacia el extremo opuesto. Es decir conforme se va acercando el bello al oído interno, el bello va bajando de longitud gradualmente, haciéndose más pequeño como se va acercándose al oído interno. Cada bello esta conectado al nervio auditivo, entre más grande sea el bello resonara a menor frecuencia y los más pequeños resonarán a mayores frecuencias.

Como puedes observa la cóclea sirve para convertir una señal de presión de aire en información de frecuencia la cual puede ser interpretada con tonalidad y textura. De esta forma nosotros podemos diferenciar entre diferentes notas de piano, aun si estas notas son tocadas con el mismo volumen. La Transformada de Fourier es una técnica matemática que hace una cosa similar. Es decir transforma cualquier función en el dominio de tiempo en un espectro de frecuencia, algo así como un prisma que divide la luz en un espectro de colores. (Este es un ejemplo

analógico esto no es perfecto, sin embargo da a entender lo que pretende transformar la transformada de Fourier)

La Transformada de Fourier como Concepto Matemático

La transformada de Fourier como concepto matemático esta basado en el descubrimiento en el cual es posible el tomar cualquier función periódica tiempo $\mathbf{x(t)}$ y resolverla en una suma infinita equivalente de senos y cosenos con frecuencias que empiezan en $\mathbf{0}$ y se incrementan en múltiplos de enteros de una frecuencia base $\mathbf{f_0 = 1/T}$, donde $\mathbf{T}$ es el periodo de $\mathbf{x(t)}$. Aquí esta como luce la expansión:

$$x(t) = a_0 + \sum_{k=1}^{\infty} \left[a_k \cos\left(2\Pi k f_0 t\right) + b_k sen\left(2\Pi k f_0 t\right) \right]$$

Una expresión de la forma que se encuentra del lado derecho de la ecuación es llamada serie de Fourier. El trabajo de la transformada de Fourier es el resolver todos los valores a_k y b_k para producir una serie de Fourier, dada la frecuencia dada y la función $x(t)$. Puede pensar en el termino a_0 fuera de la sumatoria como el coeficiente para $k=0$. No hay correspondencia al seno del coeficiente b_0 en la frecuencia cero por que el seno de cero es cero, luego entonces tal coeficiente no tendrá ningún efecto.

Por supuesto, nosotros no podemos hacer sumas infinitas de cualquier cosa en una computadora real, así que tenemos configurarla para un cierto set finito de senos y cosenos. Esto nos lleva a algo posible que podemos hacer fácil con una entrada digital de muestreo, cuando nosotros estipulamos que habrá el mismo número de muestras de frecuencia a la salida tantas como las hay de muestras de tiempo a la entrada. También somos afortunados que todas las grabaciones de audio tengan un numero finito de longitud. Entonces; Notros podemos pretender que la función $x(t)$ es periódica, y este periodo es el mismo que el de la longitud grabada. En otras palabras, imagina la grabación repetida por siempre, y llama esta función $x(t)$. La duración de las secciones repetidas define la base de la frecuencia f_0.

La frecuencia base esta definida por:

$$f_0 = rango\ de\ muestreo\ /\ N$$

Donde **N** el numero de muestras en la grabación.

Un ejemplo podría ser el siguiente: Si nosotros nos encontramos utilizando un rango de grabación de 44100 muestras por segundo, y lo largo de la grabación es de 1024 muestras, la cantidad de tiempo representado por la grabación es de 1024/44100=0.02322 segundos, así que la base de la frecuencia será 1/0.02322= 43.07 Hz. Si procesamos estas 1024 muestras con el FFT, la salida será los coeficientes del seno y del coseno a_k y b_k para las frecuencias 43.07Hz, 2*43.07Hz, 3*43.07Hz. Para verificar si la transformada esta funcionando correctamente, se debe generar todas estas frecuencias multiplicarlas por sus respectivos coeficientes a_k y b_k, sumarlas todas juntas, y nosotros tendremos devuelta la grabación original.

CAPITULO I

I PRINCIPIOS Y GENERALIDADES

El objetivo de un convertidor analógico digital es el generar una palabra digital la cual sea representativa a la magnitud de una señal analógica. Fig.1.1.

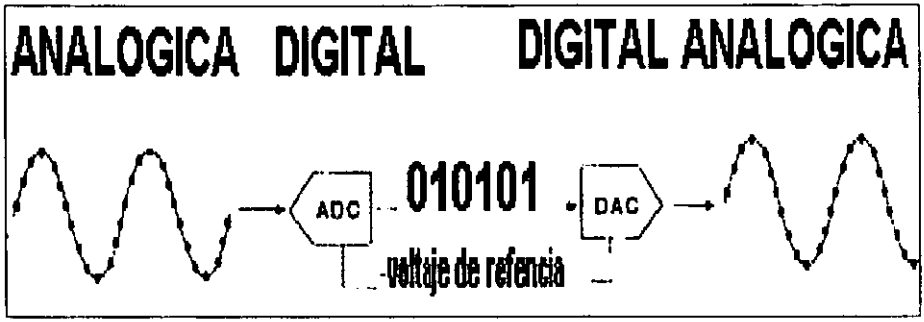


Fig.1.1.

I.1.0. Convertidor Analógico Digital - CAD

Un Convertidor Analógico-Digital (CAD) - ***Analog to Digital Converter ADC*** es un cuantizador- ***quantizer***. Fig.1.2. El CAD mapea constantemente cambios en la señal analógica en escalones y cada escalón está representado por un código digital. Este mapeo habilita la señal para ser almacenada o procesada en una forma digital. Los escalones discretos representan una porción arreglada de valor del voltaje de referencia – ***reference voltage*** que esta siendo utilizado.

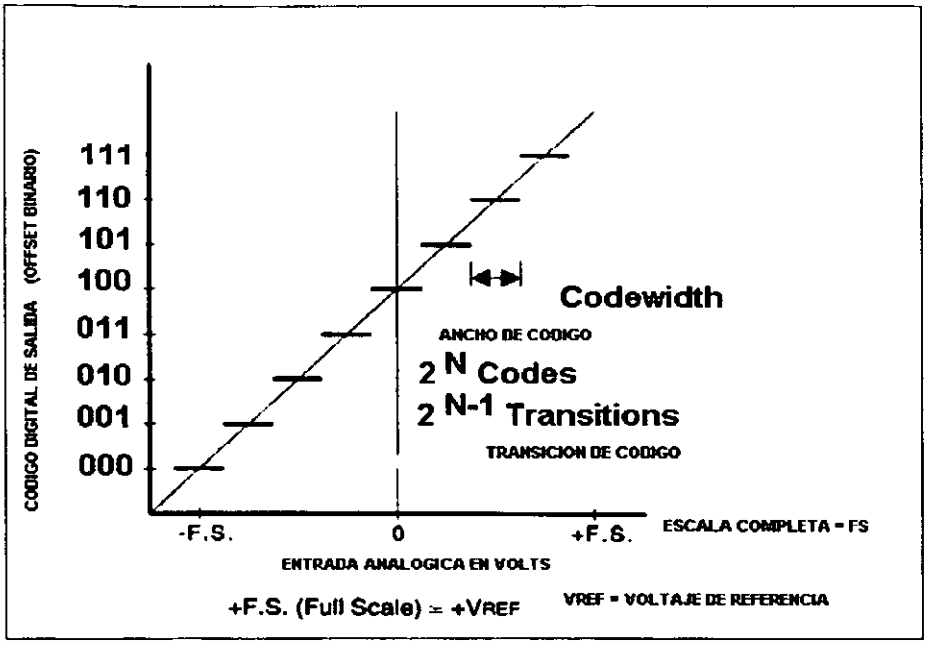


Fig.1.2.

I.2.0. Errores en el Convertidor Analógico Digital

Un CAD tiene 2^N códigos, N es el número de bits en el convertidor. El uso práctico es el de definir el centro de uno de los códigos como punto cero del convertidor. Por definición un convertidor bipolar tendrá un código menor en una mitad de la función de transferencia que en la otra. Debido al Escala Completa - ***Full Scale*** es un hecho que el código de salida describe un punto 1 del Bit Menos

Significativo(BMS) – ***Least Significant Bit (LSB)*** el cual es menor que el valor del voltaje de referencia.

Sabiendo que un Convertidor Analógico Digital CAD es un dispositivo cuantizador - ***quantizing,*** entonces este habilita el error de cuantización - ***quantization.*** *Fig.1.3.* Inclusive un CAD ideal exhibe un error de cuantización. Una vez que una señal haya sido digitalizada, el código digital representa un valor específico analógico el cual puede diferir de la señal analógica actual tanto como *½ LSB.*

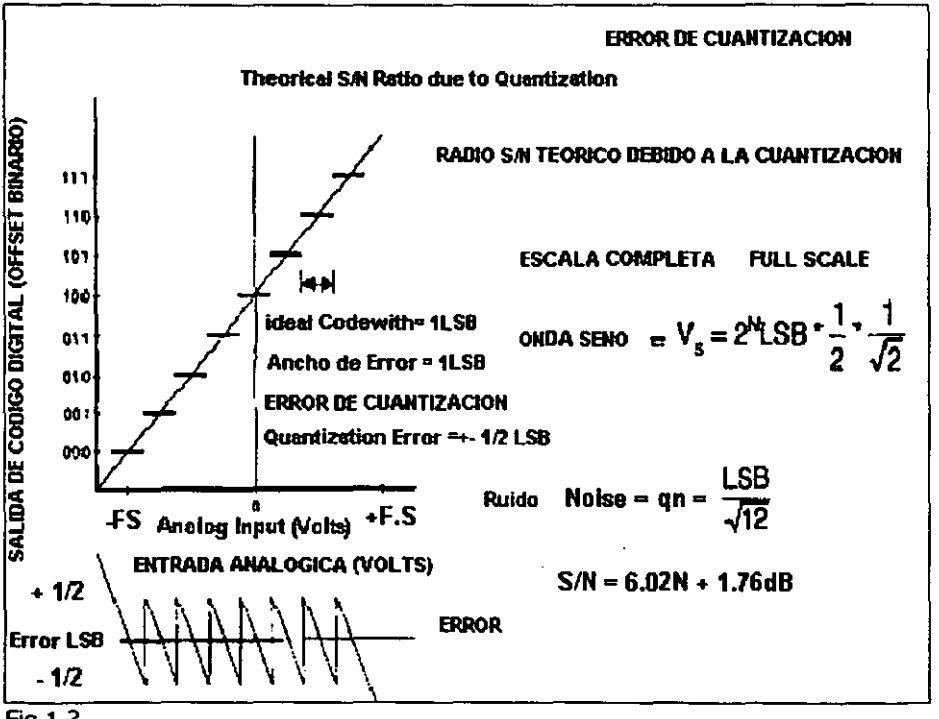


Fig.1.3.

Un CAD ideal tiene un ***S/N*** teórico el cual es definido por ***S/N* = 6.02N + 1.76 dB** (N es el número de bits del convertidor).

Cada código de la conversión representa una porción pequeña del voltaje total de entrada. Esto ayuda a examinar exactamente cuánto voltaje analógico representa cada código en la conversión. Por ejemplo si el ***Full Scale*** es de +-3V, para varias resoluciones el valor específico del voltaje disminuye, cuando la resolución aumenta. Vea ejemplo siguiente y Cuadro 1.1. Para un convertidor de 16 Bits o mayor, el voltaje representado por una cuenta va siendo menor con relación a fuentes de error potencial en el circuito, tal como el ruido y *la* desviación por temperatura-***Thermal drift.***

Ejemplo: ¿Qué tan grande es un *Count/LSB*?

Un Convertidor Analógico Digital a *Full Scale* =+-3V

$$\frac{\textbf{6 Volts}}{2^{N}\ \textbf{Códigos}}$$

N=	12	16	20	24
1-Count/LSB=	1.46mV	91.5uV	5.7uV	0.36uV

Cuadro 1.1.

El número de bits en un convertidor ideal determina su resolución y su Radio Señal/Ruido RSN - ***Signal/Noise Ratio SNR.*** Cuadro 1.2. La resolución define la pequeña porción de la señal a *Full Scale* la cual puede ser resuelta. El

Signal/Noise Ratio define el radio de la onda senoidal (rms) del *Full Scale* elevado al ruido *(rms)*. La resolución y el *S/N* pueden ser mejorados utilizando un convertidor de más bits.

N Bits del CAD	# de Códigos	Resolución	S/N ideal
12	4096	224 ppm	74 dB
16	65536	15 ppm	98 dB
20	1,048,576	0.95 ppm	122 dB
24	16,777,216	0.06 ppm	146 dB

Cuadro1.2.

1.2.1. Error Diferencial No Lineal

En un mundo real en un convertidor Analógico – Digital, el tamaño de los escalones de cuantización, pueden ser mayores o menores que el ideal. Esto introduce un error llamado ***differential nonlinearity (DNL)*** – diferencial no lineal (DNL). Fig 1.4. El *DNL* es una medida del tamaño actual de un código contra un código ideal. Una diferencia no lineal *DNL* mayor tenderá a reducir la efectividad ***Signal/Noise*** del convertidor.

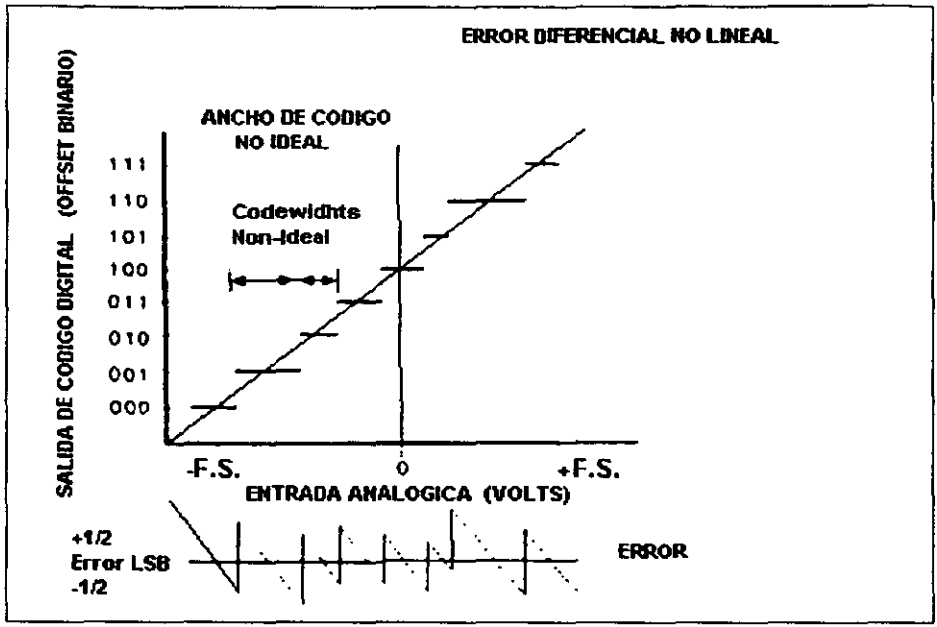


Fig.1.4.

La variación en el tamaño de los códigos es principalmente determinada por la exactitud como se conjuguen los elementos que integran el convertidor. Conjugaciones grandes con errores pueden dar como resultado códigos anchos o angostos, e inclusive pueden resultar pérdidas de código en alguna arquitecturade convertidores.

I.2.2. Error de Integración No Lineal

La Integración No Lineal – ***Integral Nonlinearity (INL)*** *(también llamado error de integración no lineal)* es una medida de linealidad de la función

de transferencia completa. Fig.1.5. El método de medición para un INL puede variar entre fabricantes. Unos utilizan el término " ***endpoint linearity*** - " linealidad punto final " el cual indica una línea recta dibujada entre los códigos en cada fin de la función de transferencia. INL es la medida de la distancia desde esta línea hasta el punto centro del código, en esta medición la distancia más alejada será la medida errónea. Algunos fabricantes utilizan el método línea recta ***"best fit"*** - "mejor ajuste". Este método produce una medida optimista de linealidad.

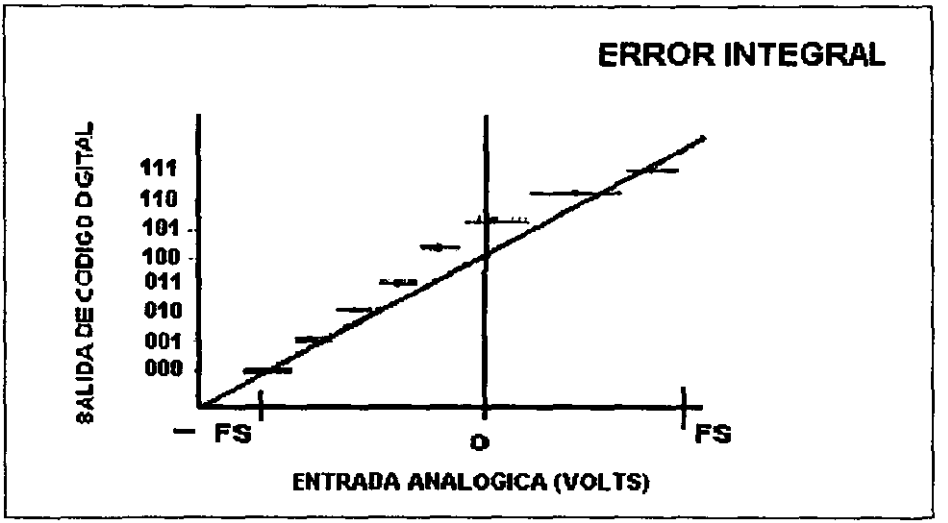


Fig.1.5.

La especificación Integración No Lineal es comúnmente encontrada en una hoja de especificaciones del CAD, sobre todo en aplicaciones en Instrumentación. En la medición del espectro o en la aplicación del proceso de la señal, el sobrepasar la linealidad de un ADC, usualmente se expresa en la forma

Señal/(ruido + distorsión)-***Signal/ (noise+distortion).*** Este parámetro esta determinado al aplicar una señal seno a FS con baja distorsión dentro de convertidor y utilizando el ***DFT (Discrete Fourier Transform)*** para determinar las características de la Densidad del espectro en el convertidor. ***El Signal/Noise Ratio y el Signal/(Noise + Distortion) Ratio*** pueden, entonces, ser calculados de los Datos de la Densidad Espectral.

I.2.3. Error de *Offset*

El error por *offset* es el punto cero de la función de transferencia. Fig.1.6.

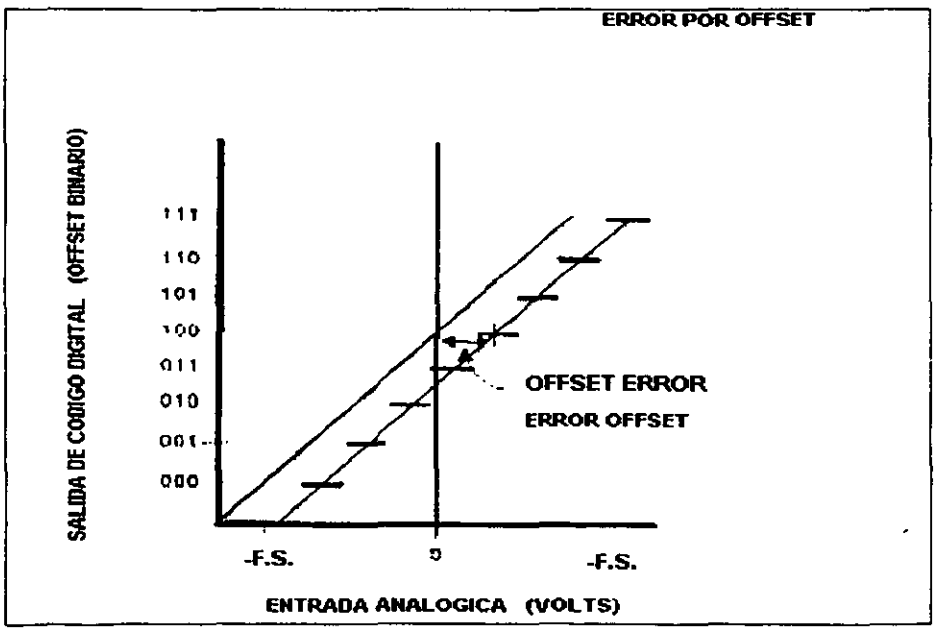


Fig.1.6.

I.2.4. Error de Ganancia

El error por ganancia es un cambio en la pendiente de la función de transferencia del convertidor relativo al estado del voltaje de referencia. Fig.1.7

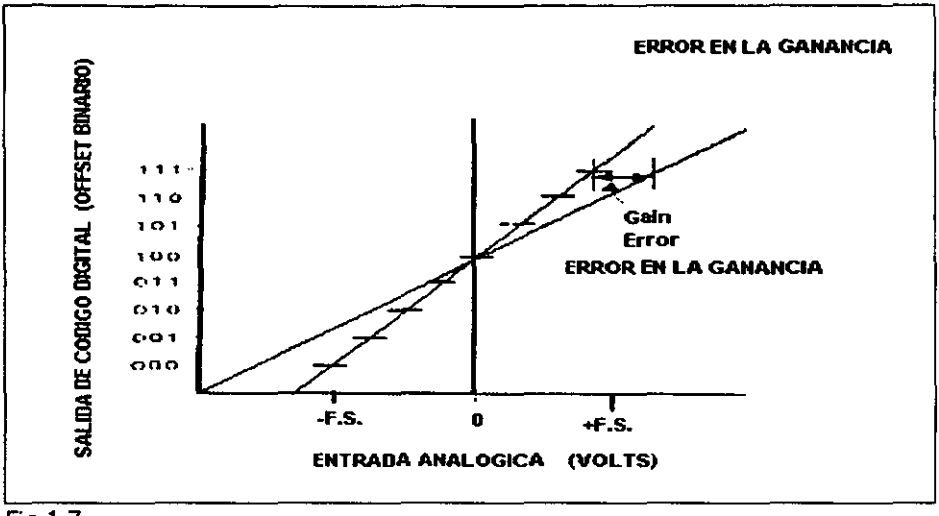


Fig.1.7.

I.3.0. Efectos en el Convertidor Analógico Digital

Jitter

La interferencia en la señal al ser medida puede degradar el desarrollo del convertidor. El ***jitter*** en el reloj muestreador puede degradar los resultados en el proceso de conversión. Una vez que una señal es convertida a digital, esta señal queda determinada por 2 variables, principalmente por un *set* de palabras

digitales, y por un período de muestreo T. En el dominio digital, T es un valor acordado y es exactamente el mismo en cualquier proceso realizado sobre los datos, es decir es múltiplo. Si en el proceso de muestreo, T varía por una cantidad A, el valor analógico que un CAD captura está siendo muestreado en un tiempo equivocado. El CAD debe convertir con exactitud el valor muestreado, en una palabra digital.

El hecho de haber muestreado la señal en Tiempo T-A, es decir si nuestro reloj tiene *jitter*, da como resultado que se obtenga, un valor de palabra correcto, pero con un tiempo de muestreo equivocado. Al tener un reloj sin *jitter* se observa que si no existe variación que afecte, en el tiempo T, los valores de las palabras no serán iguales al valor de las palabras digitales que se obtiene cuando se utiliza un reloj que tiene *jitter*. Fig.1.8.

El *jitter* es una modulación de la frecuencia del reloj de muestreo. Esta modulación es denominada como aleatoria, si en la ocurrencia del borde del reloj de muestreo tiene una distribución gaussiana parecida al significado ***"jitter free"*** – "libre de *jitter*". Luego entonces este resultado será un incremento en el nivel del ruido. Si el reloj tiene componentes con frecuencia específica, por ejemplo una modulación de un 1 kHz del reloj del muestreo, el *jitter* presenta la señal de entrada en bandas contiguas en la transición de subida y bajada (también por arriba y por debajo) en la cual la amplitud total de la señal se relaciona con la

amplitud del *jitter*. El *jitter* al parecer, en este pequeño lapso, es inaceptable, así como si no estuviese relacionado con las armónicas en la entrada.

La mayoría de los cristales osciladores lograrán un *jitter* menor a los 10 picosegundos pico a pico si la fuente de voltaje del oscilador tiene un pasabandas para mantener el ruido fuera de la fuente del oscilador. Es admisible, usualmente, desacoplar el oscilador con una resistencia de 10 ohms y un capacitor pasabandas.

Los divisores de frecuencia pueden también aumentar el *jitter* en el reloj. Si es necesario hacer el *jitter* lo menor posible, se debe utilizar C.I. *Flip Flop* duales y fuentes desaclopadas, en vez de utilizar C.I.. multiestado, contadores o divisores de frecuencias monolíticos.

Una combinación de análisis teórico, simulaciones por computadora y medidas prácticas, ha permitido la predicción confiable de efectos audibles de *jitter* en el reloj de muestreo. La resolución y el rango dinámico de los ADC se incrementan pasando los 16 bits. Además el *jitter* del reloj de muestreo será más significante. Existen técnicas de análisis que permiten saber el nivel máximo de *jitter* en el reloj y pueden ser bien determinadas.

Ha sido demostrado que el sobremuestreo en un CAD delta-sigma no es muy susceptible a los efectos del *jitter* en el reloj en comparación con la arquitectura de los CAD con muestreo Nyquist.

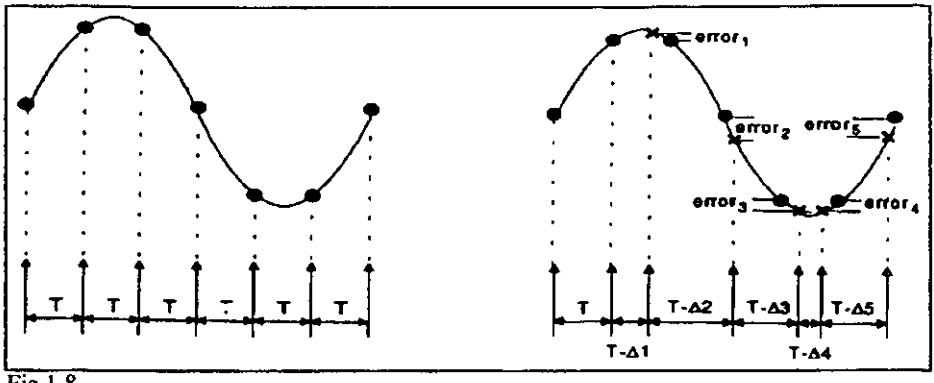


Fig.1.8.

Jitter

Son las variaciones de tiempo en el periodo de muestreo.

La Amplitud de *Jitter* – Variación de Tiempo del periodo ideal Δ.

La frecuencia del *Jitter* – rango de cambio del tiempo de muestreo ideal.

Muestrear la señal en el tiempo erróneo T +-Δ da como resultado errores de conversión de la muestra capturada precisamente en el tiempo T.

Una vez que en el Dominio digital los periodos de muestreo son siempre T. Decimos que el resultado es "Muestreo erróneo en el tiempo correcto".

I.4.0. *Aliasing* en el Dominio del Tiempo

Si se muestrea una señal a una frecuencia menor que el doble de su frecuencia máxima, causa *aliasing.* Aquellas frecuencias que son mayores que la mitad del rango de muestreo aparecen como frecuencia, las cuales son menores que la mitad del rango de muestreo. La siguiente gráfica se muestra el

comportamiento cuando f = 1/N * fs y f = (N+1)/N * fs, teniendo como dominio el tiempo. Fig.1.9.

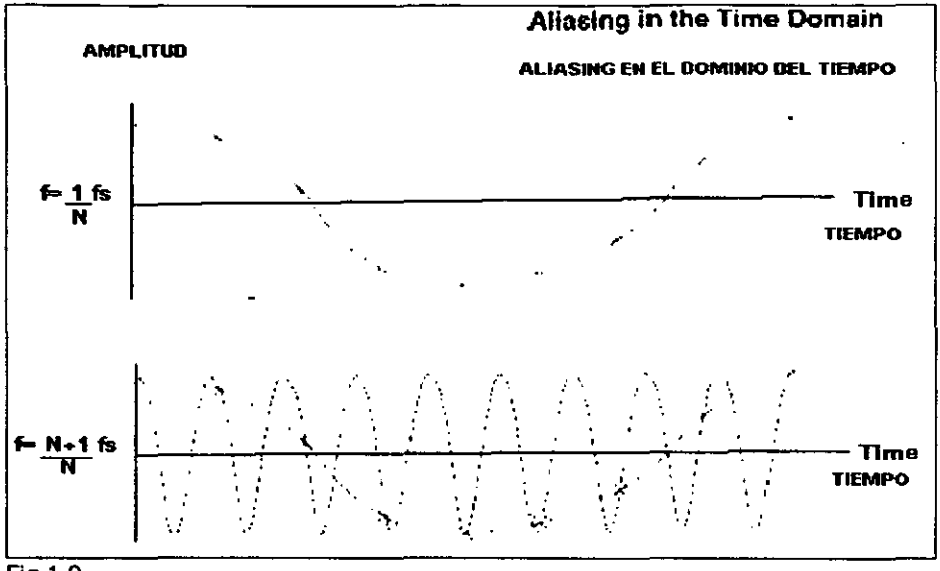


Fig.1.9.

I.5.0. *Aliasing* en el Dominio de la Frecuencia

En la siguiente figura se puede observar el ***Aliasing*** en el dominio de la frecuencia. Aquí se puede observar el mismo efecto en el dominio de la frecuencia. Fig.1.10. Muestrear una señal causa una copia del espectro de entrada para aparecer cerca de la frecuencia de muestreo. La copia replica el espectro de

entrada original a fs/2, y cualquier componente de amplitud significante será notado por debajo de la frecuencia de piso.

Esto puede ser utilizado para hacer un desplazamiento de una banda de frecuencia alta a una banda de frecuencia baja. Normalmente, pensamos que el diseñador quiere deshacerse del *aliasing*, siempre que el doblez de la información de alta frecuencia corrompa la señal dentro de la banda deseada.

La siguiente gráfica ilustra un convertidor muestreando a 48khz, un rango de frecuencia de muestreo típico en audio digital. Fig.1.10. Para alcanzar una medida de frecuencia apropiada, la señal de entrada deberá estar limitada a frecuencias menores que los 24 khz. Prevenir los componentes del espectro por debajo de la mitad del rango de muestreo, puede ser difícil. En este ejemplo, el sistema desea un ancho de banda fuera de los 22khz, pero deben de prevenir el *aliasing* de las frecuencias por debajo de los 24khz.

Para alcanzar el ancho de banda de 22khz con una atenuación de 80 dB de frecuencias a 24khz o mayores requiere de un filtro con banda de transición de sólo 2 khz de ancho y una pendiente mayor que 2100 dB/década. Para esto se necesitaría un filtro con más de 100 polos. Esto es impráctico para lograrlo con un filtro analógico *antialiasing*. La mayoría de los filtros *antialiasing* analógicos viables

y comerciales el día de hoy sólo tienen por lo menos una octava de ancho de onda para alcanzar el corto máximo, sólo alcanzan 60 dB de atenuación. Estos utilizan normalmente filtros con topología elíptica y exhiben un rizo en el paso de banda y en la zona de corte de la región del filtro.

Gracias a estas limitaciones de los filtros analógicos, puede ser un beneficio el muestrear a mayor frecuencia que el doble de la frecuencia del ancho de banda de la señal.

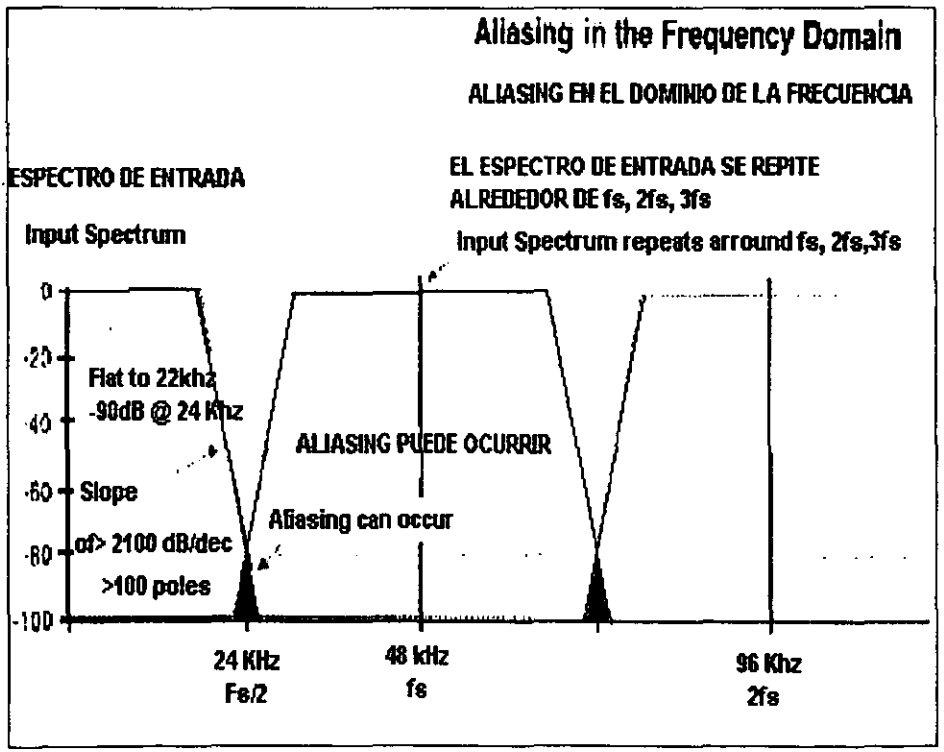


Fig.1.10.

I.6.0. Espectro de Sobremuestreo

Si el rango de muestreo para un ancho de banda dado esta aumentado, la complejidad del filtro de *antialiasing* puede reducirse. En nuestro ejemplo anterior donde se tenía que utilizar 100 polos muestreando a una frecuencia de 22khz, si muestreamos a 48 khz y el rango de muestras se dobla a 96khz, el filtro *antialias* tendrá mayor el ancho de banda hasta alcanzar una atenuación de 80dB; Los 100 polos del filtro requeridos es ahora reducido a 12 polos. Fig.1.11.

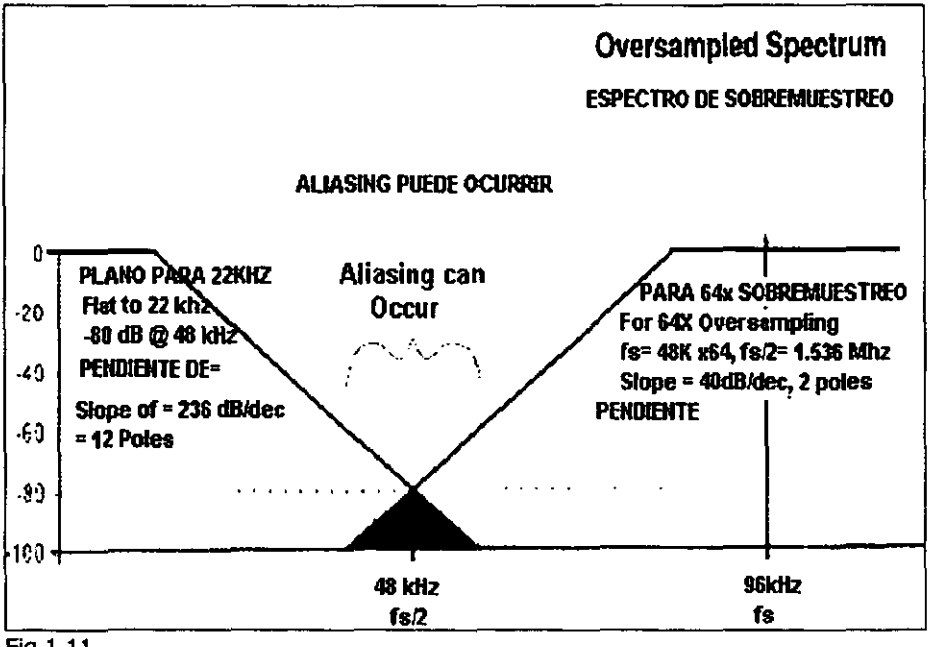


Fig.1.11.

El sobremuestreo entonces reduce la complejidad del filtro *antialias*. Este es uno los beneficios de las claves del sobremuestreo. Así pues también es común que para los convertidores Analógicos Digitales los cuales utilizan el sobremuestreo como parte de su arquitectura para sobremuestrear por radios de 64X, 128X, e inclusive tan alto como 512X. Con 64X del sobremuestreo, el filtro requerido del ejemplo de audio es reducido a un filtro de dos polos.

I.7.0. Error de Apertura

El error de apertura es definido como los errores de amplitud y tiempo de los puntos de datos muestreados debido a la incertidumbre de los cambios dinámicos de los datos durante el muestreo. En sistemas de adquisición de datos y conversión, el error de apertura puede ser reducido o ser insignificante utilizando ya sea con muestreador o con un convertidor A/D rápido.

Para los datos de una señal seno, el máximo error de apertura ocurre en el cruce cero donde los grandes dv/dt ocurren, y es expresado matemáticamente de la siguiente forma:

$$\text{Error de Apertura} = \frac{d\,(A \operatorname{sen} 2\pi ft)}{dt} \times t_A \times 100\,\%$$

$$= 2\pi f t_A \times 100\%\ \text{max}$$

donde:

f = frecuencia de datos máxima

t_A = tiempo de apertura del sistema (este puede ser el tiempo de conversión de un convertidor A/D sin retenedor/sostenedor *sample/hold*, o el tiempo de apertura de un *sample/hold* si uno esta enfrente de un convertidor A/D).

Esta expresión es mostrada gráficamente en la figura 1.12. para frecuencias de 10Hz a 1 Mhz con +- 1/2LSB de error para varias resoluciones de convertidores A/D.

La necesidad de *sample/hold* se puede leer aparentemente cuando la frecuencia de datos de 10Hz o más grande es muestreada, por que la velocidad de conversión del convertidor deberá ser de 2us o más rápida en errores de apertura menores de +-1/2*LSB* para una resolución de 12 bits. Los convertidores A/D de

alta velocidad son más complicados y caros que los convertidores más lentos con una apertura baja de *sample/hold*.

Por ejemplo un *sample/hold* con un tiempo de apertura de 50ns a 60ns produce un error de apertura insignificante para frecuencias arriba de los 100 Hz con una resolución de 10 y 12 bits, y es menor de +-1/2*LSB* con una resolución de 8 bits para frecuencias cerca de los 5kHz. Se puede analizar la figura 1.12 para determinar el error de apertura del sistema para cada canal de datos vs la resolución deseada.

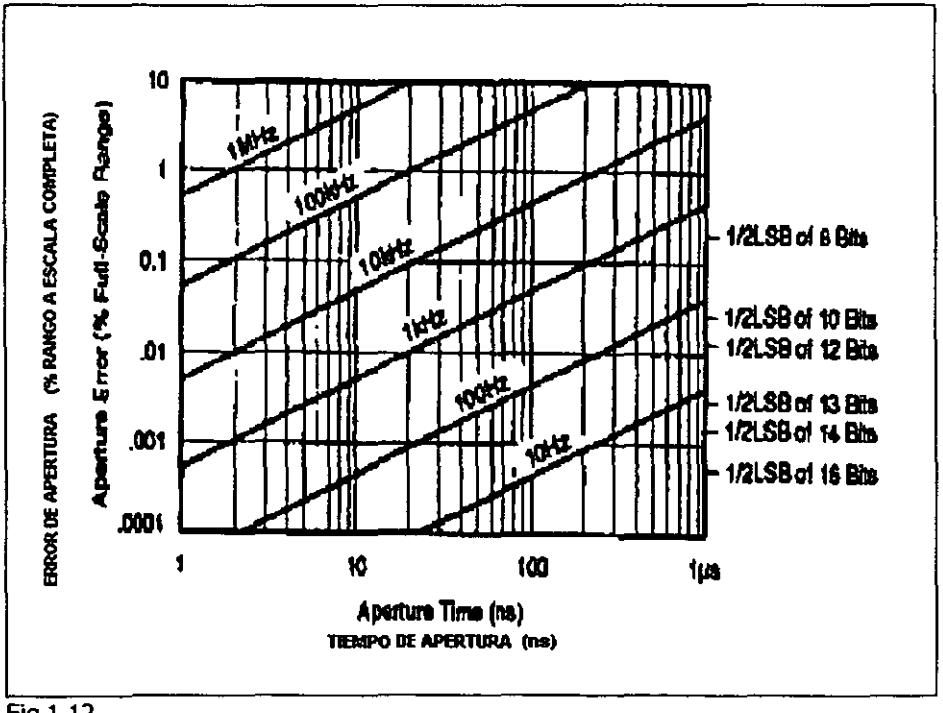


Fig.1.12.

I.8.0. Teorema de Nyquist

Un mínimo de dos muestras por ciclo en el ancho de banda de la señal de datos, son requeridas en un sistema ideal de muestreo de datos, para producir datos muestreados sin perdida de información.

I.9.0. Resolución

El número de bits en un convertidor Analógico Digital determina la resolución del sistema. La resolución del sistema esta definido de la siguiente manera:

Resolución = Un $LSB = V_{FSR}/2^n$, para un convertidor analógico Digital.

LSB = Least Significant Bit – Bit menos significativo.

V_{FSR} = ***Full Scale Input Voltage Range*** – Rango del voltaje de entrada de la escala completa.

El número de bits define el número de códigos digitales , existen 2^n códigos digitales discretos para un CAD, donde hay n= número de bits.

En el siguiente cuadro se encuentra la relación de los valores *LSB* de un convertidor A/D y la resolución de los números binarios. Cuadro.1.3.

Resolución del Convertidor A/D (Código Binario)		Valor de 1 LSB		Valor de un ½ LSB	
Número de Bits (n)	Número de Incrementos (2^n)	Rango de 0 a +10 V (mV)	Rango +-10V (mV)	Rango de 0 a +10V (mV)	Rango +-10V (mV)
16	65536	0.152	0.305	0.076	0.152
12	4096	2.44	4.88	1.22	2.44
11	2048	4.88	9.77	2.44	4.88
10	1024	9.77	19.5	4.88	9.77
9	512	19.5	39.1	9.77	19.5
8	256	39.1	78.2	19.5	39.1

Cuadro.1.3.

I.10.0. Rango de Velocidad en la Conversión en la colocación final *(Troughput)*

El rango de colocación final *Troughput* de un sistema esta determinado por la configuración de los tiempos requeridos en el multiplexor analógico, el

amplificador de entrada, el tiempo de adquisición en *sample/hold*, la configuración del CAD y el tiempo de conversión.

El método más común utilizado para describir la adquisición de datos y la exactitud en la conversión del sistema es el calcular en una computadora ***la raíz de la suma de los cuadrados*** de cada uno de los errores que componen el sistema- ***the root-sum squared (RSS).*** Este error *RSS* tiene un valor estadístico, el cual puede ser calculado por la desviación estándar (1σ), y representa la raíz cuadrada de la suma de los cuadrados de los picos de los errores de cada componente del sistema, incluyendo el error de cuantización ADC:

$$\varepsilon RSS = \sqrt{\varepsilon MUX^{2} + \varepsilon AMP^{2} + \varepsilon SH^{2} + \varepsilon ADC^{2}}$$

Donde:

εMUX = Error en el multiplexor analógico

εAMP = Error en la entrada del amplificador

εSH = Error en el *Sample/Hold* (muestreador/sostenedor)

εADC = Error en el Convertidor Analógico Digital

La impedancia de la fuente, el ancho de banda, la resolución del convertidor analógico – digital y el rango de *troughput* del sistema afectan los cálculos de estos errores. Para simplificar, los errores pueden ser calculados asumiendo que un error es insignificante:

1.- Si es menor de que 1/10 *LSB.*

2.- Si la impedancia de la fuente es menor que 1000 ohms.

3.- Si el rango de la señal es de +-10 volts.

En el Cuadro.1.4. se observa la contribución del error del sistema, donde se presenta el error *RSS* vs la resolución para un sistema típico de adquisición de datos.

Error de la Fuente	Resoluciones		
	8 Bits	10 Bits	12 Bits
Error del MUX	0.0025 %	0.0025 %	0.0025 %
Error del AMP	0.0100 %	0.0100 %	0.0100%
S/H Error	0.0100 %	0.0100 %	0.0100 %
Errores analógicos de cuantización en el ADC	0.2000 % 0.2000%	0.0500 % 0.0500 %	0.0120 % 0.0120 %
Error RSS	0.2830 %	0.7200 %	0.0220 %

Cuadro 1.4.

En la Figura 1.13. pueden observar el rango de *Troughput* de (a) Una Multiplexación Serial Programada (b) Un translape de Multiplexación.

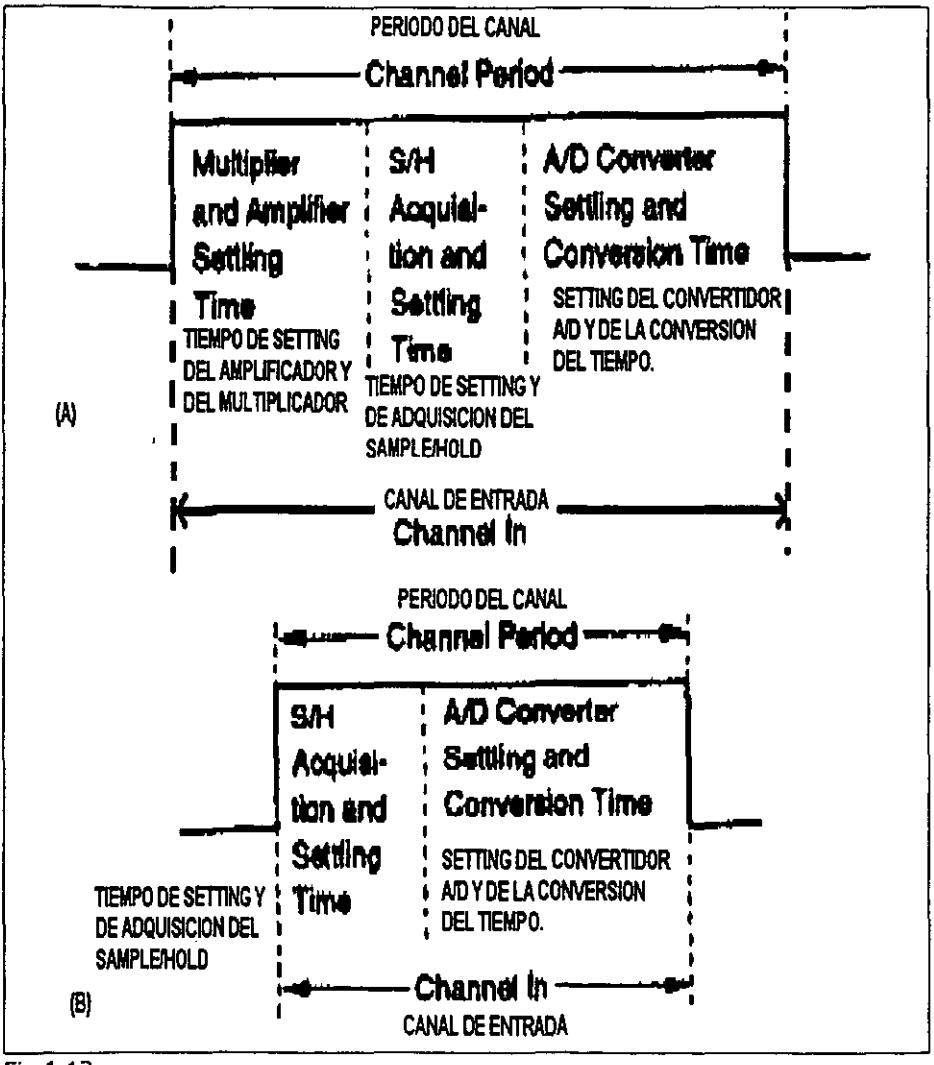


Fig.1.13.

CAPITULO II

II TIPOS DE ARQUITECTURA DEL CONVERTIDOR ANALOGICO DIGITAL

Existen muchas arquitecturas del Convertidor Analógico Digital, cada uno tiene sus beneficios. Esta gráfica ilustra la resolución típica y el *troughput– velocidad en la conversión* del Convertidor Flash, Aproximaciones Sucesivas y el alfa sigma ΛΣ.Fig.2.1.

El convertidor Flash sirve para digitalizar las altas frecuencias, y así lo hace pero con muy poca resolución.

El convertidor de Registro de Aproximaciones Sucesivas (RAS) - *Successive approximation register´s (SAR)* sirven para un rango de frecuencia intermedia, regularmente utilizado en aplicaciones con multiplexores para digitalizar muchos canales con información de baja frecuencia.

El Convertidor $\Lambda\Sigma$ es un convertidor de sobremuestreo y se desarrollan efectivamente con frecuencias por debajo de los 200khz del ancho de banda.

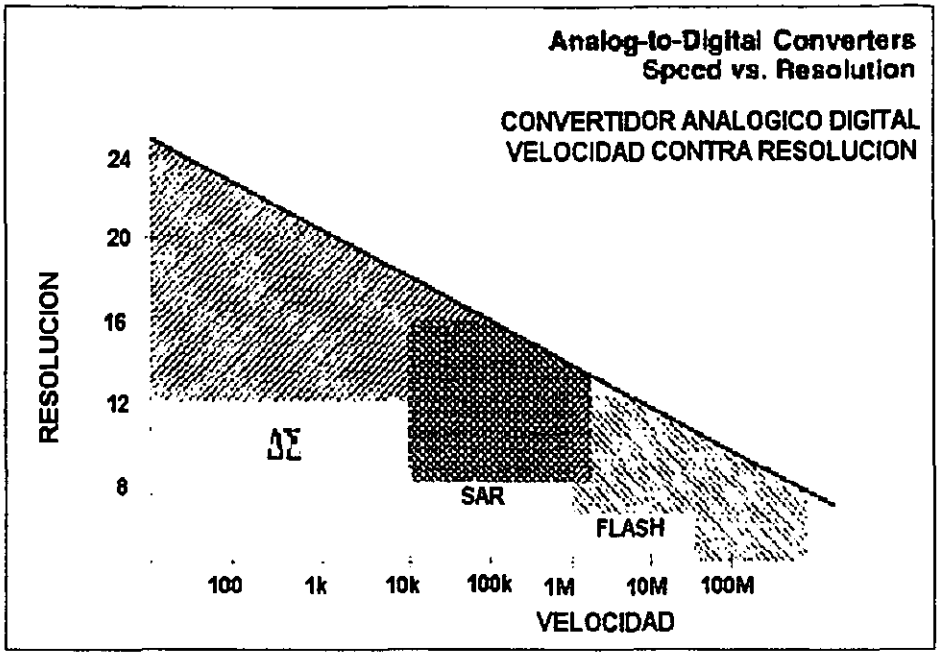


Fig.2.1.

II.1.0. Convertidor *Flash*

El convertidor Flash de 8 bits que operan con rangos de frecuencia de muestreo, arriba de los 100 Mhz son comunes. Fig.2.2. El convertidor flash esta disponible en diferentes arquitecturas. El convertidor Flash paralelo es uno de los convertidores más común.

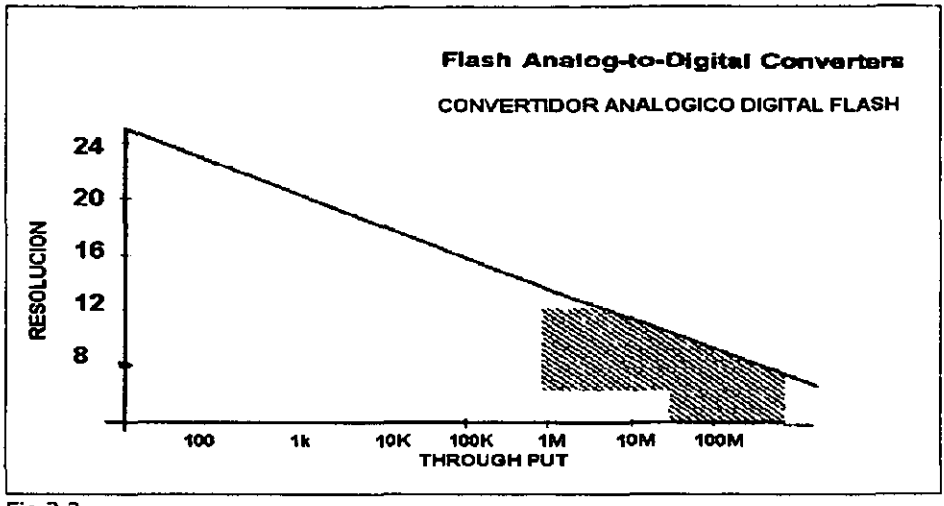


Fig.2.2.

II.1.1. Convertidores *Flash* Paralelo

La arquitectura *Flash* es una de los métodos más rápidos de conversión analógico digital. En la tecnología *Flash* la entrada analógica esta simultáneamente aplicada a un banco de comparadores. Un comparador tiene un único voltaje de referencia, derivado de un generador de voltaje de referencia. Los niveles están igualmente espaciados, cada vez que el voltaje de entrada se incrementa, más comparadores son encendidos. El número de comparadores los cuales se encienden es proporcional a la magnitud del voltaje de entrada. Las salidas del

comparador están alimentando un decodificador, donde (2^N)-1 entradas crean una palabra digital de N-bits. Fig.2.3.

Un convertidor *Flash* analógico-digital puede completar una conversión en un ciclo de reloj. Los rangos de conversión de la arquitectura paralela se extienden hasta los 100´s de Mhz. La última velocidad del flash depende de los retardos del comparador más los tiempos de retardo en el decodificador. Velocidades mayores pueden ser alcanzadas a expensas de un consumo mayor de potencia. En el Flash paralelo, la resolución es mayor implicando un consumo de potencia mayor, de tal manera que los Flash paralelos requieren (2^N)-1 comparadores y niveles de voltaje de referencia para alcanzar las N-bit salidas del convertidor.

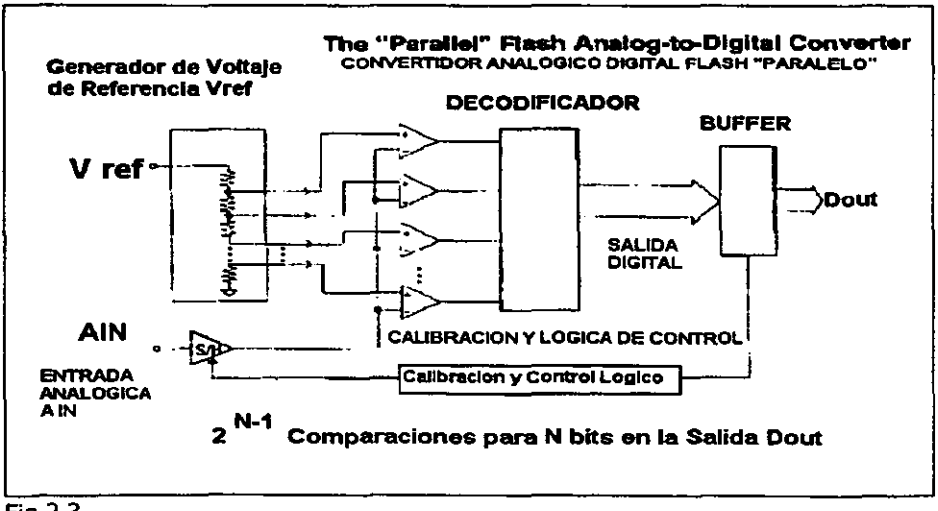


Fig.2.3.

II.1.2. Convertidores *Flash Subranging y Pipelined*

La arquitectura *Flash subranging y pipelined* utilizan menos comparadores que los *Flash* paralelos para la misma resolución, y para velocidades equivalentes, teniendo menor consumo de potencia. Fig.2.4.

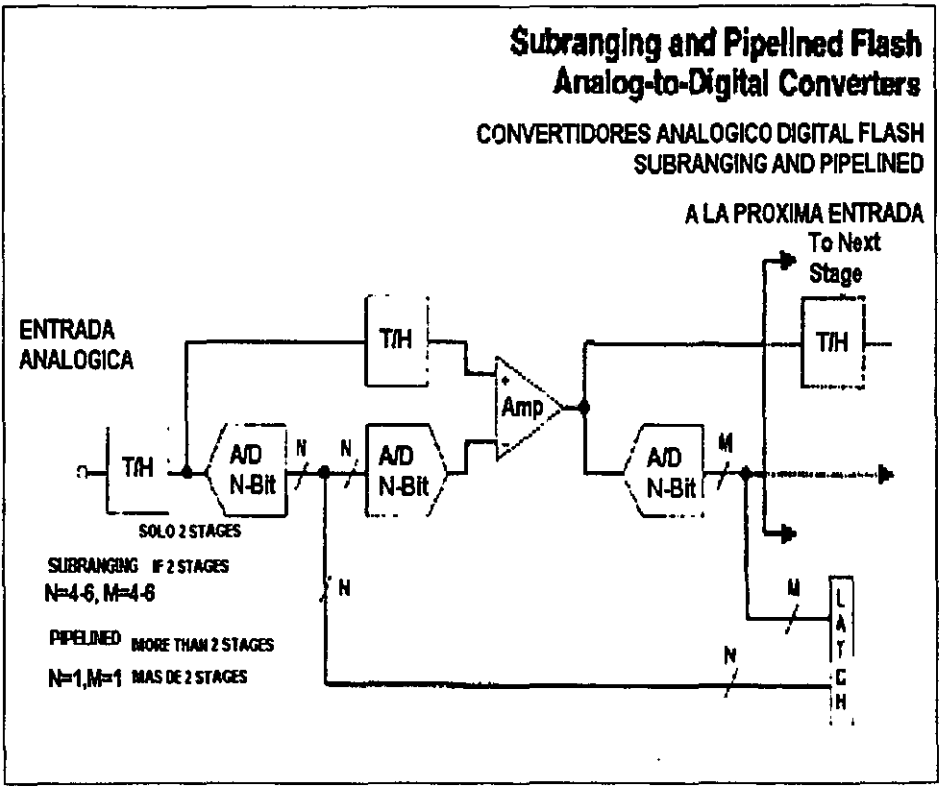


Fig.2.4.

El convertidor *subranging* usa 2 estados, cada uno tiene de 4 a 6 comparadores. La conversión resultado del primer estado es reconstruida utilizando un *Converter Digital Analog (CDA)* - Convertidor Digital Analógico (CDA). La salida del CDA es entonces obtenido de la entrada analógica. El voltaje diferencial es amplificado y la entrada llega al segundo estado de la conversión.

Si la arquitectura utiliza una aproximación *"feedback"*, la diferencia amplificada es realimentada para utilizar el primer estado del convertidor una segunda vez. Si la arquitectura utiliza una aproximación *"feedforward"*, la diferencia amplificada continúa la alimentación hacia otro estado del convertidor. En la arquitectura *subranging*, las palabras de salida de cada uno de los dos pasos de la conversión, son combinadas para generar la palabra de datos de salida del convertidor.

La arquitectura *pipelined* es similar a la aproximación *subranging*, pero utiliza un solo bit comparador para cada estado. N comparadores son usados para obtener un convertidor analógico-digital. La arquitectura *pipelined* utiliza un número menor de comparadores, pero tiene que utilizar al menos un ciclo de reloj para acompañar cada bit de la conversión. Por esta razón, este tiene un mayor tiempo de retardo que la arquitectura en paralelo.

II.2.0. Convertidor de Aproximaciones Sucesivas

Los convertidores Analógicos digital con registro de aproximaciones sucesivas están disponibles con una variedad amplia de atributos en su desarrollo. Algunos son optimizados por velocidades mientras que otros son optimizados para baja potencia. Por esta razón, el rango de los *SAR´s* en velocidad se pueden adquirir desde el rango de los KHz a los MHz. Mientras los *SAR´s* de 12 bits son más comunes, los *SAR´s* de 16 bits se están convirtiendo en la norma para los nuevos diseños. Fig.2.5.

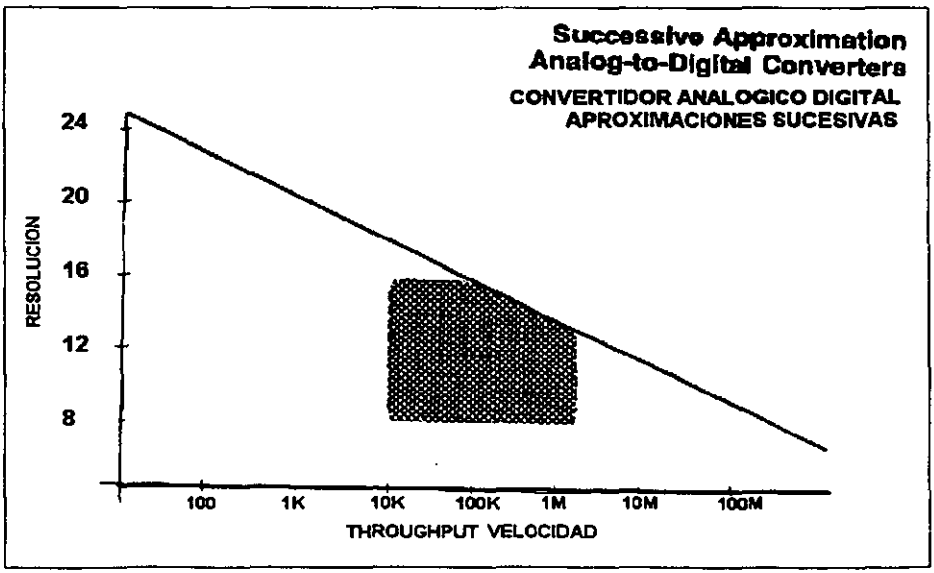


Fig.2.5.

En la Arquitectura de aproximaciones sucesivas, la entrada analógica es sucesivamente comparada con la salida del CDA, controlado por el algoritmo de conversión. La aproximación sucesiva empieza comparando la entrada analógica a la salida del CDA el cual esta configurado en "encendido (1 lógico)" a media escala del *FS (bit más significante (Most Significant Bit (MSB))* y todos los demás bit en "apagado (0 lógico)". Si la entrada es encontrada por estar por debajo de la escala media del *FS*, el *MSB* es puesto en *reset* a cero, y la entrada es comparada con un cuarto de la escala FS (próximo *MSB* encendido, todos los demás apagados). Si la entrada estuviese por debajo de la mitad de la escala, el *MSB* permanecería encendido y la próxima comparación será a tres cuartas partes de la escala completa *FS*. Este procedimiento continúa hasta que todos los bits hayan sido ejercidos.

II.2.1. Arquitectura para Balancear la Carga en un Convertidor de Aproximaciones Sucesivas.

La mayoría de los convertidores analógicos-digitales de aproximaciones sucesivas utilizan arreglos de capacitivos con peso binario. Una de las ventajas de

la construcción a base de arreglos de capacitores es la inherente función muestreador/sostenedor *sample and hold* del arreglo de capacitores.

Ya existen en el mercado convertidores CAD *SAR´s* de 16 bits con auto calibración, basados en capacitores. Los *SAR´s* basados en capacitores, comparten un nodo común al el comparador de entrada. Sus otras terminales se pueden conectar a la entrada analógica, al voltaje de referencia, o a tierra. En los convertidores basados en arreglo de capacitores, el proceso de conversión consiste en manipular los platos libres del arreglo de capacitores hacia el voltaje de referencia o tierra. El primer paso en el algoritmo de aproximaciones sucesivas es el conectar el capacitor *MSB* al voltaje de referencia y los otros capacitores a tierra. Esto forma un divisor de voltaje en el cual el nodo central es la entrada al comparador. El comparador hace una decisión entre si la entrada de voltaje capturada en el arreglo de capacitores es mayor o menor que la mitad de la escala. Continuando el algoritmo de aproximaciones sucesivas implica el manipular las conexiones de los capacitores para ir através de los diferentes radios de los divisores capacitivos, y examinar el resultado del voltaje con un comparador. Fig.2.6.

El convertidor *SAR* tiene una función de transferencia multipunto en la cual la exactitud depende de los elementos de los cuales esta hecho el CDA. Errores de *ratio* en el tamaño de los elementos CDA causa errores (*DNL-difeerential*

Nonlinearity) Diferencial No lineal e *(INL-Integral Nonlinearity)* Integral No lineal; y puede terminar en una pérdida completa del código.

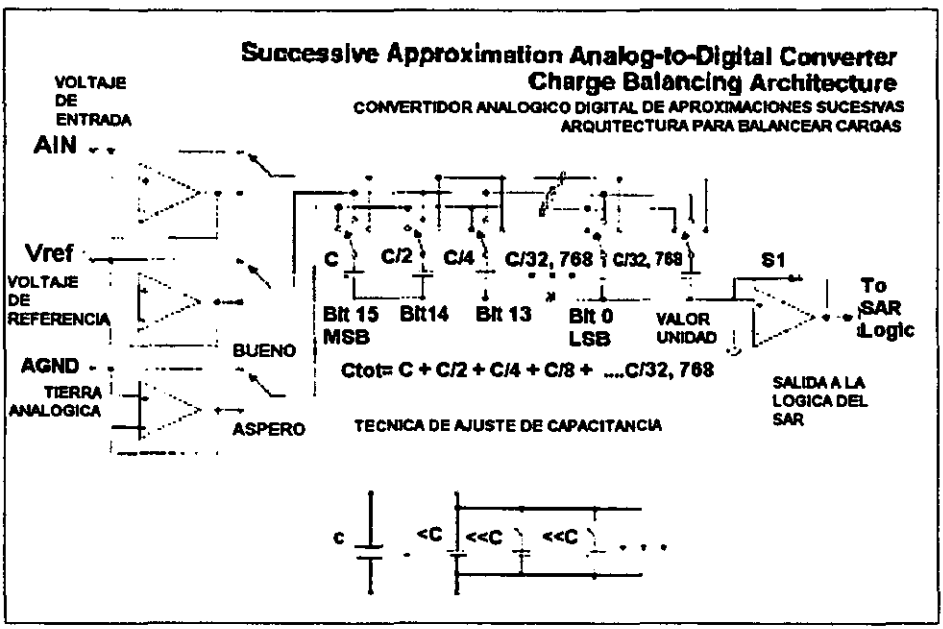


Fig.2.6.

Es típico que los *SAR´s* con arreglos capacitivos usen convertidores R- 2R, para inyectar una corrección de voltaje dentro del nodo de suma, del arreglo de capacitores. Esto es realizado para que el arreglo de capacitores corrigen los errores de peso.

II.2.2. El Sobremuestreo para Mejorar el Rango Dinámico

En la siguiente gráfica podemos observar el espectro de muestreo de un convertidor analógico-digital. El ruido debido a la cuantización esta a lo largo entre Corriente directa (CD) y fs/2. Fig.2.7.El espectro de muestreo de un convertidor es también repetido como múltiplos del rango de la frecuencia de muestreo, como se muestra en la figura.

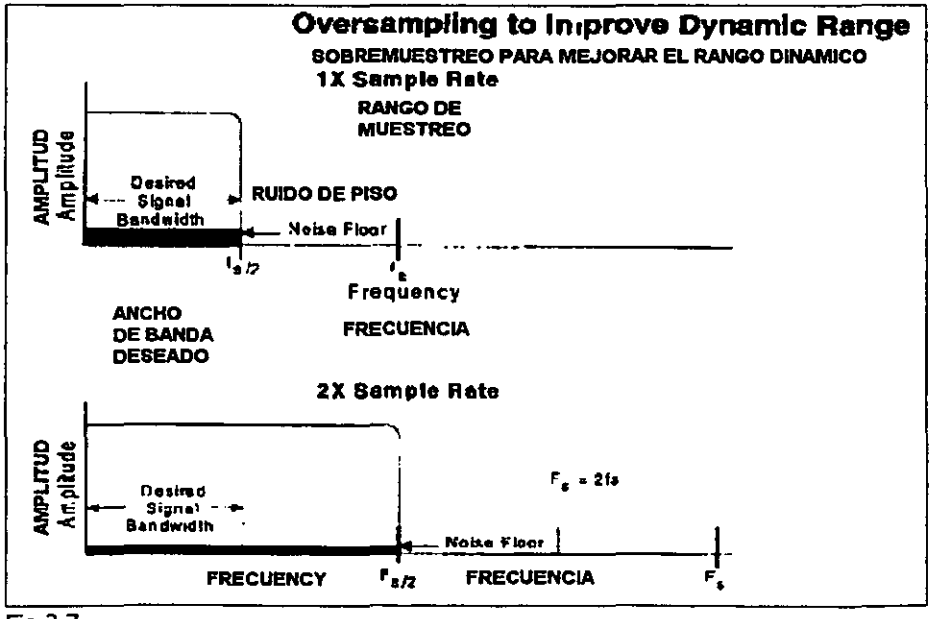


Fig.2.7.

Si la frecuencia de muestreo es el doble tanto como el ancho de banda, el ruido debido a la cuantización se extiende tanto como el doble del ancho de banda. Bajo esta condición la potencia del ruido por unidad de frecuencia será reducida a la mitad.

Una vez que la señal haya sido sobremuestreada, un filtro digital puede ser usado para postprocesar las muestras del convertidor. Si el ancho de banda de la señal esta limitado a la mitad del ancho de banda del convertidor *Nyquist*, el *signal/noise ratio* puede ser definido alrededor de los 3dB. Fig.2.8. Notar que cada vez que el rango de muestreo es doblado, la potencia del ruido en un ancho de banda arreglado es reducida a la mitad, la cual da como resultado una mejora de 3dB en el *signal/noise*.

Para alcanzar una mejora en el S/N *ratio*, el sobremuestreo de datos deberá ser filtrado digitalmente para reducir el ancho de banda de la señal.

En el algoritmo del filtro digital, es deseable el sobremuestrear con *ratios* grandes porque esto incrementa más el número de muestras para ser procesadas.

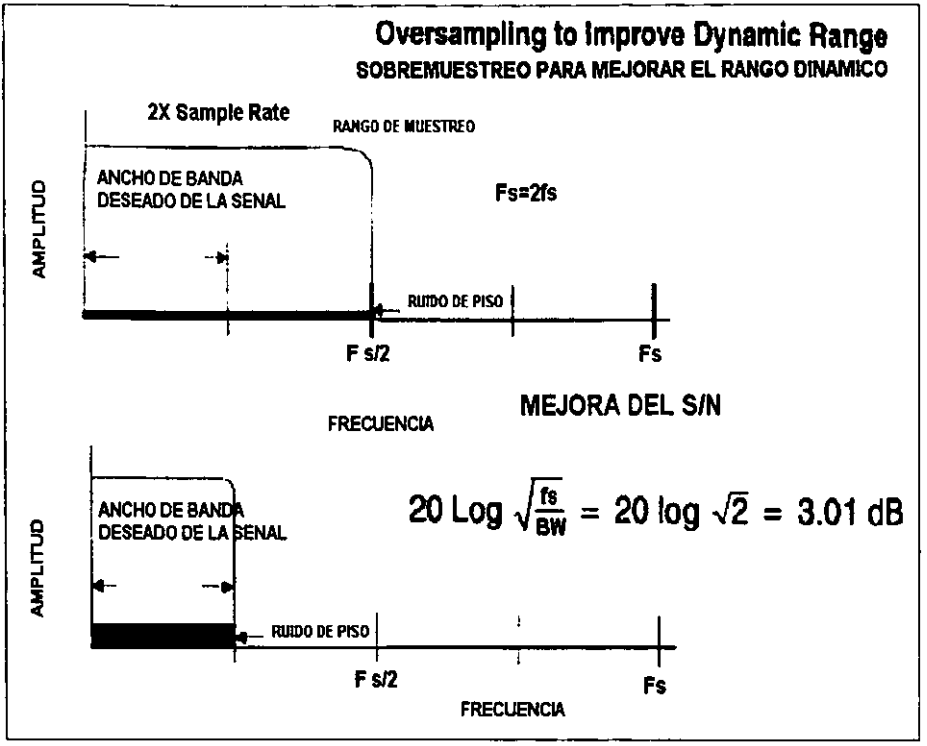


Fig.2.8.

Idealmente, es preferente no tener ruido en los convertidores analógicas digitales, pero cuando el promedio es utilizado para incrementar la resolución en las medidas, una cantidad correcta de ruido gaussiano es necesaria. Un sistema "sin ruido" no puede utilizar promedios para mejorar la medición de la resolución.

Deberán ser aleatorias las muestras que van a ser promediadas para desplegar un resultado en el cual se tenga una resolución mayor que la misma del convertidor.

La gráfica indica el ruido gaussiano del CS5102A, indicando la distribución de 8,192 conversiones. El significado indica que el promedio de 8,192 conversiones de palabra deberán caer 137.3 uV del *Least Significant Bit (LSB)* a la izquierda del centro de el código 8007 (H).

Existe un factor incierto basado en la cantidad de ruido y el número de muestras, pero la resolución medida no puede se mejorada a menos que haya algunas muestras aleatorias. Fig.2.9

Para las aplicaciones de medidas de Corriente Alterna (CA), el ruido (*rms*) deberá ser menor a la mitad del LSB o deberá ser la mitad del ancho de un código del convertidor. Si es menor la cantidad de ruido gaussiano que es utilizado como *dither*, el resultado será afectado por la distorsión.

Además podemos notar que los muestreadores de audio con un ruido gaussiano menor a la mitad del LSB suenan mejor, por que la distorsión debida al error en la cuantización esta enmascarada por ruido gaussiano.

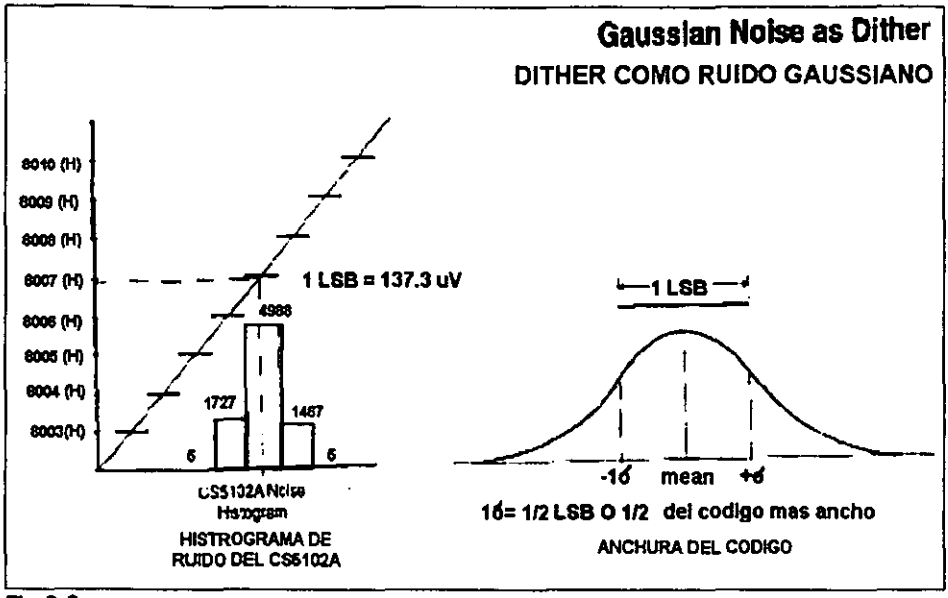


Fig.2.9.

II.3.0. Convertidores Analógicos Digitales $\Lambda\Sigma$

Los Convertidores Analógicos Digital $\Lambda\Sigma$ son utilizados en un rango amplio de aplicaciones, desde la Corriente Directa hasta los 500khz. A menores frecuencias esta arquitectura puede desplegar mayor resolución en la conversión.

Fig.2.10.

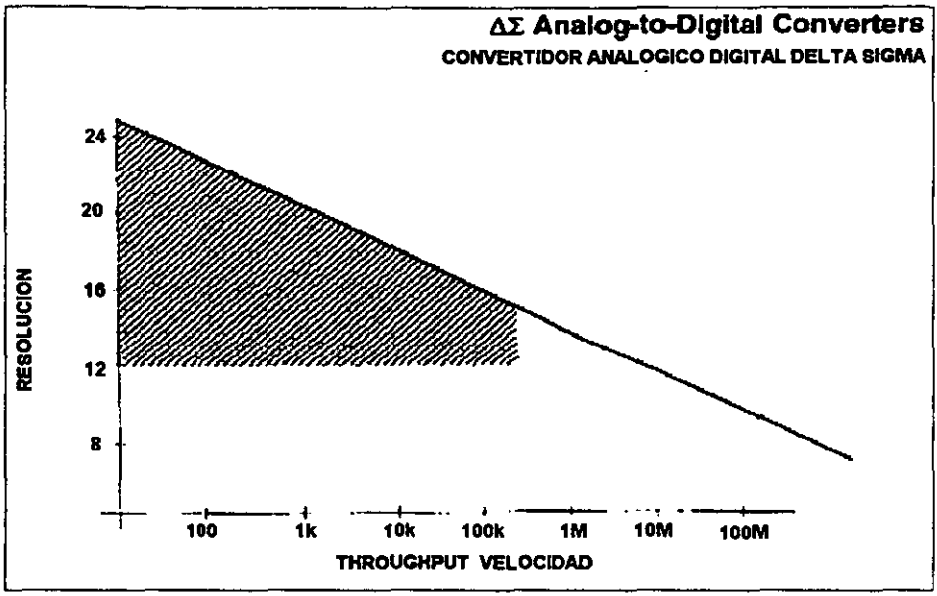


Fig.2.10.

El Convertidor $\Delta\Sigma$ utiliza el sobremuestreo como mecanismo fundamental para alcanzar una resolución alta. El convertidor incluye un modulador de sobremuestreo seguido por un filtro digital de diseño único. Fig.2.11.

El modulador esta usualmente compuesto de un muestreador de 1 bit corriendo a frecuencias de muestreo muy altas en relación con la señal que está siendo utilizada.

Los radios de sobre muestreo son comúnmente de 64, 128 y 256. Pero el sobremuestreo no es suficiente, el muestreador de 1-bit tiene un error de

cuantización muy alto a frecuencias de muestreo muy altas, consecuentemente tienen un ruido de cuantización muy alto, en relación con la señal de entrada. Para mejorar una señal más grande con respecto al ruido, el modulador de sobre muestreo incluye n filtro de *noise-shaping* el cual suprime el ruido de cuantización sobre el ancho de banda de la señal. Esto da como resultado un *cadena digital de bits - bit stream* al salir del modulador la cual ha sido filtrada espectralmente.

Esta cadena digital de bits es entonces procesada por un filtro digital la cual es diseñada para tener ventaja de la formación de ruido espectral. El filtro digital procesa la muestra de 1-bit, removiendo el ruido de cuantización *"out of band"* "fuera de rango" mientras que al mismo tiempo se incrementa la resolución del convertidor.

Los convertidores Analógicos Digital $\Lambda\Sigma$ utilizan el sobremuestreo, *noise shaping* y el filtrado digital juntos para poder implementar salidas de bajas velocidades y altas resoluciones de un muestreador de baja velocidad y baja resolución.

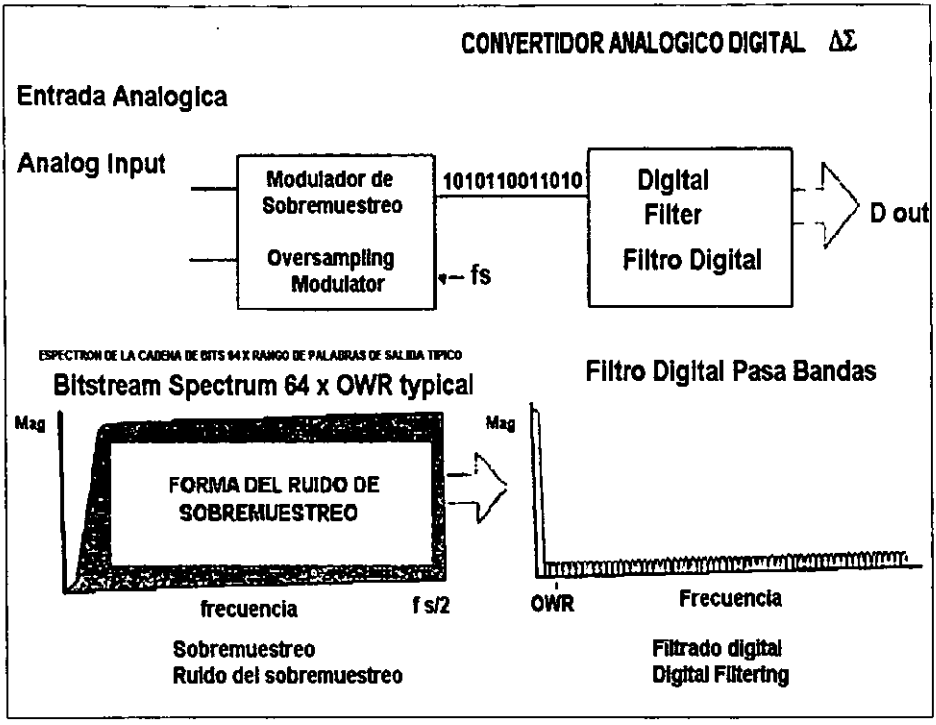


Fig.2.11.

II.3.1. El Modulador $\Lambda\Sigma$

El modulador $\Lambda\Sigma$ consiste en un amplificador diferencial, el cual mide la delta entre la señal de entrada y la realimentación de la señal del convertidor digital analógico de 1-bit. El delta es amplificada y se aplica como la entrada al

filtro, el cual consiste de varios estados de integración. Los integradores suman el error del voltaje, de ahí el nombre del sigma para el bloque del filtrado. Entre más grande sea el número de integrales en el filtro *noise-shaping* ruido-forma, mejor se puede suprimir la cuantización del ruido dentro del ancho de banda de interés.Fig.2.12.

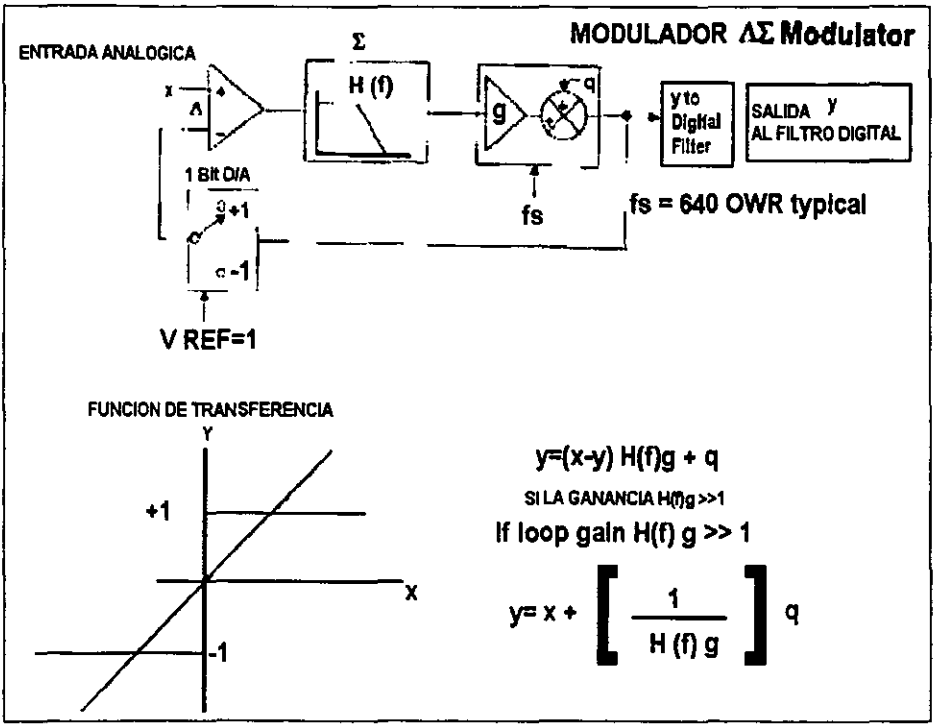


Fig.2.12.

Cada integrador adicional suma 6 dB de mejora en el *Signal/Noise* al doblar el rango de muestreo. Esto pasa porque cada integrador adicional aumenta el

desdoblamiento del filtro en 6dB por octava de frecuencia incrementada. El número de intregrales en el filtro *noise-shaping* indica el orden del modulador de un convertidor ΛΣ.

La salida del filtro *noise-shaping* es entones cuantizada. Utiliza un sólo comparador como el cuantizador de un convertidor analógico-digital ΛΣ. es común porque un comparador es un convertidor de *"one bit"*– "un bit". Un convertidor de 1 bit requiere no conjugar los componentes. Esto da como resultado 2 puntos de la función de transferencia en la cual la linealidad es inherente.

La salida del comparador da como resultado una cadena de un bit con una densidad de 1´s proporcional al radio de la señal de entrada y el voltaje de referencia de la realimentación Digital-Analógica. El convertidor analógico digital esta en un lazo de realimentación en la cual la salida esta tratando de minimizar el error del voltaje en las entradas del amplificador diferencial.

Cuando la señal de entrada va a positivo, el error del voltaje del amplificador diferencial causará que el convertidor Digital Analógico gaste más periodos de reloj sacando +1. Para que ocurran más 1´s se necesitará ser generados por el comparador. Si la señal va a negativo, solamente lo opuesto es verdad. Por lo tanto, la densidad de 1´s del modulador será proporcional a la señal de entrada.

La ecuación de la ganancia del modulador será proporcional a la señal de entrada. La ecuación de la ganancia del modulador indica como el filtro *noise-shaping* reduce el ruido de la cutización.

La cadena de bits del *noise-shaped* es después procesada por un filtro digital. El filtro digital esta diseñado junto con el configurador de ruido del modulador par dar como resultado el *Signal/Noise* ratio através del ancho de banda del filtro digital. El filtro rechaza estar fuera de la banda del ruido, en la cuantización.

El filtro digital determina principalmente el comportamiento del convertidor analógico digital $\Lambda\Sigma$. EL modulador se sobremuestrea a mayor frecuencias que las señales de interés sin embargo es diseñado para operar con un mínimo de efectos sobre la señal que esta siendo procesada.

Los Convertidores Analógica – Digital $\Lambda\Sigma$ utilizan sobremuestreo y *noise-shaping* seguido por un filtraje digital para alcanzar un rango dinámico alto. Esta gráfica indica el S/N ratio la cual resulta cuando se usa varios níveles de *noise-shaping* combinado con ratios de sobremuestreo alternados. Fig.2.13.

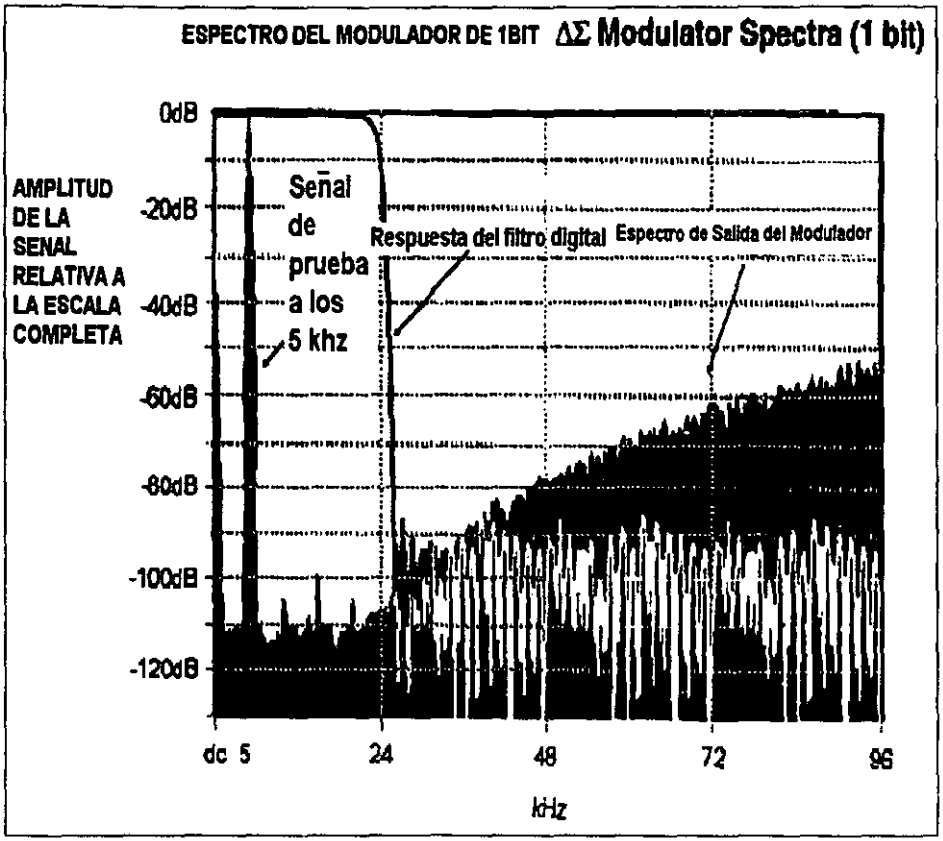


Fig.2.13.

Nota que para un número *S/N*, por ejemplo 100 dB, hay varias opciones de sobremuestreo y orden modular las cuales pueden alcanzar el número *S/N* deseado. También nota que doblando el reloj de muestreo con cualquier modulador de orden mayor (de tercer orden en adelante) significantementes se incrementa en tal radio S/N. Fig.2.14.

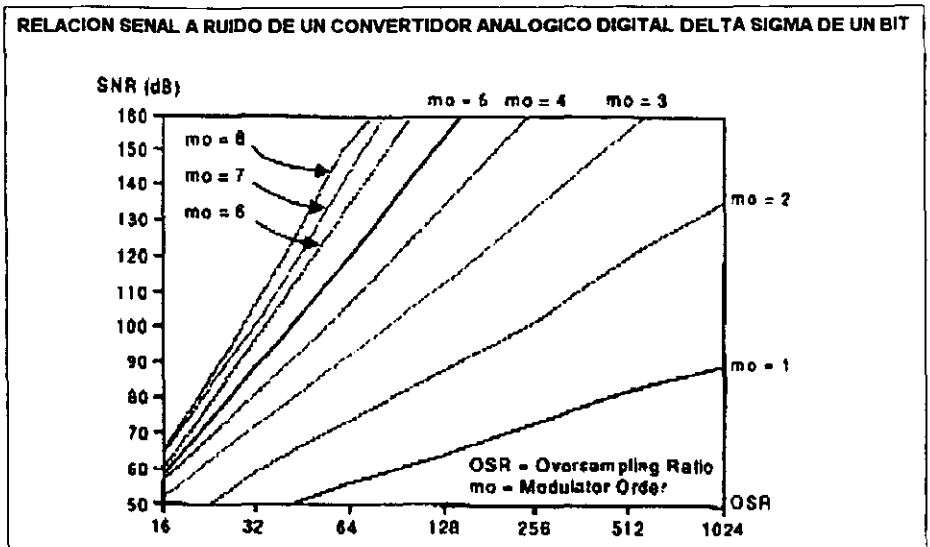


Fig.2.14.

La siguiente gráfica ilustra el espectro del modulador con una cadena binaria de 1-bit de salida. Nota que el *noise-shaping* suprime el ruido de cuantización solo sobre una porción pequeña del CD en el ancho de banda fs/2. Si el *signal/noise ratio* de este modulador fuera procesado en una computadora para este ancho de banda *Nyquist* (desde el CD hasta 1.536MHz), el resultado sería un convertidor de Analógico Digital de 1 bit. Fig.2.15.

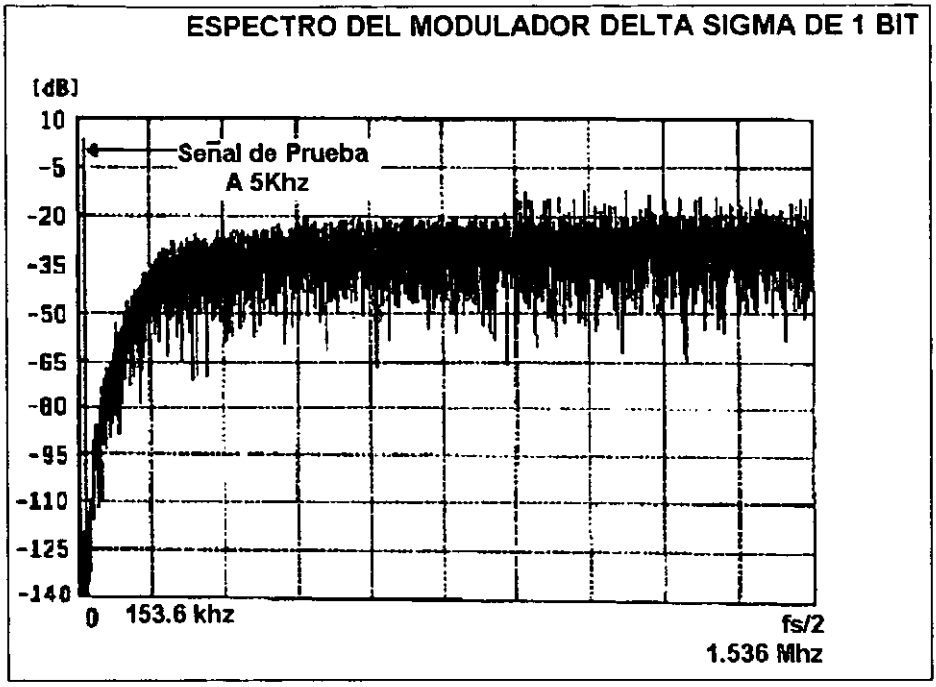


Fig.2.15.

II.3.2. Filtro Digital

En un Convertidor Analógico – Digital $\Lambda\Sigma$, el filtro digital domina el comportamiento de la señal del componente. El filtro digital determinará:

a) El ancho de banda de la señal.

b) Lo plano del pasabanda.
c) La respuesta de la fase sobre el pasabandas.
d) El retardo del grupo.
e) Las características de la banda de transición.
f) La atenuación del fuera de banda.

El filtro digital puede también ser diseñadas para prevenir el *aliasing*. En aplicaciones con entradas transitorias, el filtro dedicará las características de configuración del tiempo. Fig.2.16.

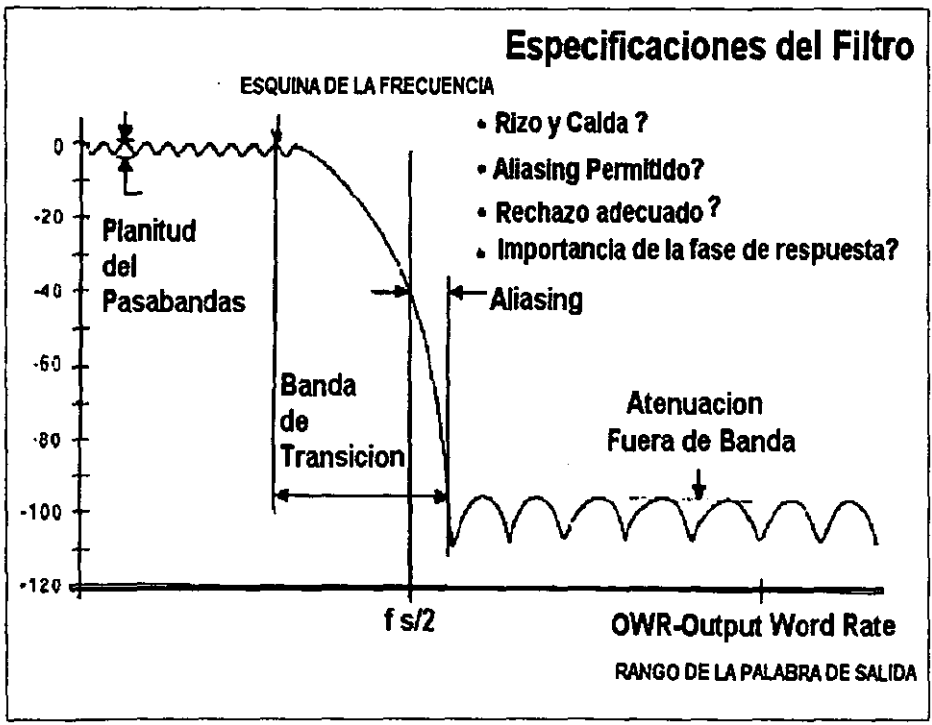


Fig.2.16.

Un filtro digital ofrece un número de ventajas sobre los filtros analógicos. Estos ofrecen más funciones de filtrado complejas a mucho menor costo. Un filtro analógico requerirá un número similar de polos y ceros si esto fuese para alcanzar un desarrollo comparable. Muchos de los filtros digitales alcanzan más de 1000´s de *taps* de filtraje Un filtro analógico de estas características no son realizables. Los filtros digitales ofrecen desarrollo de fase lineal lo cual da un retardo de grupo, y una particularidad muy deseable para aplicaciones de audio.

Un alto nivel de Integración soporta múltiple circuitos con las mismas características de filtro a filtro, en los cuales permanecen como constantes el tiempo y la temperatura. Cuadro.2.1.

Ventajas de los Filtros Digitales sobre los Filtros Analógicos
++Menor costo para una mayor complejidad
Fase Lineal / Retraso del grupo constante
Reproducción exacta
Estabilidad sobre el tiempo y la temperatura
Más funciones complejas que puedan ser leídas
Pueden ser utilizados para adicionar más desarrollo en las funciones DSP
Cambios de Tracks en el rango de muestreo

Cuadro 2.1.

II.4.0. Diagramas a Bloques de Convertidores Analógicos Digital $\Lambda\Sigma$

Existen diferentes requerimientos para Convertidores Analógico – Digital (CAD´s) diseñados para aplicaciones de instrumentación y audio. Fig.2.17.

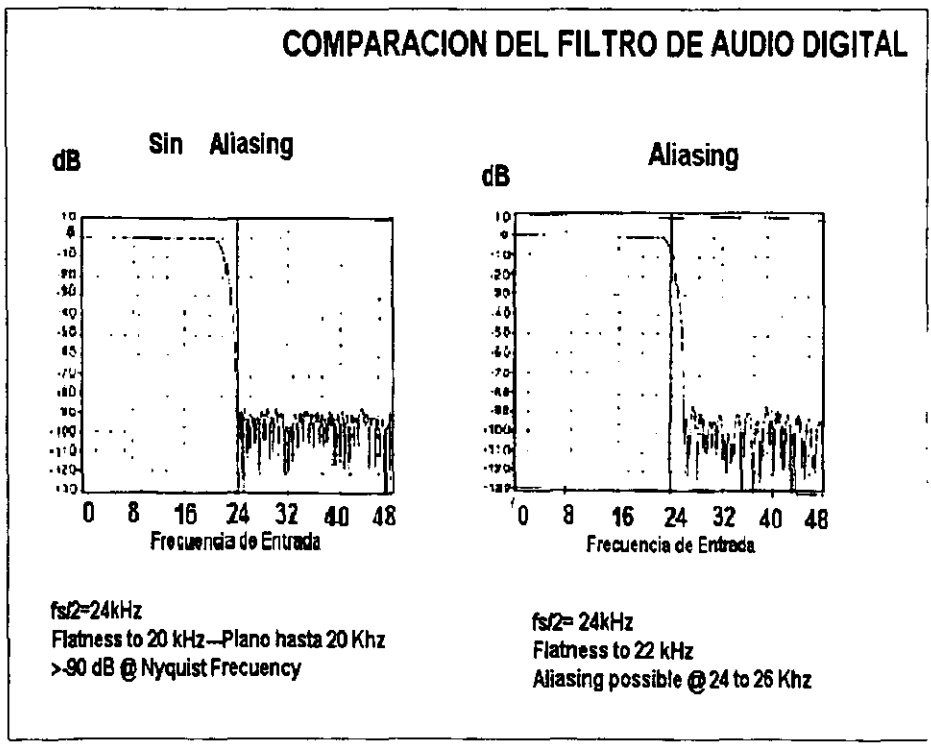


Fig.2.17.

Las aplicaciones de instrumentación generalmente requieren de la capacidad para medir con exactitud la Corriente Directa la cual puede ser lograda cuidando la estabilidad del voltaje de referencia y el control de los offsets. Los convertidores generalmente un chip externo como referencia de voltaje el cual permite al diseñador del sistema conocer los alcances en el desarrollo del sistema.

Muchos de los convertidores $\Lambda\Sigma$ para instrumentación utilizan *choppers* para estabilizar los *front-end* para eliminar los efectos de los *offsets* de voltaje generados internamente. El filtrado digital ofrece un número de ventajas sobre los filtros analógicos.

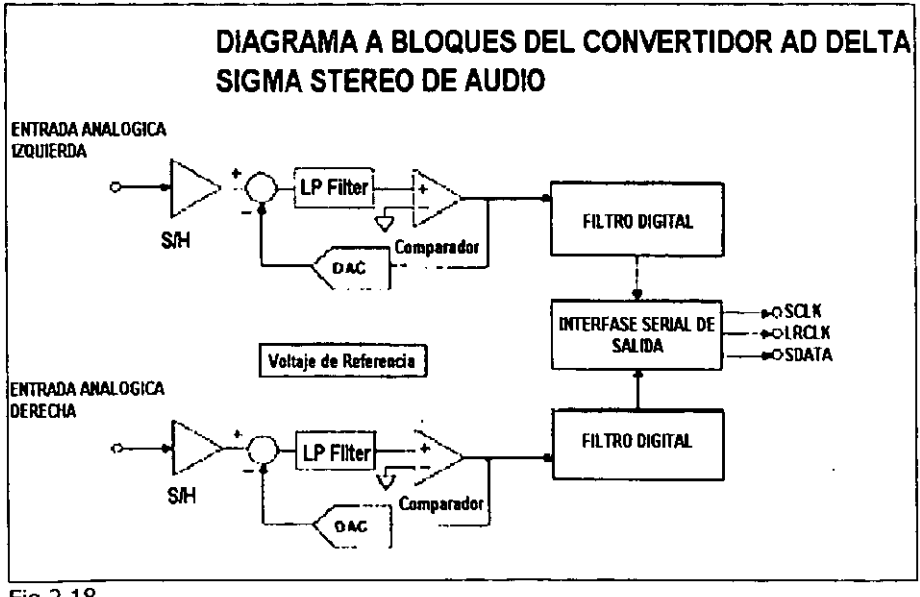


Fig.2.18.

El diagrama ilustra la configuración básica de un convertidor analógico digital de audio estereo. Fig.2.18. Los dos moduladores $\Lambda\Sigma$ comparten un voltaje de referencia común. Los moduladores alimentan separadamente filtros digitales. Los filtros digitales operan en paralelo uno con otro pero la salida de su conversión de datos es a través de un puerto serial común. Cuadro.2.2.

Convertidores A/D $\Lambda\Sigma$ para Instrumentación vs Audio	
Intrumentación	**Audio**
Choppers Estabilizadores de front-end	**Referencia Interna**
Voltaje de referencia externo al C.I.	**Calibración de Offset**
Calibración de Offset y Ganancia	**Algunos incluyen filtro pasa Altas para eliminar el Offset**
Lee y Escribe registros de calibración	**Filtro Digital - Optimizado para Audio**
Filtros Digitales - Optimizados para la instrumentación	

Cuadro.2.2.

Los datos es una salida de datos seriales utilizando una línea de datos (SDATA), una línea serial para el reloj (SCLK), una línea de reloj derecho-izquierdo (LRCLK). LA LRCLK indica qué palabra filtrada esta saliendo. Fig.2.19.

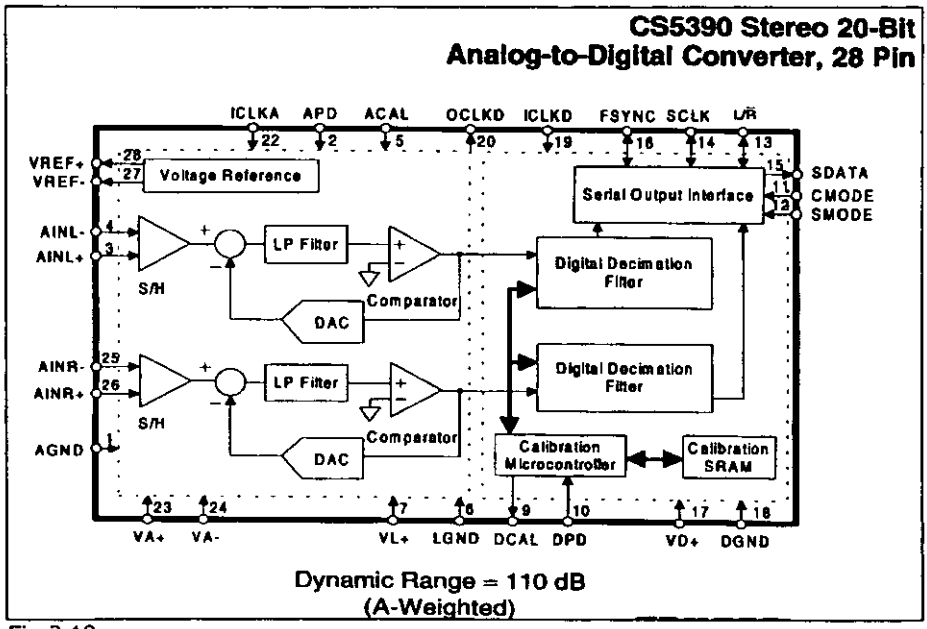


Fig.2.19.

CAPITULO III

III MEDICION, PRUEBA Y RESOLUCION DE PROBLEMAS DE DISEÑO AL UTILIZAR CAD´S

Existen diferentes técnicas utilizadas para medir y cuantificar el desarrollo de los convertidores analógicos digitales y el *front-end* analógico. Dos pruebas básicas incluyen el uso de los histogramas y de la Transformada Rápida de Fourier. *(Fast Fourier Transform FFT).* Cada prueba provee una única perspectiva de desarrollos de circuitos. Cuadro.3.1.

La prueba de selección esta basada sobre el parámetro específico medido. Por ejemplo, los histogramas son usados para medir la Corriente Directa con exactitud o las características del desarrollo estático tales como el *offset.* Las *FFT´s* son utilizadas para medir las características del desarrollo dinámico como la linealidad. Ciertas características tales como la linealidad y el ruido pueden ser medidas con cualquier técnica.

La combinación de los histogramas y la *FFT* pueden ser utilizadas para prueba y resolución de problemas. un ejemplo son, los histogramas son utilizados para cuantificar el bajo nivel de ruido y la *FFT* es utilizada para identificar y cuantificar el ruido asíncrono tal como la alimentación a través del reloj. Ambas

pruebas pueden utilizar el mismo *set* de datos para determinar si el ruido es síncrono o asíncrono.

Médición y Cuantificación del desarrollo
Los Histogramas se utilizan para medir y cuantificar el desarrollo estático
Las FFT´s se utilizan para medir y cuantificar el desarrollo dinámico
La combinación de los histogramas y las FFT´s se utilizan para pruebas y resolución de implementacion de los Convertidores.

Cuadro.3.1.

Los convertidores analógicos digitales son utilizados para medir el nivel de magnitud de señales estadísticas. Las aplicaciones incluyen la medida del peso, presión o temperatura. Estas aplicaciones envuelven señales de bajo nivel las cuales requieren alta resolución y precisión.

III.1.0. Histogramas.

Los histogramas son utilizados para medir el desarrollo estático. Un histograma es una tabulación de ocurrencia de frecuencias por el significado de rectángulos cuya área es proporcional correspondiente a las frecuencias. Los datos de los histogramas son utilizados para cuantificar el error y el ruido asociado con el proceso de conversión.

Una entrada estática es aplicada al convertidor analógico digital - CAD y múltiples muestras de salida son colectadas. El histograma para un convertidor digital analógico CDA sin ruido consiste solo de un cubo, o una cuenta, por que la salida del código será solo de un valor. Cada muestra sumada de un CAD sin ruido siempre será del mismo valor. Si el proceso de conversión tiene ruido consigo, los valores del *set* de la muestra variarán de acuerdo con la cantidad de ruido. El histograma de un CAD con ruido exhibirá una distribución de más de un código. El histograma con ruido de la siguiente figura sugiere que la salida de una salida estática puede ser una de siete códigos posibles. Fig.3.1.

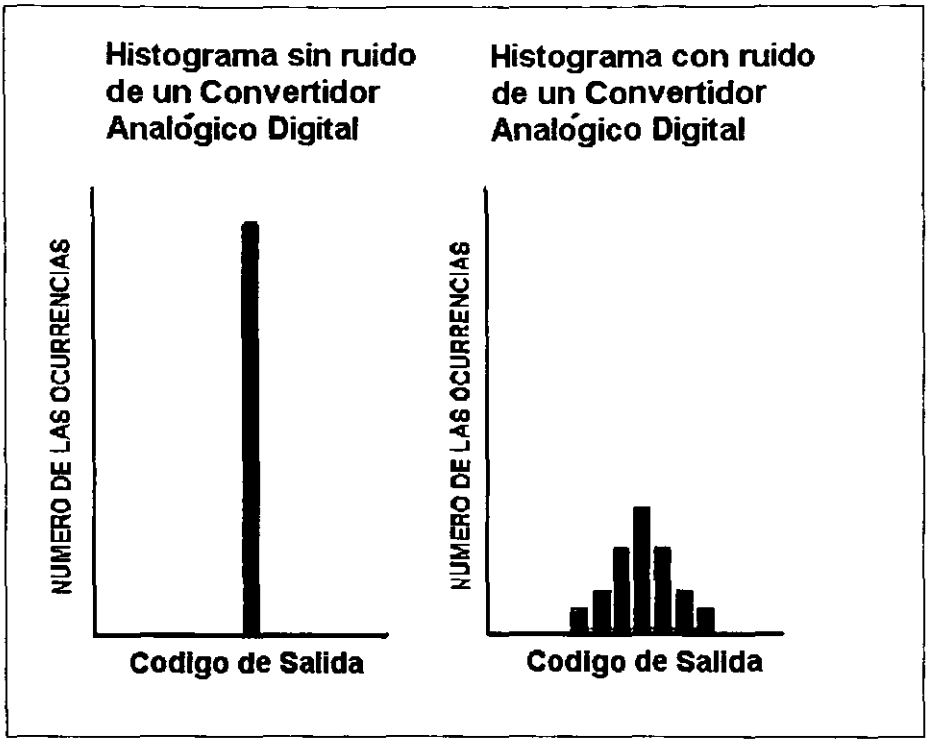


Fig.3.1.

Un histograma es una tabulación de la ocurrencia de la frecuencia que se representa por áreas de rectángulos que son proporcionales a sus frecuencias correspondientes.

III.1.1. Estimación de la Función Gaussiana de Densidad de Probabilidad

La salida de un convertidor analógico digital varía para una constante de entrada debida al ruido. El ruido es definido por una función de densidad de probabilidad (FDP) - *Probability Density Function (PDF),* la cual representa la probabilidad de los eventos discretos. La figura del *PDF* describe la certidumbre de las salidas de los CAD y las características del ruido.

Las técnicas estáticas son utilizadas para alcanzar un desarrollo de medidas, para valorar un desarrollo de medidas, y también para compensar el ruido. Un *set* de muestras de conversión de datos son recolectadas de un CAD y analizada. Los resultados son utilizados para crear a un *PDF* los cuales describen la verdadera operación de los CAD.

Los resultados del ruido eléctrico de efectos aleatorios forman una distribución normal o Gaussiana, la cual es una figura de campana llamada distribución normal. La PDF Gaussiana es continua y completamente determinada por la media (μ) y la desviación ($\sigma 2$). La *PDF* Gaussiana esta definida por la siguiente ecuación:

$$p(x) = \frac{n}{\sigma\sqrt{2\pi}} e^{-(X-\mu)^2/2\sigma^2}$$

.n=1 para el actual *PDF*.

Para otros valores de una escala **n** el *PDF* ajusta un set de muestras. Los datos presentados en el histograma pueden ser utilizados para estimar el *PDF*. El diagrama siguiente muestra el histograma de un CAD con ruido y con su *PDF* estimado. La media y la desviación fueron estimadas de un set de muestras de datos utilizando las siguientes ecuaciones:

$$\sigma^2 \approx \frac{\sum_{i=1}^{n}(X_i - \mu)^2}{n-1} \qquad \mu \approx \frac{1}{n}\sum_{i=1}^{n} X_i$$

donde **Xi** es una muestra de salida digital de un CAD y **n** es el número de muestras.

Notar que el *PDF* y el histograma en el ejemplo están muy identificados y el *PDF* estimados parecen ser un buen modelo del sistema actual.

Estimando la función gaussiana de densidad de probabilidad, los resultados eléctricos del ruido de los efectos aleatorios forman una distribución gaussiana.

La Función Gaussiana de Densidad de Probabilidad (p(x)) esta completamente determinada con la especificación de una media μ y una desviación σ. Fig.3.2.

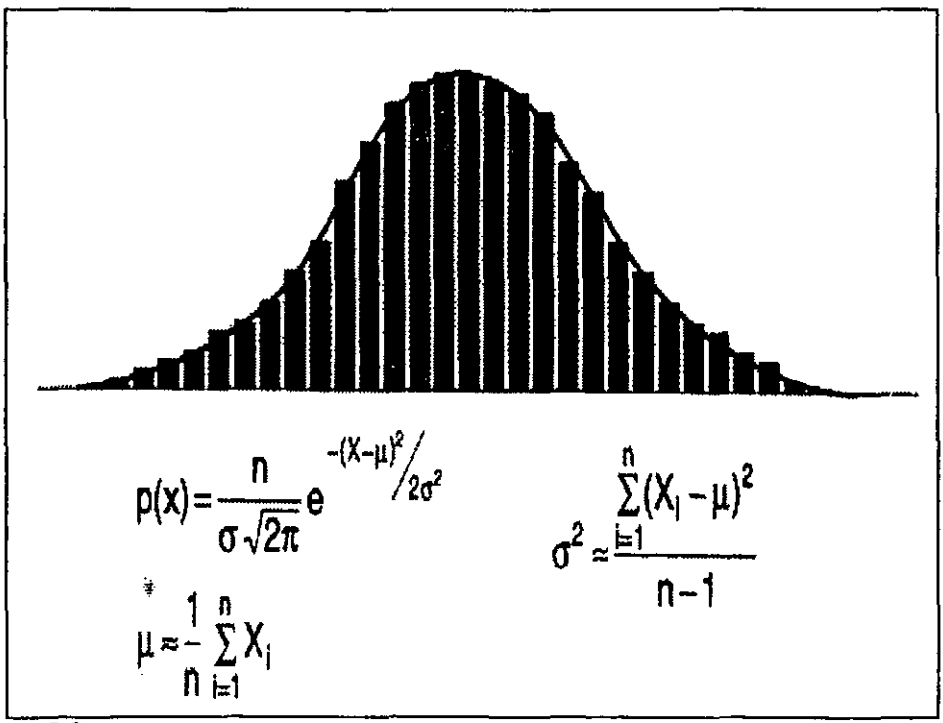


Fig.3.2.

III.1.2. Histograma Estadístico

De los parámetros de la función de densidad de probabilidad *PDF*, el desarrollo del convertidor CAD puede ser cuantificado. La media es el valor esperado o el valor promedio. Es utilizado para medir los errores de *offset*. La desviación describe la variabilidad de la distribución cerca de la media. Es usada como una medida de incertidumbre o ruido.

La raíz cuadrada de la desviación es llamada la desviación estándar (σ), y esta es una medida de la efectividad o de la raíz media cuadrada *(rms)* del ruido. El ruido pico a pico puede ser determinada del valor *rms* del ruido.

Cuando la entrada del CAD es colocada a cero, la salida digital esperada es cero. El *set* medio de la muestra (μ) es el error *offset* cuando la entrada es puesta a cero.

En el siguiente histograma estadístico el *PDF* calculado cuantifica el desarrollo del convertidor. Fig.3.3.

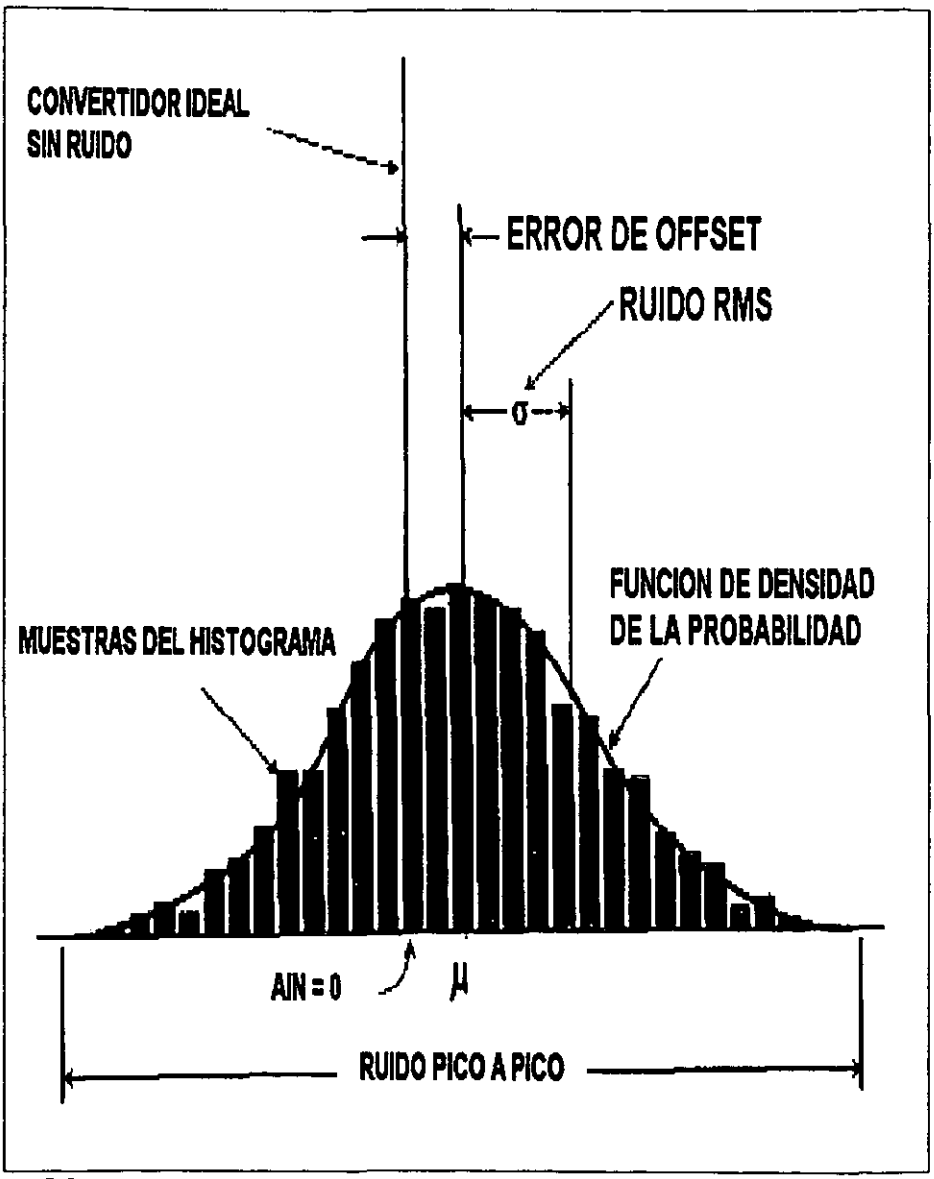

CONVERTIDOR IDEAL
SIN RUIDO
ERROR DE OFFSET
RUIDO RMS
σ
FUNCION DE DENSIDAD
DE LA PROBABILIDAD
MUESTRAS DEL HISTOGRAMA
AIN = 0
μ
RUIDO PICO A PICO

Fig.3.3.

III.1.3. Histogramas Estadístico con un *AIN = GND*

Los valores estadísticos del histograma son transferidos a los números desarrollados. Los niveles de *offset* y ruido son estimados del *set* de muestras de datos. Por ejemplo, un histograma es hecho con los datos de un CAD del cual la entrada esta a tierra. La salida esperada es cero. El error de *offset* es estimado por media del *set* de muestras. Si el error de *offset* es cero, la media del *set* de muestras será cero.

La desviación del *set* de muestras es usado para estimar el nivel de ruido. El ruido (*rms*) es estimado por la desviación estándar del *set* de muestras y el ruido de pico a pico es estimado a 6.6 veces del ruido *rms.*

En una distribución normal, 6.6 de la desviación estándar contiene 99.90 % de todas las ocurrencias. Otros números en lugar de 6.6 pueden ser utilizados para estimar el ruido pico a pico. Por ejemplo 5.0 de la desviación estándar contienen 98.8% de todas las ocurrencias. La aplicación dictará el intervalo oficial. Cuadro.3.2.

Histogramas Estadísticos con AIN = GND	
1.- Error de OFFSET	**μ = Error de Offset**
2.- Ruido RMS	**σ = Ruido RMS**
3.- Ruido Pico a Pico	**6.6 x σ =Ruido Pico a Pico**
Estadística del Histograma puede ser traducido en números necesarios para la medición del Desarrollo	

Cuadro.3.2.

Estos cálculos pueden ser representados con diferentes niveles de entrada en diferentes puntos de la función de transferencia de los DAC´s.

III.1.4. Ejemplo de un Histograma Estadístico Analizando el CS5508

Un ejemplo utilizando histograma estadístico para medir el desarrollo de un convertidor Crystal CS5508. En este caso, el convertidor CS5508 de 20 bits es

usado de modo bipolar con un voltaje de referencia de 2.5 volts. La amplitud del código es de 4.768 uV debajo de estas condiciones.

$$4.768uV = \frac{(2.5V- -2.5V)}{2^{20}}$$

Las 8192 muestras son colectadas de los CAD´s con su entrada a tierra. El error *offset* es estimado utilizando la media del *set* de muestreo como es para -2.99 cuentas o –14.3 uV. El ruido *rms* es estimado utilizando el *set* de muestras de la desviación estándar como es para 2.40 cuentas o 11.4 uV. Utilizando set de muestras de la desviación estándar, el ruido pico a pico es estimado a 15.8 cuentas o 75..5 uV. Fig.3.4.

Esta prueba puede ser repetida con un *AIN* cerca de la escala completa. En este caso, la media puede ser utilizada para medir la ganancia en el error. La desviación estándar del ruido *rms* permanecerá constante. Aunque, puede existir una diferencia debida al error de muestreo. Un significante incremento de la desviación estándar indica ruido en la fuente de la señal o referencia de voltaje al convertidor CAD.

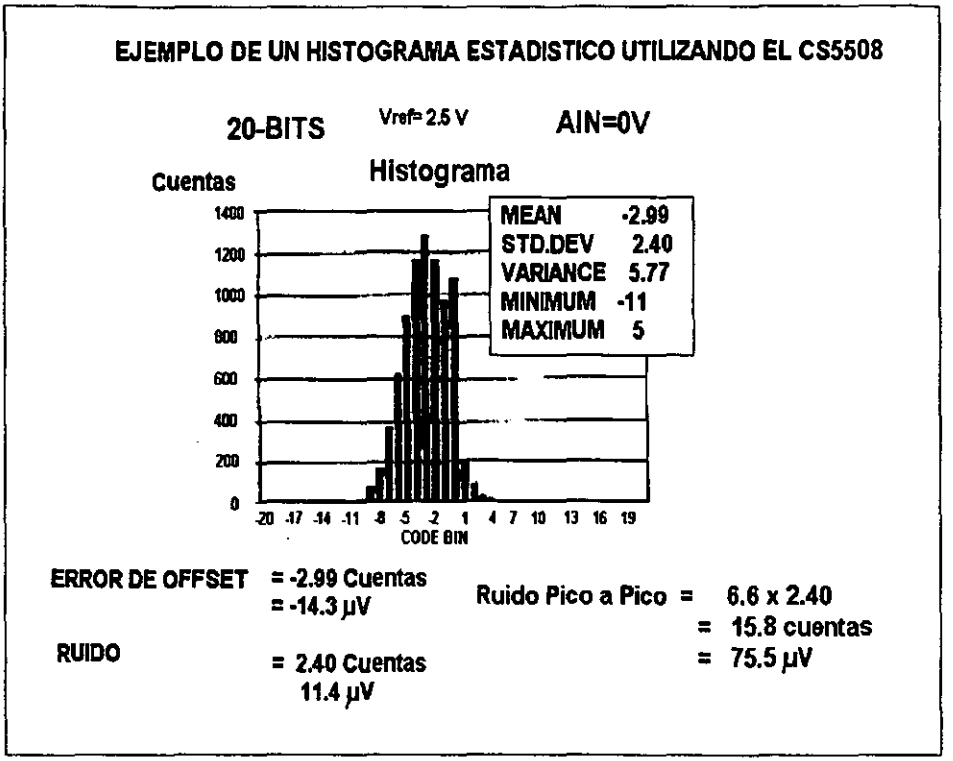


Fig.3.4.

III.1.5. Estimación del PDF.

Previamente, el error del *offset* fue estimado a –14.7 uV y el ruido a 11.4 uV *rms*. Al ser los números calculados de los datos muestreados a un grado de

incertidumbre es asociado con el desarrollo estimado. La exactitud de la estimación es dependiente del número de muestras coleccionadas.

El número de muestras que necesitan ser coleccionadas dependen de algunos factores incluyendo el intervalo confidencial, grado de exactitud y la distribución de la variables.

Intervalos confiables son el grado de certeza donde la estimación esta dentro de un rango específico. Más altos grados de exactitud o de distribuciones con una gran variación requerirán más muestras para estimar. Cuadro.3.3.

Es mejor usar más muestras para calcular estimaciones para el sistema actual.

Estimación de la Función de Densidad de Probabilidad (PDF)
¿Cúantas muestras son necesarias para estimar el PDF?
Depende de la Confiabilidad del Intervalo
Depende del grado de Exactitud
Depende de la distribución de la variable
En resumen mas muestras es mejor.

Cuadro.3.3.

III.2.0. Promedio del Muestreo Múltiple

En este punto el promedio es un método utilizado para incrementar la resolución de un convertidor analógico digital CAD y disminuir la incertidumbre debida al ruido.

En el siguiente ejemplo, la entrada al CAD es equivalente a 1.45 cuentas. La salida del CAD será siempre 1, si no existe ruido presente. Sin embargo, el ruido está presente y las 12 muestras del CAD son 1,2,3,1,1,2,1,2,1,0,2,1. Si sólo una muestra del CAD es utilizada, la muestra puede ser cualquiera de las cuatro posibilidades de 0 –3. Por ejemplo, Si solo la primer muestra es usada, el resultado es 1.

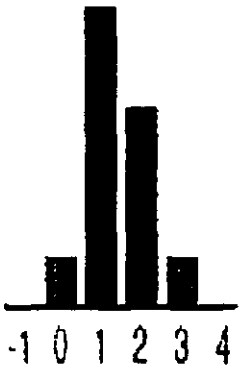
Las muestras múltiples pueden ser un promedio para incrementar la exactitud. Si en las primeras cuatro muestras el promedio es de 1.750. Promediando 9 muestras, da como resultado 1.556, y promediando 12 muestras da como resultado 1.417. Notar como utilizando más muestras se produce un resultado el cual se acerca más al valor actual 1.45 cuenta. Promediando se incrementa la precisión de la conversión. Fig.3.5.

PROMEDIO MULTIPLE DE MUESTRAS

Ejemplo:

Configura la entrada analógica a 1.45 cuentas
Salida Digital del convertidor analogico digital

1,2,3,1,1,2,1,2,1,0,2,1



PRIMER MUESTRA = 1

PROMEDIANDO PRIMERAS 4 MUESTRAS = 1.750

PROMEDIANDO PRIMERAS 9 MUESTRAS = 1.556

PROMEDIO DE LAS 12 MUESTAS = 1.417

EL PROMEDIO ES UN METODO UTILIZADO PARA AUMENTAR LA RESOLUCION DE UN CAD Y REDUCIR LA INCERTIDUMBRE DEBIDA AL RUIDO

Fig.3.5.

III.2.1. Exactitud del Resultado Promediado

Como discusión, en el histograma donde se analiza el ruido, la exactitud del promedio y la incertidumbre asociada con los datos muestreados, depende del *set* de muestras de la desviación.

La incertidumbre puede ser calculada. Del ejemplo anterior, la muestra tiene un set de 12 y tiene la siguiente estadística:

Media de muestreo =	1.417
Desviación del muestreo =	0.629
Desviación estándar del muestreo =	0.793

Las características estadísticas de una distribución normal establece que un 95% de todas las ocurrencias será dentro de +-1.96 de la desviación estándar. Utilizando una muestra, la desviación es +-(1.96*0.793) o +-1.554 cuenta. Si la salida del CAD es 1, hay un 95% de certeza que el valor actual y esta dentro de –0.554 y 2.554.

El promedio de las muestras múltiples reduce la desviación estándar del *set* de datos promedios por 1 sobre la raíz cuadrada del número de muestras. Utilizando el ejemplo del *set* de muestreo, promediando 12 muestras da como

resultado en la cuenta 1.417 con una incertidumbre de +-0.449 cuentas. La incertidumbre fue reducida por $1 / \sqrt{12}$. Promediando se mejora la exactitud de la conversión.

El promedio puede mejorar la resolución de un CAD. Si se promedian suficientes muestras de un CAD el resultado puede ser exacto igual que una resolución de 17 o 18 bits.

Ahora que la Cuantización produce una incertidumbre de +- 0.5 cuentas. Si una muestra es utilizada (y no promediada), la incertidumbre es de 1.554 cuentas. La incertidumbre total será 3.108 cuentas. Si el CAD tiene una resolución de 16 bits, la resolución efectiva está entre 14 y 15 bits libres de ruido.

Si 12 muestras son promediadas juntas la incertidumbre es reducida a +- 0.449 cuentas o a un total de incertidumbre de 0.898 cuentas. El total de incertidumbre de la conversión del ruido es 1.0 cuenta debido al error de cuantización. Promediando 12 muestras de un CAD de 16 bits se comporta ligeramente mejor que una resolución 16 bits libre de ruido.

Si 39 muestras son promediadas juntas, la incertidumbre es reducida a 0.249 cuentas. La incertidumbre total es menor a 0.5 cuentas. El resultado de promediar 39 muestras es mejor que 17 bits de resolución libres de ruido. Fig.3.6.

QUE TAN EXACTO PUEDE SER EL RESULTADO DE UN PROMEDIO?

LA INCERTIDUMBRE DEPENDE DE LA VARIACION DEL SET DE MUESTRAS

EJEMPLO :

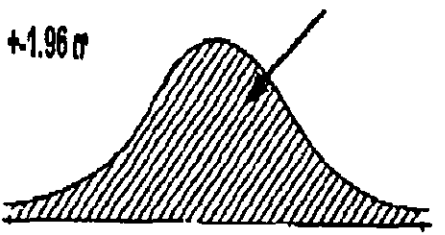
RUIDO RMS = σ

95% DEL AREA TOTAL

CONFIANZA DEL INTERVALO=95%

95% DE LAS OCURRENCIAS ENTRE +-1.96 σ

$\mu = 1.417$ $\sigma 2 = 0.629$ $\sigma = 0.793$



-1.96 σ -1.96 σ

PROMEDIO INCERTIDUMBRE

1	MUESTRA 1	± 1.554	DONDE	1.554 = (1.96)(0.793)
4	MUESTRAS 1.750	± 0.777	DONDE	0.777 = 1.554 / $\sqrt{4}$
9	MUESTRAS 1.556	± 0.518	DONDE	0.518 = 1.554 / $\sqrt{9}$
12	MUESTRAS 1.417	± 0.449	DONDE	0.449 = 1.554 / $\sqrt{12}$

PROMEDIAR REDUCE LA INCERTIDUMBRE EN $1/\sqrt{\text{Numeros de los Promedios}}$

Fig.3.6.

III.3.0. Cómo afecta el DNL en los Histogramas

La diferencia no lineal pobre (*DNL*) afecta negativamente la distribución del histograma. Este punto es muy importante. En los cálculos del histograma, se asume que la incertidumbre es un resultado de ruido aleatorio y el set de muestras posee una distribución normal o de campana de Gauss. Fig.3.7.

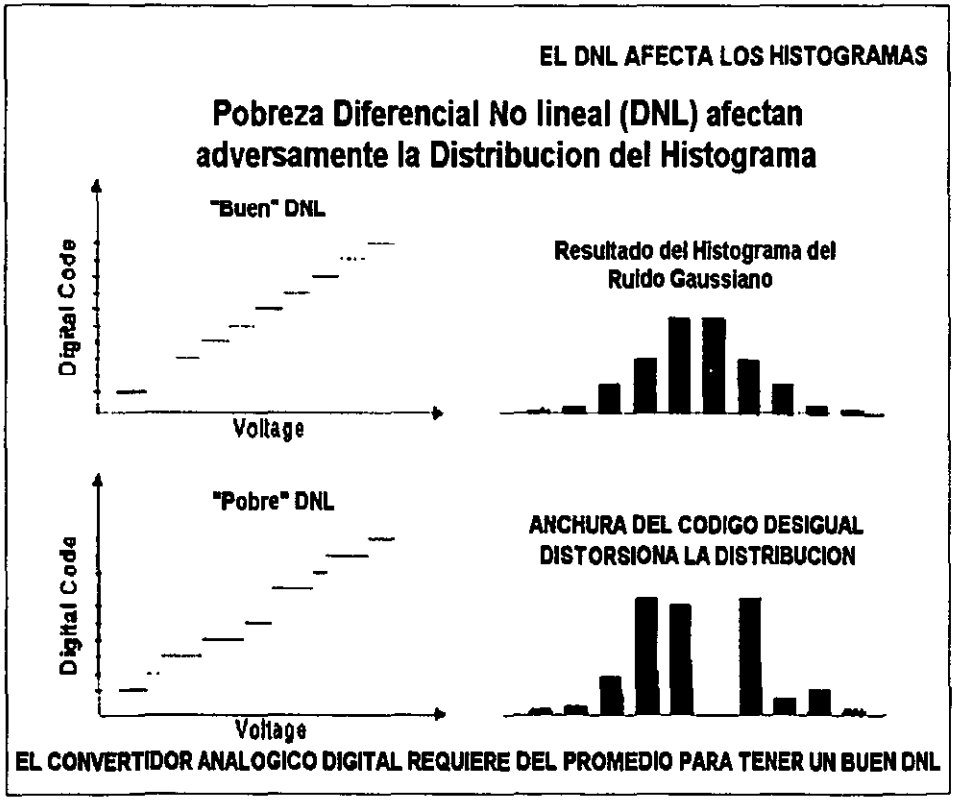


Fig.3.7.

Una pobre DNL distorsiona la distribución de ruido aleatoreamente. Códigos con mayor amplitud tienen una alta probabilidad de ocurrencia que los códigos angostos. El error DNL afecta la distribución del histograma.

Si la distribución esta severamente distorsionada, la estadística puede ser no confiable. Es importante escoger un CAD de buenas características si la aplicación requiere promediarse.

III.3.1. Cómo se utilizan los histogramas para medir el DNL

Los histogramas pueden ser utilizados para medir la Diferencia no lineal (DNL) de un convertidor analógico digital sobre un pequeño tramo. Una señal triángulo es aplicada a la entrada del CAD y un set de muestras de datos es colectada. Si el DAC tiene un DNL perfecto, cada código en la función de transferencia tiene la misma probabilidad de ocurrencia. Una señal triángulo perfecta producirá un histograma perfectamente plano.

Para un CAD con un DNL perfecto, la altura de cada bin será:

Peso del Cuadro *(Bin Height)* = Número de muestras / *Número de cuadros (bins).*

Si hay un DAC con un DNL perfecto, la altura del ***cuadro-bin*** del histograma es proporcional al tamaño de la amplitud del código. Un bin con altura cero indica cero error perdido.

Para esta prueba se recomienda que por lo menos 200 muestras por bin sean utilizadas. Esto significa que un histograma de 8192 muestras podría manejar un trozo de 40 cuentas. Debido a esto es bueno probar o verificar el DNL de pequeñas secciones de la función de transferencia. Fig.3.8.

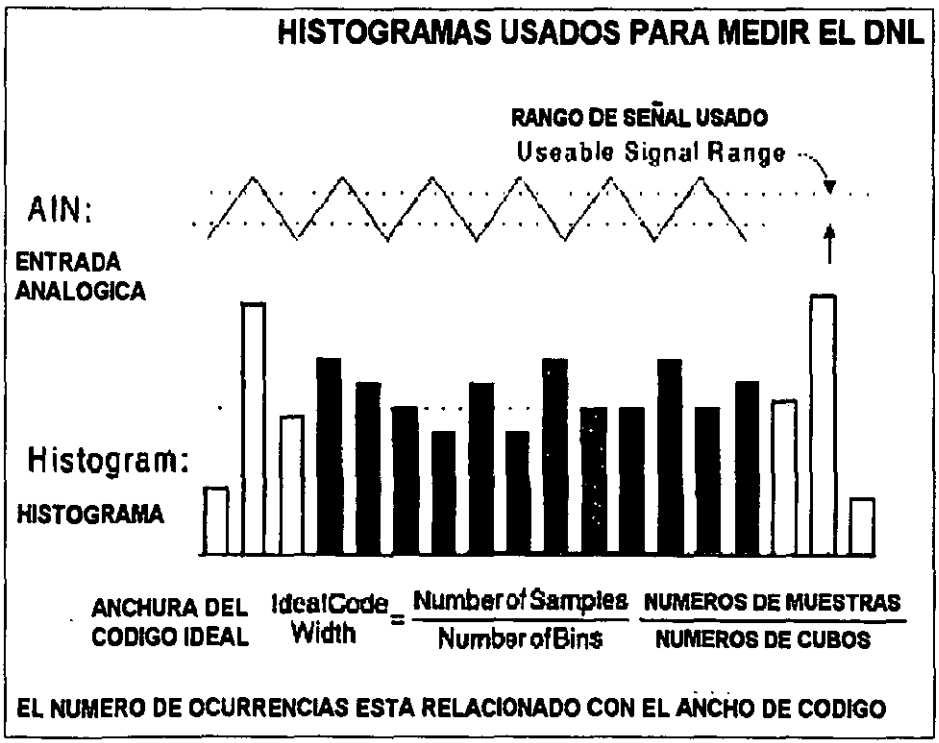


Fig.3.8.

Regularmente el equipo de prueba es incapaz de producir una señal triángulo perfecta para prueba. En este caso, la porción central del histograma deberá ser utilizada para estadística. Las dos finales del histograma pueden ser ditorsionadas debido al swing de la amplitud de la señal de la fuente y el tiempo de conversión.

III.4.0. Medición del Desarrollo Dinámico utilizando la Transformada Rápida de Fourier (*FFT*)

El crecimiento rápido de las aplicaciones para el proceso digital de señales tiene como resultado la necesidad de que los CAD´s sean probados utilizando las señales dinámicas.

En la medición del desarrollo dinámico de un CAD, el principal objetivo es el investigar como se altera la función de conversión del espectro de la señal trasmitida a través del sistema. El análisis de espectros se utiliza para encontrar el contenido de frecuencia de las señales. Aquí el dominio de las señales en el tiempo es transformado al dominio de la frecuencia. Las porciones de magnitud de los elementos de frecuencia son utilizadas para determinar la distribución con respecto

a la frecuencia, construyendo una gráfica espectral de potencias. La potencia del espectro es utilizada para medir la distorsión del sistema, el rango dinámico, la atenuación, y el ruido inducido. Fig.3.9.

El análisis en el dominio de la frecuencia empieza con la translación de la señal del dominio del tiempo hacia adentro del dominio de la frecuencia. El dominio de la frecuencia consta de elementos de magnitud y componentes de fase.

Aplicaciones que utilizan la información en el dominio de la frecuencia incluyen el radar, el sonar, el reconocimiento por voz, análisis vibracional, inspección por video, imágenes médicas, e instrumentos de control. Estas aplicaciones específicas enfatizan, analizando cierta información de los componentes de frecuencia. Por ejemplo, el rango de búsqueda por radar utiliza la información de fase de la señal transmitida y sus reflexiones recibidas. La diferencia de fase entre las señales de transmisión y recepción contienen un rango de información. Para cuantificar el desarrollo del CAD, solo la magnitud se utiliza.

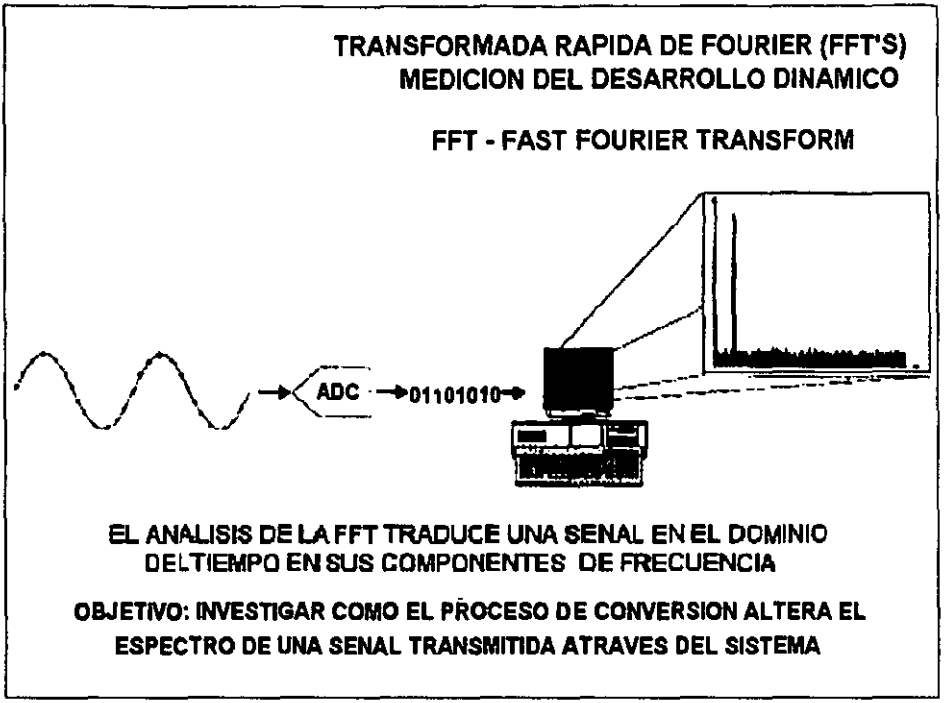


Fig.3.9.

III.4.1. Requerimientos de la Transformada Rápida de Fourier (*FFT*)

El proceso de la transformada rápida de Fourier *(FFT)* empieza en el CAD, el cual convierte una señal analógica a un representación digital discreta (muestreo). La amplitud de la señal analógica es convertida a una representación n-bit binaria

llamada palabra digital. El CAD saca estas palabras digitales a un rango seleccionado por una frecuencia de muestreo (*Fs*). Esta frecuencia de muestreo es primeramente determinada o por el *troughput* del CAD.

Una muestra del *set* de puntos de datos es colectada por el algoritmo *FFT*. El algoritmo *FFT* opera sobre un *set* de muestras N, para las cuales N deberá ser una potencia de 2, por ejemplo, N= 2^8 =256 o N=2^9=512. El tamaño del *set* de muestras es determinado por la capacidad del proceso (tamaño de la memoria y procesamiento *troughtput*), por las señales de entrada dinámicas, y por el grado de resolución de la frecuencia.

El algoritmo *FFT* asume que la señal de entrada es periódica sobre un número de puntos en la *FFT*. Esto significa que copias múltiples de la señal pueden ser colocadas al final sin ninguna discontinuidad en el punto donde se unen. El diagrama de requerimiento posterior de la figura 3.10; muestra una señal que es periódica para un número de puntos en la FFT. El primer *set* de muestras ilustra la periodicidad, cuando 3 *sets* de datos son combinados, no hay discontinuidad (suave transición entre el *set* de muestras).

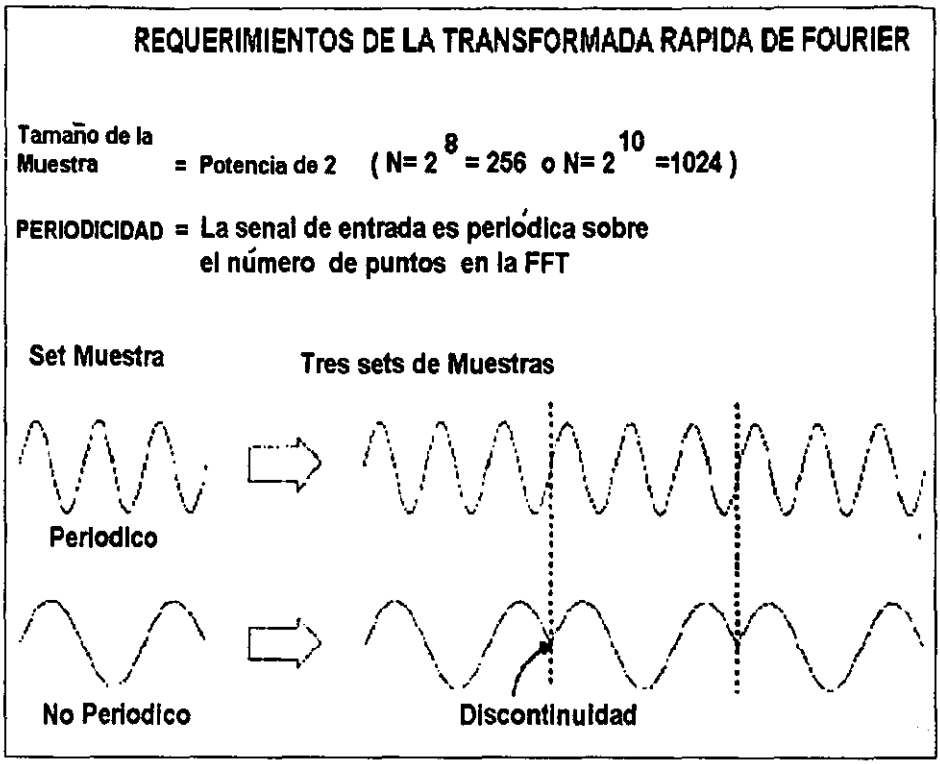


Fig.3.10.

El segundo *set* de muestras ilustra una señal que no es periódica para un número de puntos de muestra y resalta las discontinuidades en algunos puntos donde las copias se juntan. Una FFT de un dato no periódico será el resultado en elementos de alta frecuencia que no existen en la señal actual.

III.5.0. Muestreo de Datos con *Windows*

La periodicidad del algoritmo de la *FFT* requiere que el *set* de muestras contenga un número entero de ciclos de señal y no es práctico para la mayoría de las aplicaciones. Esto en efecto hacen que *Fs* y N sean función de la frecuencia de entrada. La periodicidad es más complicada cuando la señal está compuesta de múltiples elementos de frecuencia o no es solamente una señal seno. Fig.3.11.

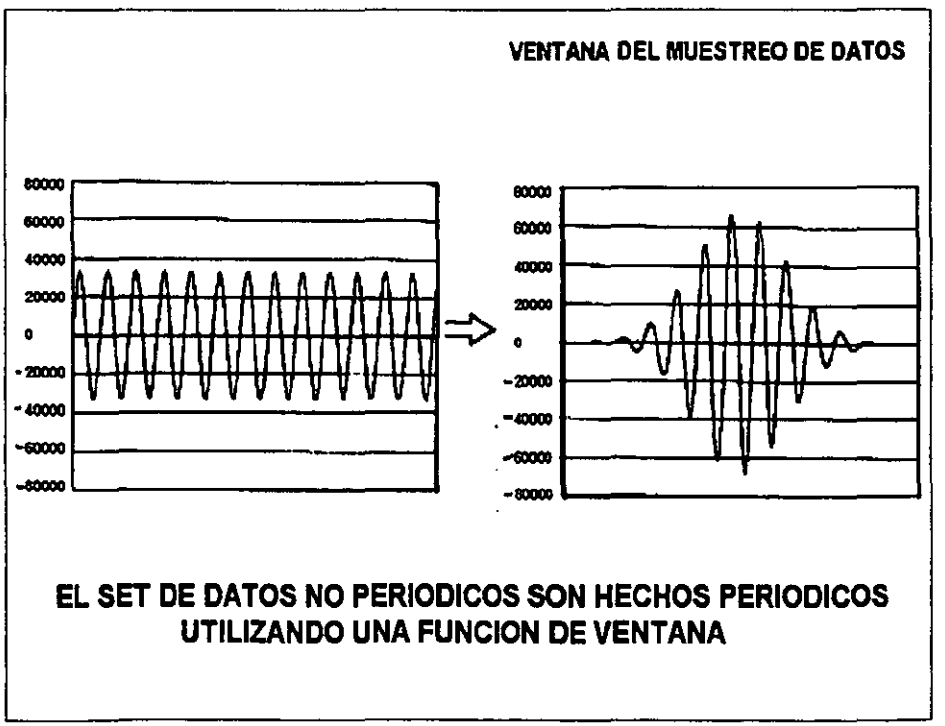


Fig.3.11.

Las ventanas son utilizadas para reducir la fuga espectral que resulta de utilizar *FFT* sobre señales no dinámicas o no periódicas. *Windowing* multiplica la señal en el dominio del tiempo por una función que atenúa la amplitud al finalizar. Debido a esto se tienen las discontinuidades. Fig.3.12.

El efecto de los datos de *windowing* es que la energía senoidal es desplegada de un *bin* a muchos otros *bins* con el lóbulo principal y con los lóbulos de frontera.

El objetivo en la selección *windows* para medidas dinámicas es el mantener la energía fundamental en el lóbulo principal y tener una fuga insignificante en los lóbulos de adyacentes.

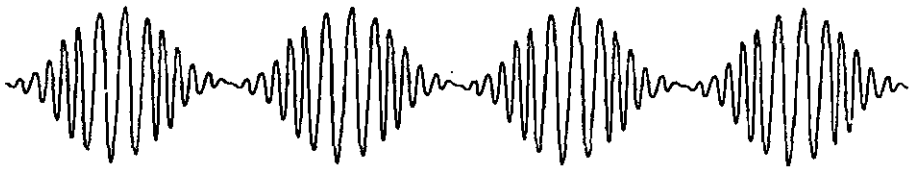


Fig.3.12.

3.5.1. Efectos en los *Windows*

Con las funciones *windows*, el primer paso a tratar es la mancha de la señal de entrada entre varios *bins* y segundo tratar con los lóbulos adyacentes. La resolución de la frecuencia y la fuga espectral son influenciados por la selección de una función *windows*.

Cuando se utiliza una ventana la señal de entrada es desplegada a lo largo de todas las frecuencias *bins*, las cuales representan el lóbulo principal debido a la frecuencia exacta estando muy probablemente definida. Una señal de lóbulo principal tiene mayor amplitud que una mancha de señales que tiene un lóbulo principal más angosto.

El grado de fuga de la señal para encasillar fuera del lóbulo principal es dependiente de la magnitud de los lóbulos adyacentes. Sí el nivel de lóbulo adyacente es mayor que el ruido de piso, la señal mancha a frecuencias fuera del lóbulo principal. La magnitud del lóbulo adyacente es generalmente inversamente proporcional a la amplitud del lóbulo principal. Debido a esto la resolución como frecuencias más bajas es tratada afuera para mejor atenuación de los lóbulos adyacentes. Fig.3.13.

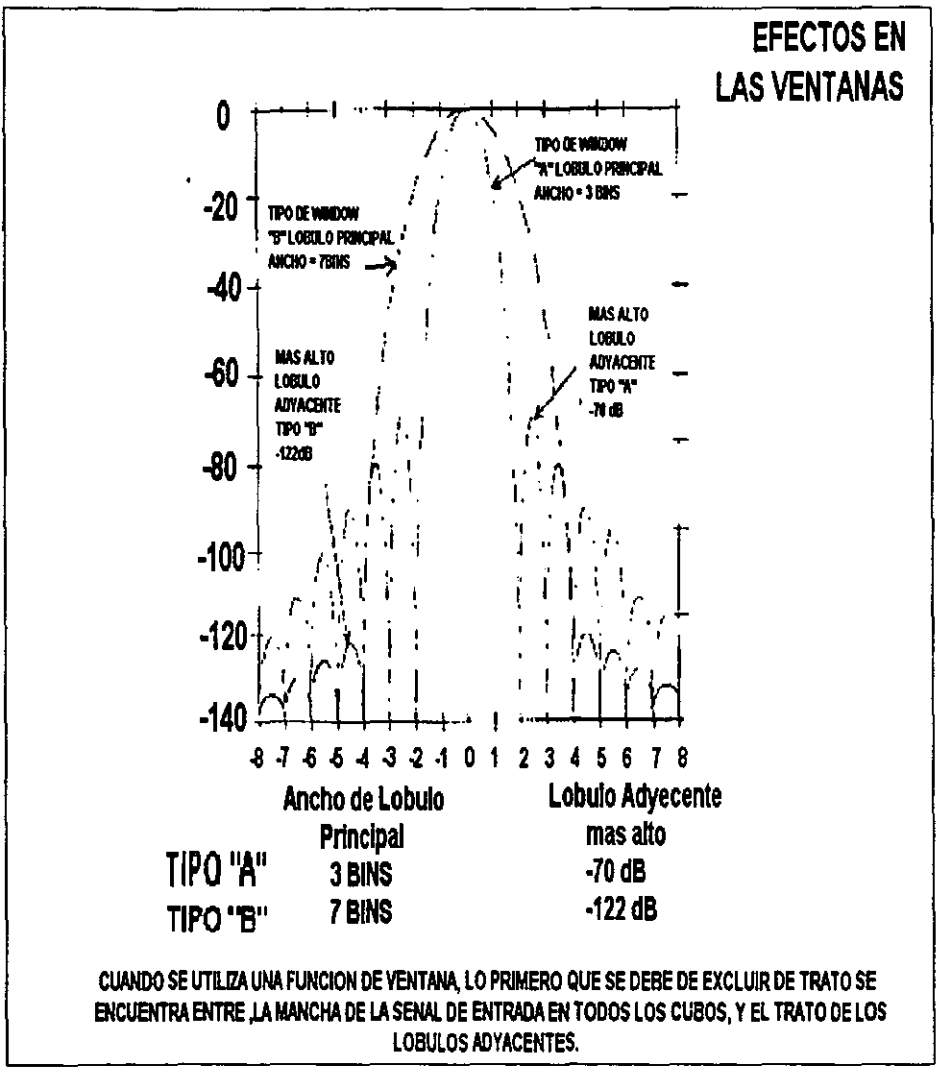

EFECTOS EN
LAS VENTANAS
0
-20
-40
-60
-80
-100
-120
-140
TIPO DE WINDOW
"A" LOBULO PRINCIPAL
ANCHO = 3 BINS
TIPO DE WINDOW
"B" LOBULO PRINCIPAL
ANCHO = 7BINS
MAS ALTO
LOBULO
ADYACENTE
TIPO "A"
-70 dB
MAS ALTO
LOBULO
ADYACENTE
TIPO "B"
-122dB
-8 -7 -6 -5 -4 -3 -2 -1 0 1 2 3 4 5 6 7 8
Ancho de Lobulo
Principal
Lobulo Adyecente
mas alto
TIPO "A"
3 BINS
-70 dB
TIPO "B"
7 BINS
-122 dB
CUANDO SE UTILIZA UNA FUNCION DE VENTANA, LO PRIMERO QUE SE DEBE DE EXCLUIR DE TRATO SE ENCUENTRA ENTRE ,LA MANCHA DE LA SENAL DE ENTRADA EN TODOS LOS CUBOS, Y EL TRATO DE LOS LOBULOS ADYACENTES.

Fig.3.13.

En el diagrama la amplitud de los lóbulos principales para el tipo de *windows* "A" tiene tres bins y mayor número de lóbulos adyacentes siendo de -70 dB. El tipo "B" tiene un *window* más alto en los lóbulos adyacentes siendo esta de –122dB, la cual es 55 dB mejor que el tipo "A".

La mejoría en la atenuación del lóbulo adyacente resulta en un lóbulo principal más amplio que siete *bins*, esto es un incremento de cuatro *bins*.

El tiempo de subida/bajada del *window* y de su amplitud caracterizan el desarrollo de *window*. Subida/bajada es el rango de la atenuación del punto final. Un tiempo de subida/bajada rápido produce niveles más altos de los lóbulos adyacentes. La amplitud del *window* es el número de muestras donde la ventana está en su máximo valor. La amplitud del lóbulo principal es inversamente proporcional a la amplitud del *window*. Una *window* grande crea una amplitud pequeña en el lóbulo principal. Cuando la amplitud de la ventana está alargada para disminuir la amplitud del lóbulo principal, el tiempo de incremento subida/bajada, da como resultado en lóbulos adyacentes más altos.

III.5.2. Cinco Tipos de Windows

Escoger el *windows* apropiado depende de la aplicación, el desarrollo del sistema, y de la información requerida.

Medir el desarrollo de un sistema con un CAD es una aplicación típica para el análisis con la transformada rápida de fourier *FFT*. El Objetivo es el de obtener la potencia espectral de un señal seno pura. Una señal senoidal pura reduce el número de variables y permite la medida de la ocurrencia del desarrollo del sistema.

El objetivo es el mantener toda la señal de `potencia en lóbulo principal. Esto requiere que una función *window* de lóbulos adyacentes sean más bajos que el ruido de piso teórico. Lo pequeño del lóbulo reduce la fuga espectral para encasillar afuera el lóbulo principal, haciendo la lectura por el procesador de la potencia de la señal más fácil. La amplitud del lóbulo principal y un nivel más alto de lóbulos adyacentes para las 5 *windows* son cinco tipos descritos posteriormente. Cuadro 3.4.

Cinco tipos de Windows		
Tipo de Ventana	**Amplitud del Lóbulo Principal (BINS)**	**Altitud de Lóbulo Adyacente Nivel (d B)**
Hanning	**5**	**-32**
Blackman	**7**	**-58**
Minimum 4-Term Blackman-Harris	**9**	**-92**
5-Term HODIE	**11**	**-125**
7-Term HODIE	**15**	**-175**

Cuadro.3.4.

III.6.0. Ruido de Piso Mínimo

En la medición del desarrollo dinámico, los cálculos son más fáciles cuando una de las *windows* de los lóbulos adyacentes se encuentran por debajo del ruido de piso mínimo. Los resultados en la figura del ruido de piso están dictados por el desarrollo CAD.

El ruido de piso mínimo se configura por la resolución y por el número de muestras utilizadas en el *FFT*. La siguiente figura ilustra el cálculo del *spot noise* para un CAD de 16-bits y 1024 muestras de FFT. Utilizando la fórmula siguiente, el *Signal-Noise Ratio* (SNR) es 98.08 dB. Fig.3.14.

SNR=6.02N-1.76 (dB)

El ruido de piso se determina por el error de cuantización y es 98.08 DB menor que una señal a escala completa. En una FFT con 1024 muestras, hay 512

bins entre cero y la frecuencia de Nyquist. El error de cuantización es desplegado aún entre los 512 *bins*.

***Spot Noise* =-98.08-10log(512)= -125dB**

Una fórmula para el *spot noise* como una función de la resolución del CAD y el tamaño de la muestras *FFT* es:

***Spot Noise* = -6.02N-1.76-10log (muestras/2)dB**

El ruido de piso para un Convertidor analógico digital CAD, el cual se utiliza para seleccionar el tipo de window apropiado, y es configurado por el error de cuantización y el número de muestras. Cuadro.3.5.

Ruido de Piso para Convertidores Analógicos Digitales ideales				
	Número de Bits Resolución	**Número de Muestras**	**Promedio del Spot Noise**	
	12	256	-95	
	14	256	-107	
	16	256	-119	
	12	1024	-101	
	14	1024	-113	
	16	1024	-125	
Promedio Spot Noise = Nivel de Ruido de Piso para un ADC ideal				
El nivel más alto del lóbulo adyacente deberá ser menor que el promedio del Spot Noise				
Ejemplo:	**16-bit ADC, 1024 muestras**	**5-Term 7-Term**	**HODIE HODIE**	

Cuadro.3.5.

Un convertidor de 16 bits y 1024 muestras colocan el ruido teórico de piso a −125 dB. Esto está basado en el error de cuantización y el número de muestras. Fig.3.14.

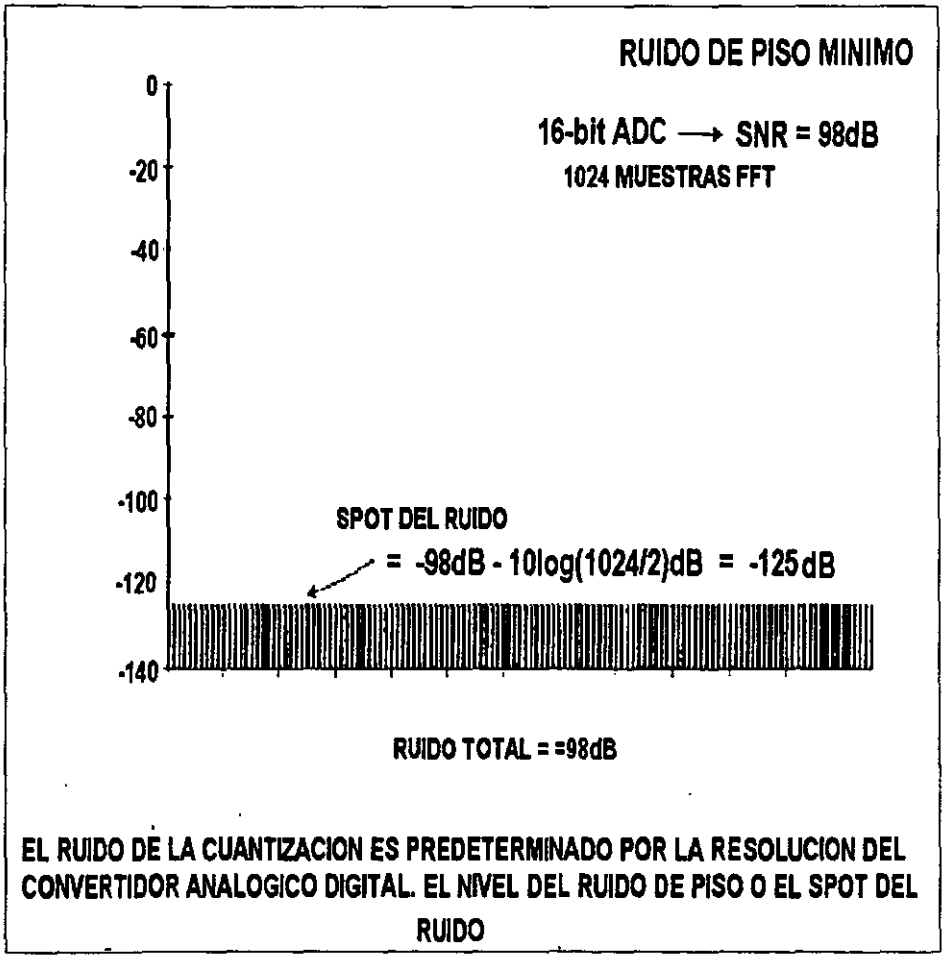


Fig.3.14.

El algoritmo *windowing* que utiliza debe atenuar los lóbulos adyacentes debajo de los –125 dB. Utilizando los resultados de la tabla "tipo *window*", el HODIE *5-term* o el HODIE *7-term window* es requerido para concentrar todas las señales de potencia en el lóbulo central. Las otras tres ventanas fugan señal de potencia a los lóbulos adyacentes.

III.7.0. Windows Apropiado

Este ejemplo envuelve la selección de una ventana para medir el desarrollo dinámico de un CAD cuando se desarrolla 1024 muestreos *FFT*. El *spot noise* es calculado para ser mayor que –173 dB. Debido a esto el lóbulo adyacente es más alto pero deberá ser menor que –173 dB. Fig.3.15.

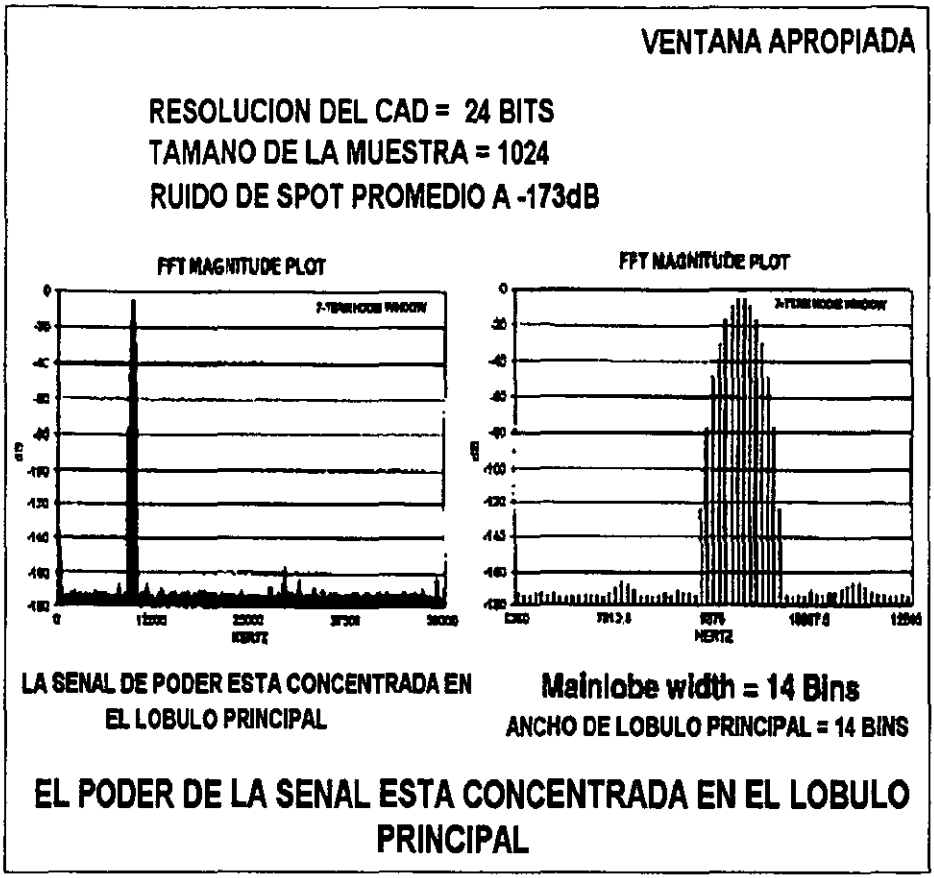


Fig.3.15.

El *window* HODIE *7-term* es configurable para esta aplicación. Notar que la figura FFT del ruido de piso es configurada por las características del CAD. Fig.3.16.

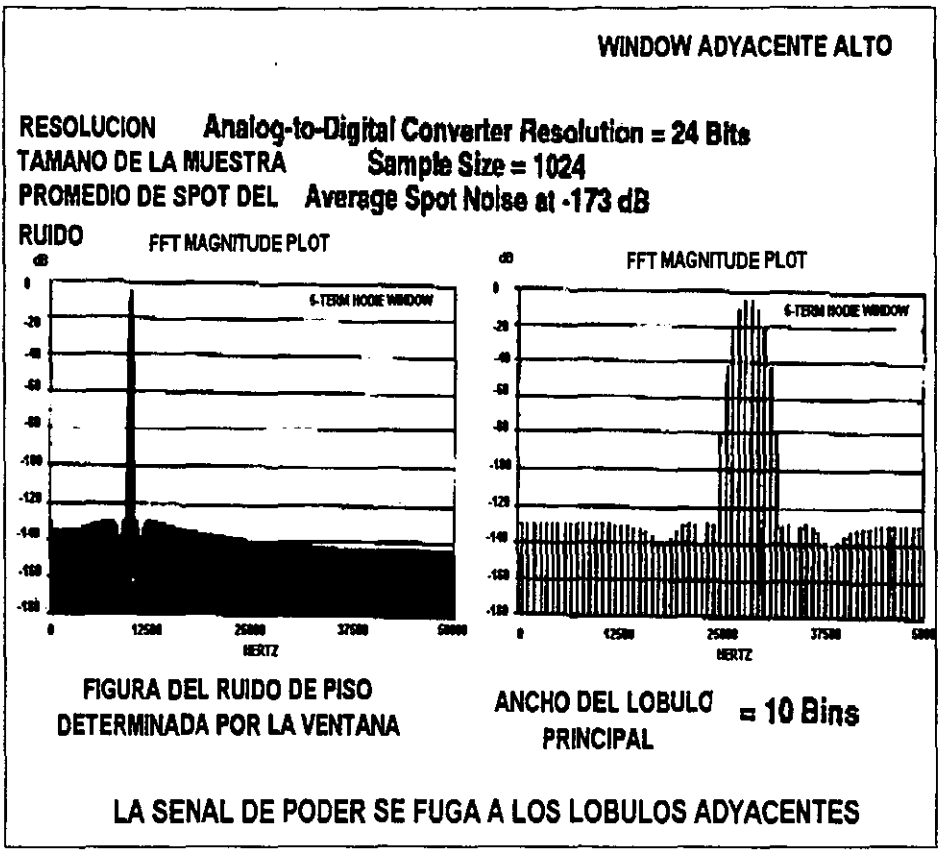


Fig.3.16.

El *window* HODIE *5-term* no será adecuada para esta aplicación debido a los altos lóbulos adyacentes. Notar como la energía de una señal a escala completa

ha manchado hacia fuera del lóbulo principal y se desplaza hacia los lóbulos adyacentes.

La figura del ruido de piso se determina por las características de la *window*.

III.8.0. Componentes de un *Plot* de Magnitudes de una Transformada Rápida de Fourier (*FFT*)

Medidas cuantitativas del desarrollo del sistema pueden ser hechas de los resultados presentados en un espectro de potencias.

El espectro de potencia ilustra la densidad espectral de la señal medida la cual puede mostrar la corrupción del seno puro por un sistema electrónico.

En un sistema perfecto, toda la energía de la señal se concentra en bins correspondiendo a la frecuencia de la onda seno de entrada.

El piso del ruido del espectro de potencia será plano y en un nivel correspondiente al número de los bits de resolución, utilizados por el CAD (error de cuantización). Los radios cuantifican el desarrollo de los CAD midiendo cómo una señal es corrompida con ruido y distorsión.

La siguiente gráfica muestra un espectro de potencia obtenido de 256 muestras (N=256) *FFT*. EL rango de muestreo es de 256 Hz (Fs = 256 Hz). Cada *bin* sobre el eje de las x representa una banda de 1 Hertz de potencia. Una escala completa de 55.1 Hz se aplica al CAD. Después de 256 muestras coleccionadas, el set de datos se multiplica por un *window* en la cual la amplitud del lóbulo principal es siete *bins* y los lóbulos adyacentes son más altos y menores que –120dB. Fig.3.17

El eje de las ordenadas representa la potencia concentrada en un rango de frecuencias representado por cada *bin*. La potencia del espectro puede ser dividida dentro de varias subsecciones en las cuales se encuentra la fundamental, las armónicas secundarias, la componente de CD, y el ruido de piso.

La potencia de cada una de estas secciones son calculadas. La potencia de los números son utilizados en el cálculo de la señal para hacer ruido en los radios utilizados para cuantificar el desarrollo.

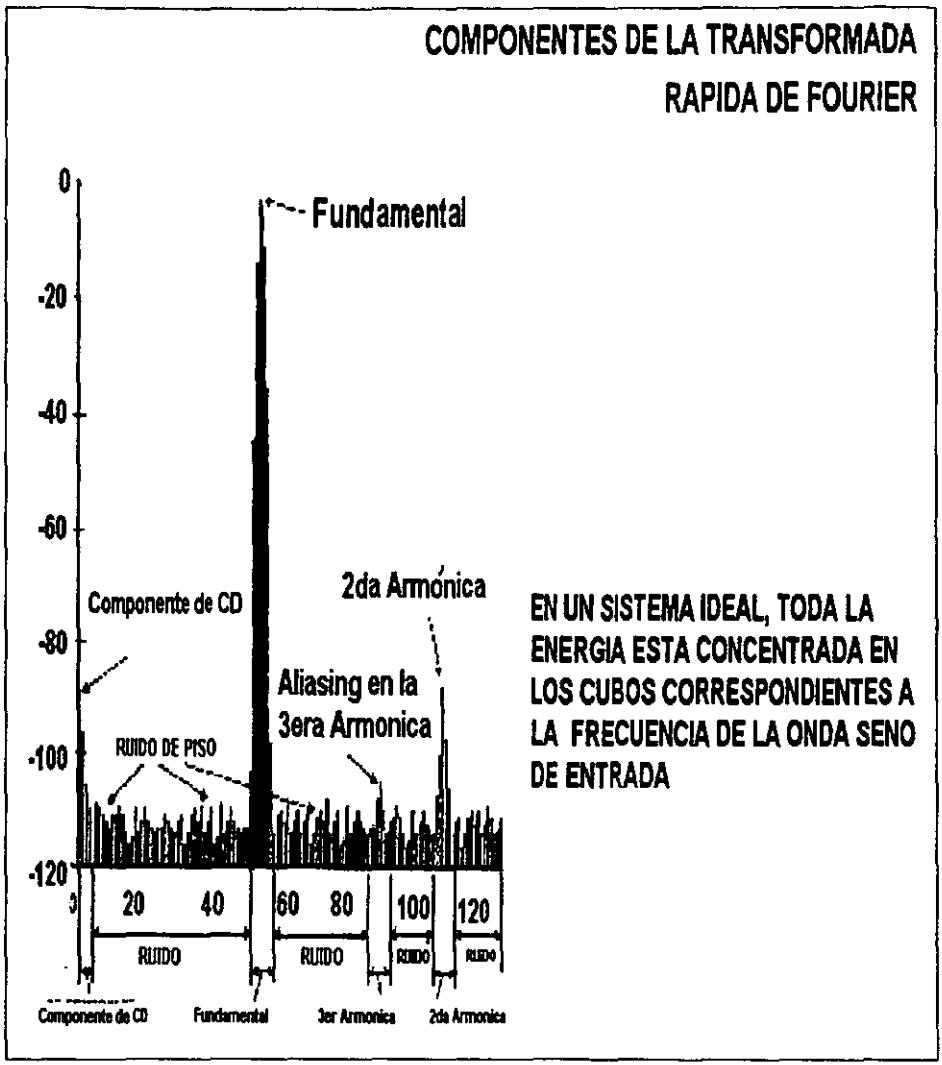

COMPONENTES DE LA TRANSFORMADA
RAPIDA DE FOURIER
0
-20
-40
-60
-80
-100
-120
Fundamental
Componente de CD
2da Armónica
Aliasing en la
3era Armonica
RUIDO DE PISO
EN UN SISTEMA IDEAL, TODA LA ENERGIA ESTA CONCENTRADA EN LOS CUBOS CORRESPONDIENTES A LA FRECUENCIA DE LA ONDA SENO DE ENTRADA
0
20
40
60
80
100
120
RUIDO
RUIDO
RUIDO
RUIDO
Componente de CD
Fundamental
3er Armonica
2da Armonica

Fig.3.17.

III.9.0. Diagrama a Bloques para Calcular el Desarrollo Dinámico

Son cuatro las medidas de desarrollo que pueden ser calculadas de un espectro de potencia obtenido de l *FFT*: *Signal-to-Noise Ratio (SNR), Signal-to-Noise, Distortion (SINAD), Signal-to-dsitortion (SDR), y Signal-to-Peak-Noise (SPN).* Cada parámetro da una perspicacia a las características de linealidad y ruido del sistema. Estos son utilizados en los bancos de desarrollo o identificando problemas específicos de desarrollo. Fig.3.18.

Con el espectro de potencia construido, el desarrollo estadístico dinámico puede ser calculado. Un algoritmo es utilizado para identificar las varias subsecciones del espectro de potencia. Primero el espectro es buscado por el *bin* más alto, localizando la señal fundamental. La potencia de la señal fundamental es calculada con consideración por el tipo de *window* utilizado por el algoritmo.

La siguiente sección identifica los *bins* de los off*sets* de CD. El tamaño del área depende del tipo *window* utilizado en el algoritmo. El CD no se utiliza para medir el desarrollo dinámico.

Con las secciones definidas de la señal fundamental y la señal de CD, la potencia que permanece en los *bins* se suman y un radio determinado versus la señal de potencia fundamental. Este radio es el valor *SINAD (Signal-to Distortion).*

El siguiente paso es para localizar el segundo bin más alto fuera de las áreas de la fundamental y del *offset de* CD. Este *bin* localiza el área del pico de ruido. La potencia en el pico de la señal de ruido se calcula y el radio del *Signal-to-Peak-Noise (SPN)* se obtiene.

El último paso es para localizar las armónicas secundarias asociadas con la señal fundamental. Aquí el algoritmo busca el espectro para las armónicas sin *aliasing* o con *aliasing*. La potencia asociada con las armónicas secundarias se considera una distorsión. Calculando el poder de la distorsión se tiene que lidiar para obtener el *SD* y los radios *SNR*.

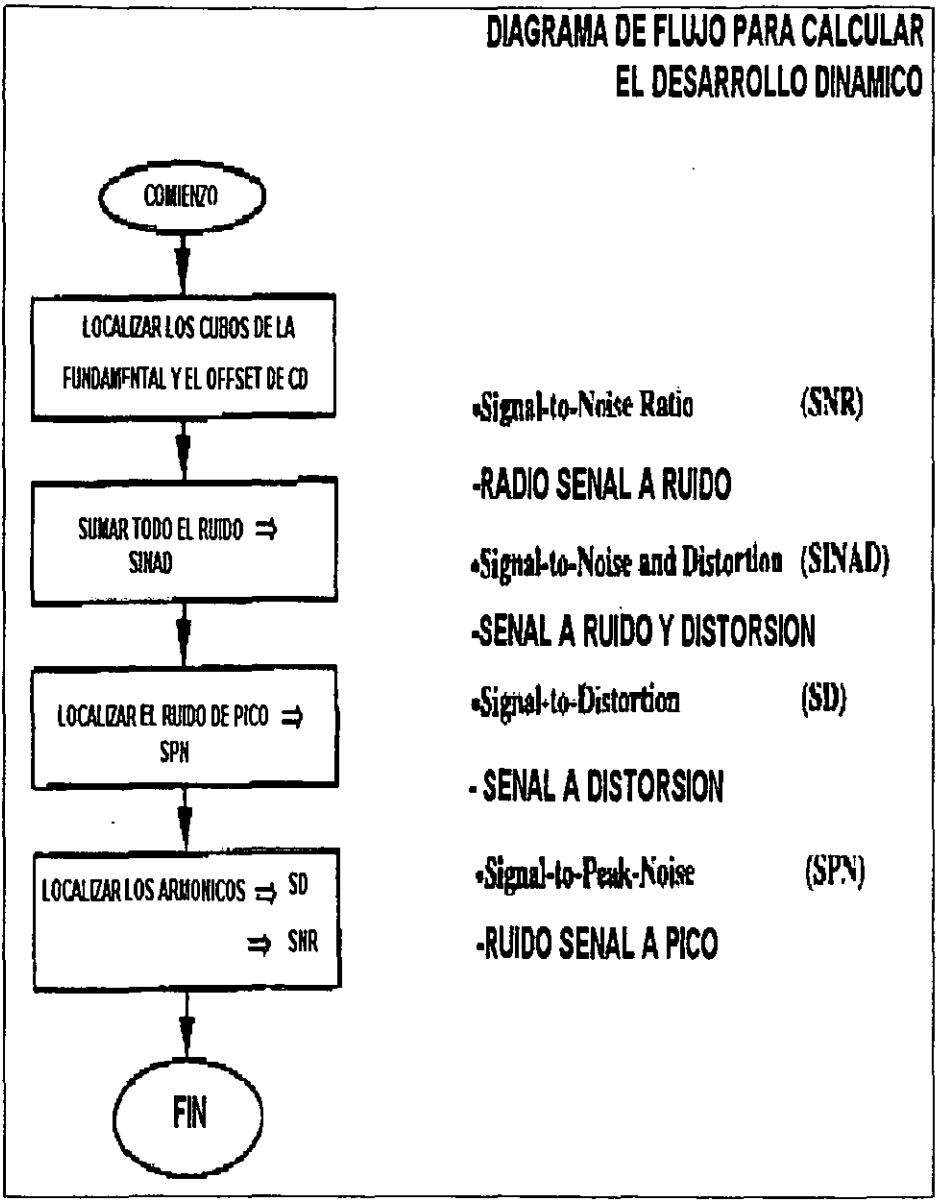


DIAGRAMA DE FLUJO PARA CALCULAR
EL DESARROLLO DINAMICO
COMIENZO
LOCALIZAR LOS CUBOS DE LA FUNDAMENTAL Y EL OFFSET DE CD
SUMAR TODO EL RUIDO ⇒ SINAD
LOCALIZAR EL RUIDO DE PICO ⇒ SPN
LOCALIZAR LOS ARMONICOS ⇒ SD
⇒ SNR
FIN
•Signal-to-Noise Ratio (SNR)
-RADIO SENAL A RUIDO
•Signal-to-Noise and Distortion (SINAD)
-SENAL A RUIDO Y DISTORSION
•Signal-to-Distortion (SD)
- SENAL A DISTORSION
•Signal-to-Peak-Noise (SPN)
-RUIDO SENAL A PICO

Fig..3.18.

III.9.1. *Signal to Noise and Distortion Ratio (SINAD)*

El signal to Noise and Distortion Ratio– Radio de la Señal de Ruido y distorsión son una medida de emisión en la banda de ruido y una distorsión introducida dentro de la señal. La magnitud *rms* de la entrada de la señal seno es comparada a la suma de todas las demás frecuencias, excepto la componente de CD. El radio de la señala de entrada para la suma del ruido individualmente y los componentes armónicos están expresados en términos de decibeles (dB). Entre más alto sea la figura del *SINAD*, mejor será el desarrollo general.

III.9.2. *Total Harmonic Distortion plus Noise (THD+N)*

Distorsión total de las armónicas más Ruido - *Total Harmonic Distortion plus Noise (THD+N)* son similares al *SINAD* excepto por que el numerador y el denominador están intercambiados. *THD +N* usualmente se utiliza en aplicaciones de audio como una figura única que amerita el desarrollo de la cuantificación y puede ser expresada en términos de porcentaje (%) o decibeles (dB). Fig.3.19.

% vs. Represenatación en dB

% THD + N = (THD + N) x 100 %

THD + N (dB) = 20 log (THD + N)

Ejemplo:

THD + N	
(típico)	**(máximo)**
-100 dB	**-94 dB**
0.001 %	**0.002%**

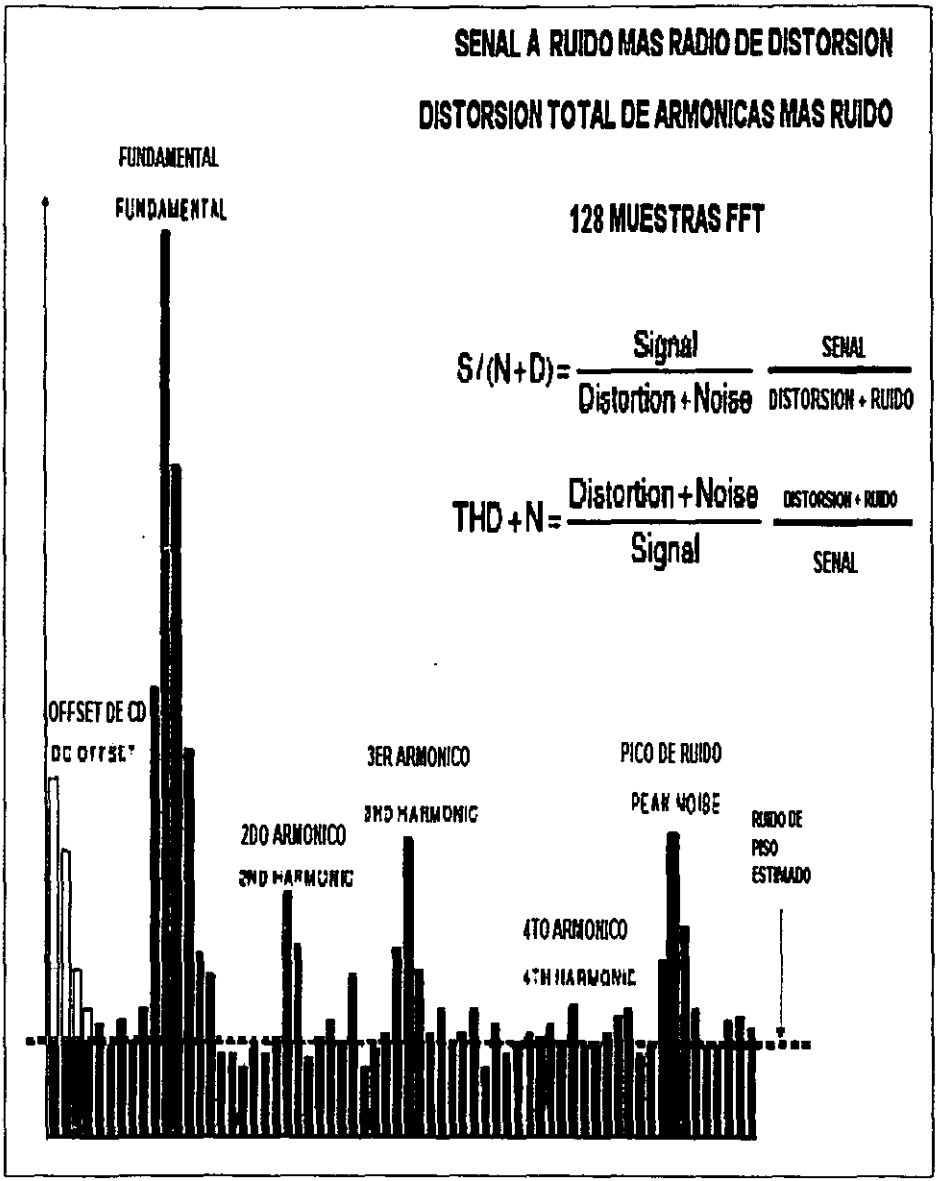

SENAL A RUIDO MAS RADIO DE DISTORSION
DISTORSION TOTAL DE ARMONICAS MAS RUIDO
FUNDAMENTAL
FUNDAMENTAL
128 MUESTRAS FFT
S/(N+D)= Signal / Distortion + Noise
SENAL / DISTORSION + RUIDO
THD+N= Distortion + Noise / Signal
DISTORSION + RUIDO / SENAL
OFFSET DE CD
DC OFFSET
2DO ARMONICO
2ND HARMONIC
3ER ARMONICO
3RD HARMONIC
4TO ARMONICO
4TH HARMONIC
PICO DE RUIDO
PEAK NOISE
RUIDO DE PISO ESTIMADO

Fig.3.19.

III.9.3. *Signal to Peak Noise Ratio (SPN)*

El ruido del pico de la señal - *Signal to Peak Noise* es una medida del ruido más alto o la distorsión más alta introducida en la señal. Las componentes de ruido o distorsión más altas determinan el nivel más bajo que una señal en el cual una señal puede ser detectada.

La magnitud *rms* de una señal de entrada se compara con la magnitud *rms* de la componente de ruido o distorsión más alta.

El radio de la señal de entrada en los componentes del ruido de pico, es expresado en términos de decibeles (dB). Entre más alto sea el valor del SPN, es mejor la detección del nivel bajo de señales. Fig.3.20.

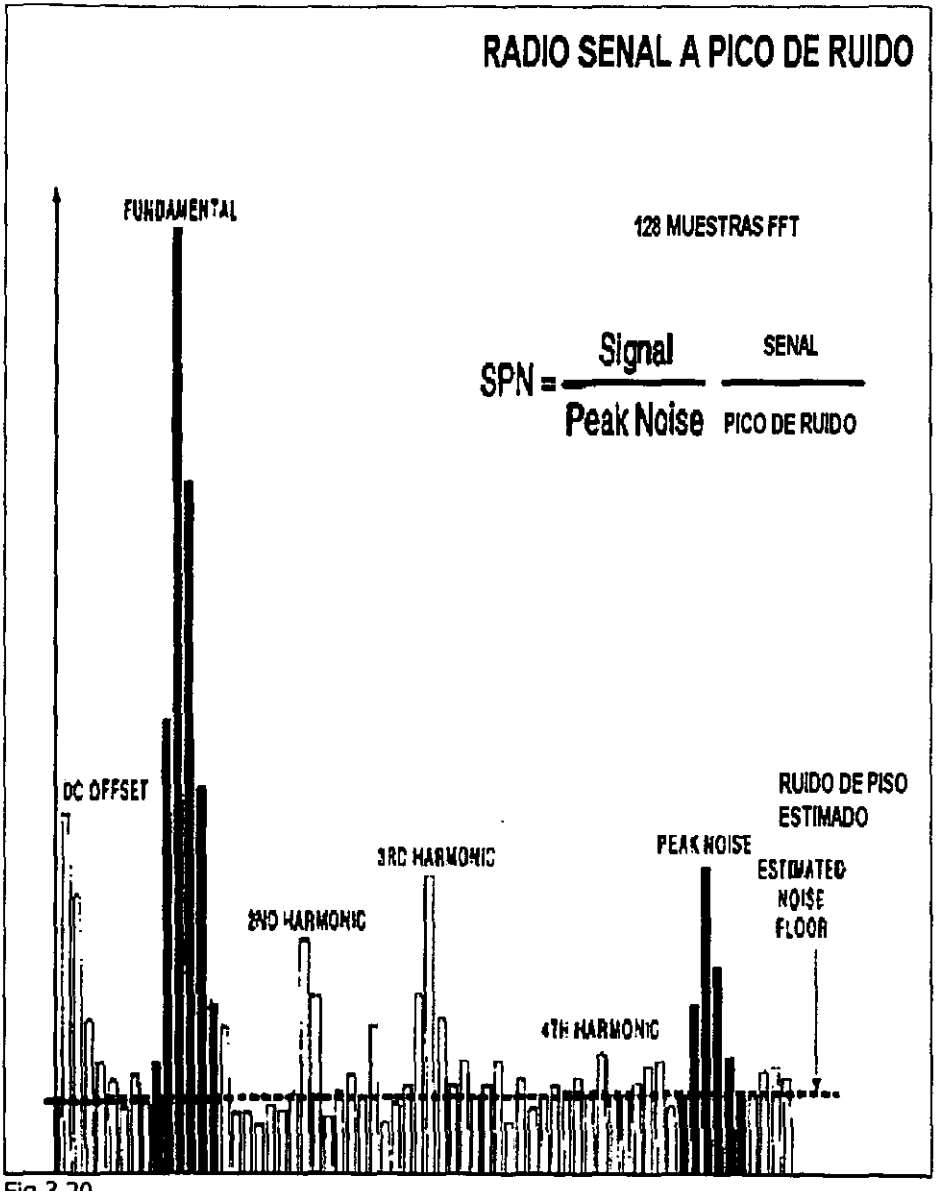

RADIO SENAL A PICO DE RUIDO
FUNDAMENTAL
128 MUESTRAS FFT
SPN = Signal / Peak Noise
SENAL
PICO DE RUIDO
DC OFFSET
2ND HARMONIC
3RD HARMONIC
4TH HARMONIC
PEAK NOISE
RUIDO DE PISO ESTIMADO
ESTIMATED NOISE FLOOR

Fig.3.20.

III.9.4. *Signal to Distortion Ratio (SDR)*

Signal to Distortion Ratio es una medida de la distorsión armónica introducida dentro de la señal. La distorsión armónica degrada la fidelidad de la señal medida. Una buena figura *Signal to Distortion* sugiere una buena integración no lineal (INL). La magnitud rms de la señal de entrada se compara con la suma de los componentes armónicos (señales múltiplos de enteros de las frecuencias de la señal de entrada). El radio de la señal de entrada para la suma de componentes armónicos es expresada en términos de decibeles (dB). Entre más alto sea el valor del SDR, más alta es la fidelidad. Fig.3.21.

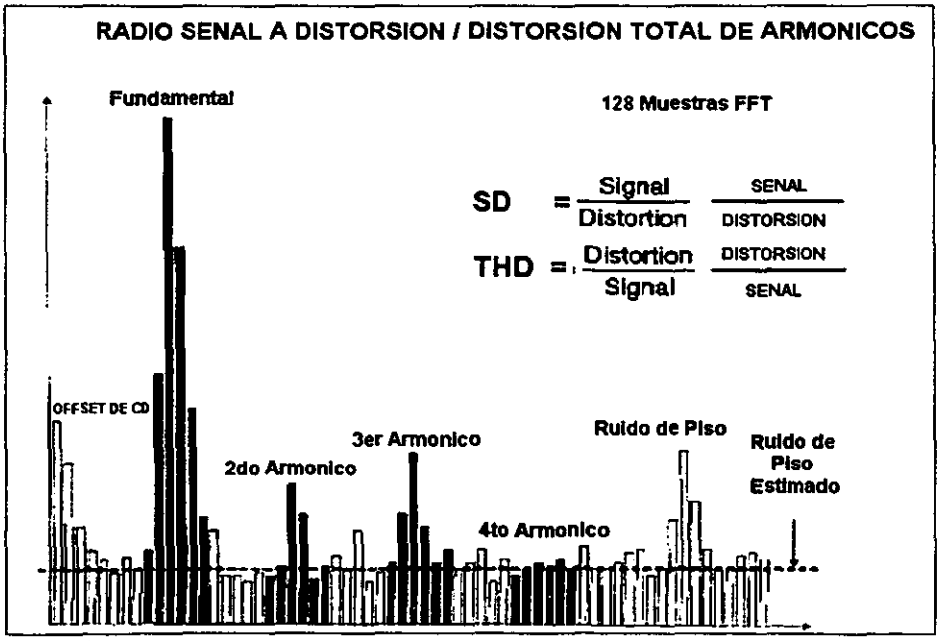


Fig.3.21.

III.9.5. *Total Harmonic Distortion (THD)*

Total Harmonic Distortion es similar al *SDR* excepto que el número se cambia en lugar del numerador y viceversa. *THD* regularmente se utiliza en aplicaciones de audio como una medida de linealidad y es expresada en términos de porcentaje(%) o decibeles(dB).

III.9.6. *Signal to Noise Ratio (SNR)*

El *Signal to Noise Ratio* es una medida de la banda ancha de ruido introducida dentro de la señal. El ruido corrompe las señales dinámicas, degradando la exactitud a niveles bajos de voltaje. La magnitud *rms* de una señal de entrada es comparada con la suma de los componentes individuales de ruido y es expresada en términos de dB. Entre más alta sea la figura del *SNR*, la banda ancha de ruido será más baja. El número *SNR* puede ser trasladado dentro de bits efectivos. Fig.3.22.

Bits efectivos = (SNR-1.76 dB/ 6.02)

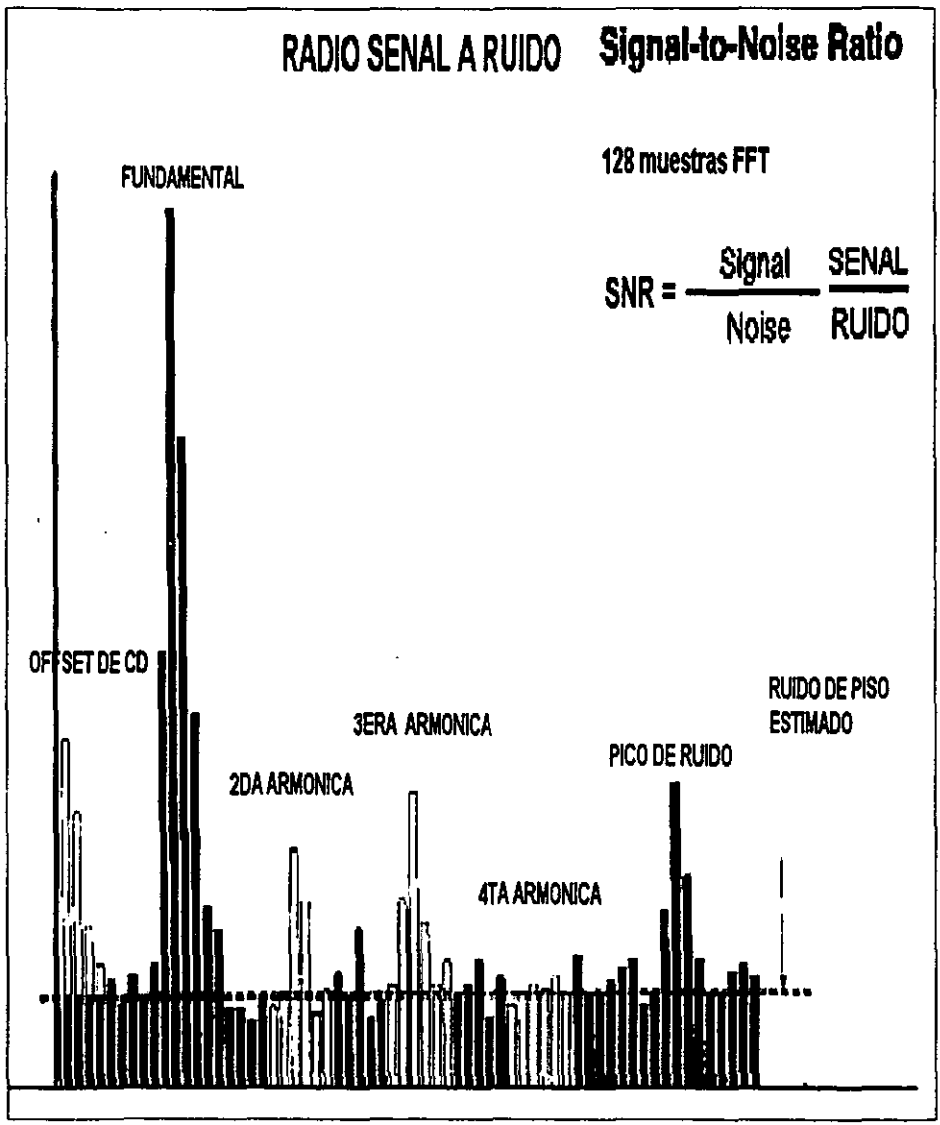

RADIO SENAL A RUIDO
Signal-to-Noise Ratio
128 muestras FFT
SNR = Signal / Noise
SENAL / RUIDO
FUNDAMENTAL
OFFSET DE CD
2DA ARMONICA
3ERA ARMONICA
4TA ARMONICA
PICO DE RUIDO
RUIDO DE PISO ESTIMADO

Fig.3.22.

III.10.0. Tamaño del *set* de muestras

El tamaño del *set* de muestras utilizado con el algoritmo *FFT* determina la resolución de frecuencia y la mancha de ruido la cual hace el ruido de piso. Más información está contenida en un *set* grande de muestras que un *set* pequeño de muestras. Un *FFT* de un *set* grande de muestras usa la información para proveer una resolución de frecuencia más fina. La amplitud de banda es igual a la frecuencia de muestreo dividida por el número de muestras (Fs/N). Cuando se muestrea a 50 khz, la amplitud del *bin* para 8192 muestras de *FFT* es 976.6 Hz y 61.0 Hz para 8192 muestras de *FFT*. Fig.3.23.

Notar que el ruido de piso para 8192 muestras de *FFT* es menor que 512 muestras de *FFT*. La cantidad total de ruido entre 0 y 25 khz es igual en ambos casos. Pero la amplitud del *bin* para 512 muestras de *FFT* es 16 veces mayor que 8192 muestras *FFT*. Debido a que el *bin* es 16 veces más grande este contendrá 16 veces mayor la mancha en la banda del ruido que será igualmente distribuido a través del espectro. Debido al piso de ruido de 8192 muestras de *FFT* deberá ser 12 dB menor que el ruido de piso para 512 muestras de *FFT*. Con un menor ruido de piso, las señales de bajo nivel son más aparentes.

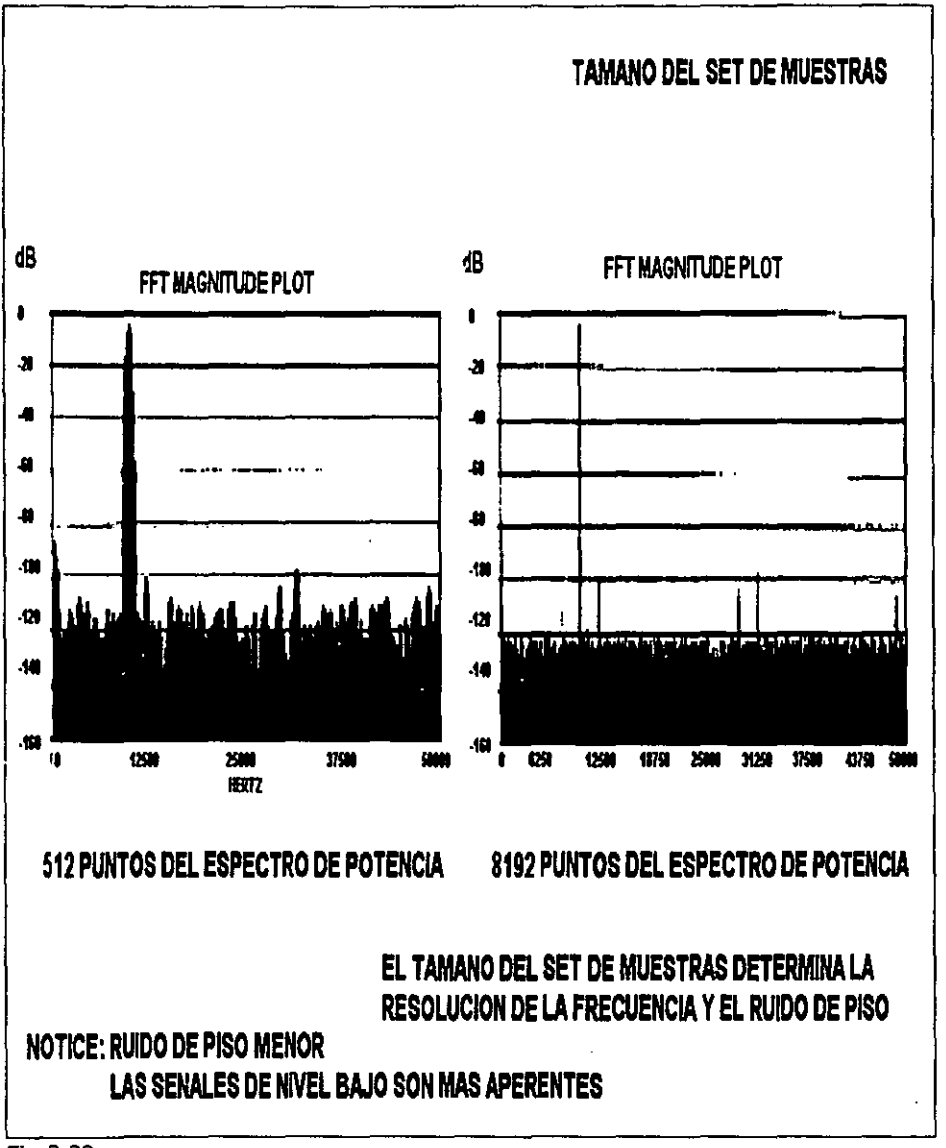

TAMANO DEL SET DE MUESTRAS
dB
FFT MAGNITUDE PLOT
HERTZ
dB
FFT MAGNITUDE PLOT
512 PUNTOS DEL ESPECTRO DE POTENCIA
8192 PUNTOS DEL ESPECTRO DE POTENCIA
EL TAMANO DEL SET DE MUESTRAS DETERMINA LA
RESOLUCION DE LA FRECUENCIA Y EL RUIDO DE PISO
NOTICE: RUIDO DE PISO MENOR
LAS SENALES DE NIVEL BAJO SON MAS APERENTES

Fig.3.23.

Los niveles de energía a frecuencias específicas permanecerán descargadas con el incremento del tamaño de las muestras. La energía de una señal senoidal permanecerá concentrada en una frecuencia específica. La energía asociada con el *bin* específico permanecerá constante.

III.11.0. Promediando las Transformadas Rápidas de Fourier (*FFT´s*)

Los espectros de potencia producidas de una *FFT* utilizan datos muestreados. Los datos muestreados son un representante del fenómeno actual. La estadística calculada de los datos de muestra solo es un estimado del valor actual, y tiene cierta incertidumbre. Promediando los múltiples *FFT´s* se reduce las desviaciones asociadas con los datos estadísticos calculados de los datos muestreados. Fig.3.24.

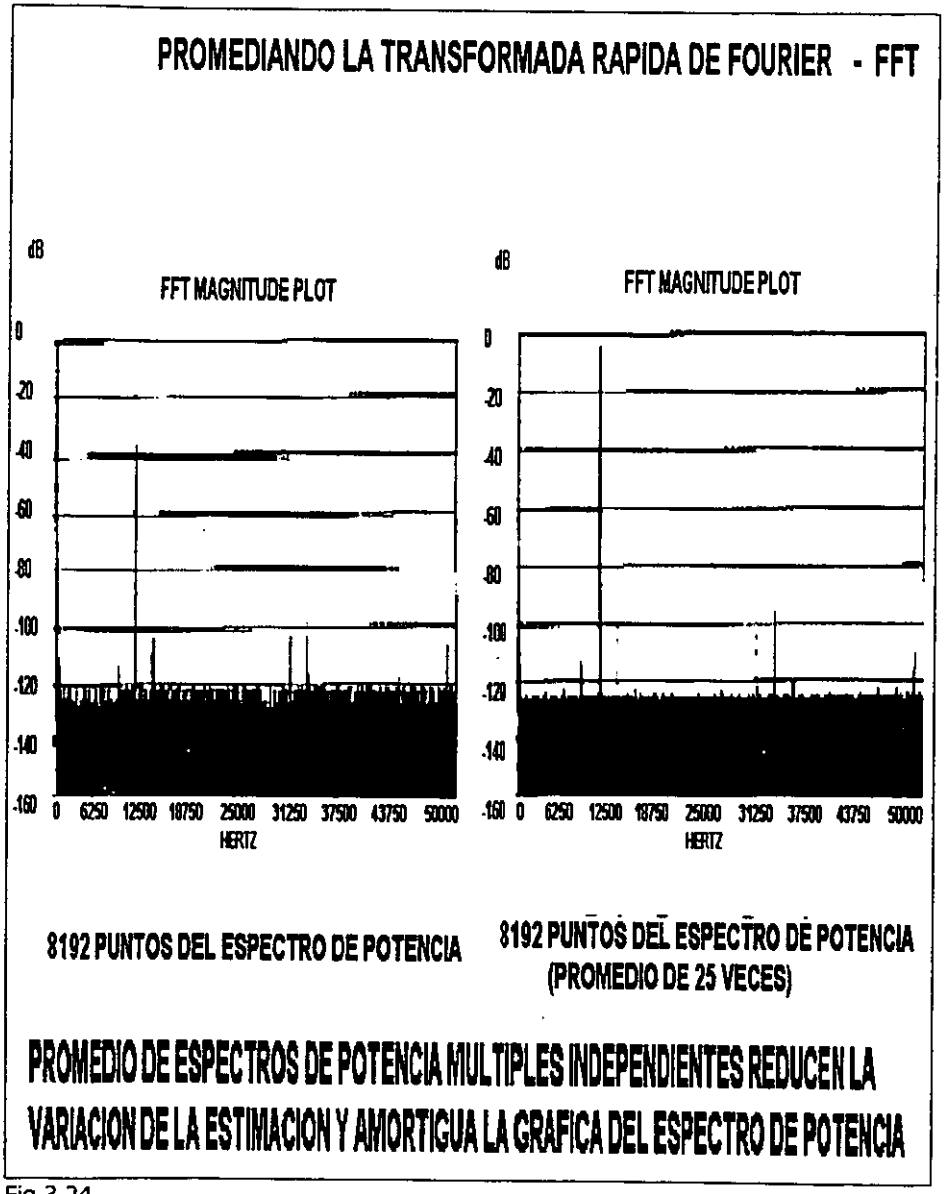

PROMEDIANDO LA TRANSFORMADA RAPIDA DE FOURIER - FFT
dB
FFT MAGNITUDE PLOT
0
-20
-40
-60
-80
-100
-120
-140
-160
0 6250 12500 18750 25000 31250 37500 43750 50000
HERTZ
dB
FFT MAGNITUDE PLOT
0
-20
-40
-60
-80
-100
-120
-140
-160
0 6250 12500 18750 25000 31250 37500 43750 50000
HERTZ
8192 PUNTOS DEL ESPECTRO DE POTENCIA
8192 PUNTOS DEL ESPECTRO DE POTENCIA
(PROMEDIO DE 25 VECES)
PROMEDIO DE ESPECTROS DE POTENCIA MULTIPLES INDEPENDIENTES REDUCEN LA
VARIACION DE LA ESTIMACION Y AMORTIGUA LA GRAFICA DEL ESPECTRO DE POTENCIA

Fig.3.24.

Dos funciones de densidad de potencia se calculan, la primera utilizando un *set* de muestras, la segundo se promedia 22 *FFTs*. En la función de densidad de potencia se calcula de una sola muestra de 8192 muestras de la *FFT*, notando que el ruido de piso varía entre -100dB y –140 dB. La segunda función de densidad de potencia promedia 25 *FFT´s* para reducir la incertidumbre del muestreo de datos. El ruido de piso para promediar el FFT varía solo alrededor de 5 dB.

Promediando se reduce la incertidumbre del muestreo de datos. Promediando múltiples *FFT´s* se produce un suave ruido de piso, haciendo coherente el ruido y fácil de identificar. La incertidumbre debida a la incertidumbre de la aleatoriedad es reducida por 1/ _/n , donde n es el no es el de muestras promediadas.

III.12.0. Comparación de dos *SINADs* diferentes

La figura del *Signal to Noise y el Distortion (SINAD)* se usa como medida para propósitos de desarrollo generales. El diagrama de la izquierda muestra la densidad del espectro de potencia para 2 diferentes CAD los cuales tienen un valor idéntico, valor de *SINAD*. A pesar de que dos convertidores analógicos digitales tengan idéntico *SINAD*, tienen diferentes características de desarrollo.

EL *plot* de la izquierda representa un CAD con pobre linealidad, siendo esta la consecuencia, el resultado de la distorsión armónica. El número del *Sigal to Distortion (SDR)* es 81.419 dB. El SNR es muy bueno a 94.097 dB. Este CAD puede ser utilizado en una aplicación donde la linealidad no es importante, como señales medibles de bajo nivel. Fig.3.25.

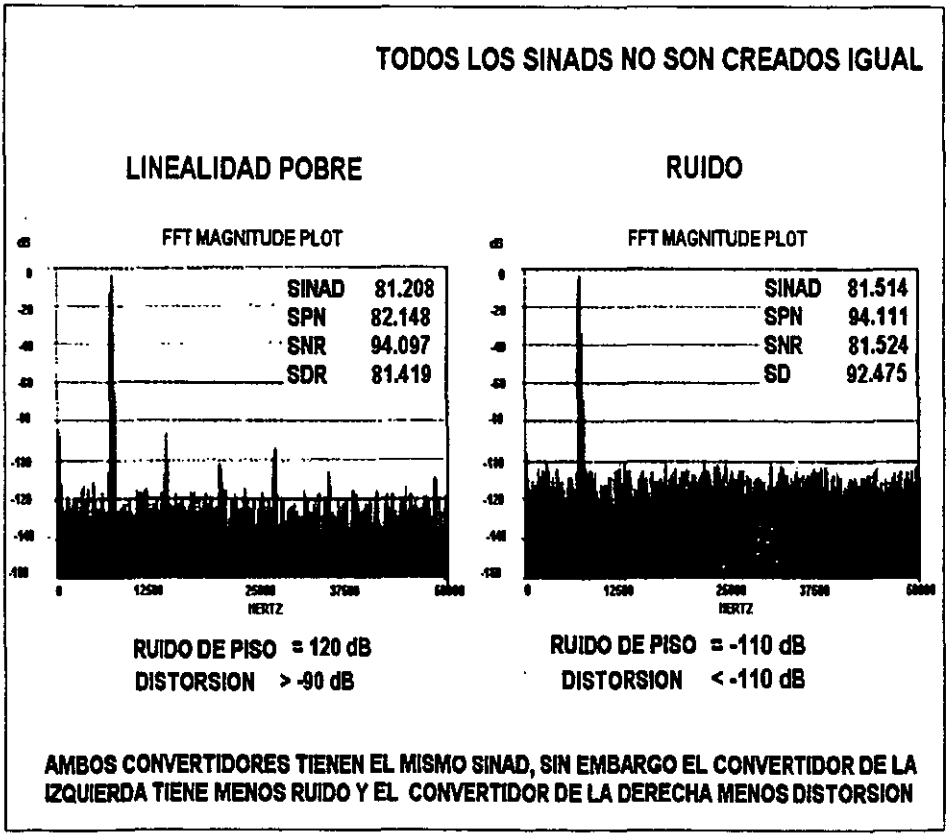


Fig.3.25.

El otro *plot* de densidad de potencia representa un CAD con un ruido alto. La función de transferencia posee una buena linealidad, representada por un SDR de 92.475 dB, sin embargo el SNR es 81.542 dB. Este ADC puede ser utilizado en una aplicación que requiera de baja distorsión.

Ciertas aplicaciones se interesan en el desarrollo dinámico con un rango de frecuencia específico. Un CAD el cual se muestrea a 51.2 kHz tiene una frecuencia Nyquist de 25.6 kHz. La estadística de la señal normalmente se colectan utilizando la energía entre CD y las frecuencia Nyquist. La función de densidad de potencia en la izquierda calcula su estadística utilizando la energía espectral del CD a 25.6 kHz.

En aplicaciones de audio la banda de interés es hasta los 20 kHz. El espectro de su interés es menor que la frecuencia de Nyquist. La estadística en la segunda función de densidad de potencia utiliza solo la energía de las señales en la banda de interés. Aquí, el *Signal to Noise Ratio (SNR)* debajo de los 20 kHz es de 93.203 dB.

III.13.0. La Señal menor que el Ancho de un Código. Si la Señal esta por debajo de los 107dB de la Escala Completa *(FS)*

Una concepción errónea es que una señal más baja que el rango dinámico de frecuencia puede no ser detectado ni medido. Utilizando un CAD ideal de 16 bits, el mejor rango dinámico es 98 dB, señales que están por debajo de los 100dB del *Full Scale* pueden detectarse. Fig.3.26.

En este ejemplo un CS5101A de Crystal se utilizan con un voltaje de referencia de 4.5 V, muestreando a 100 k Hz. La amplitud del código ideal para estas condiciones es de 137 uV; como en la siguiente fórmula:

$$137uV = (+4.5V)-(-4.5) \ / \ 2^{16}$$

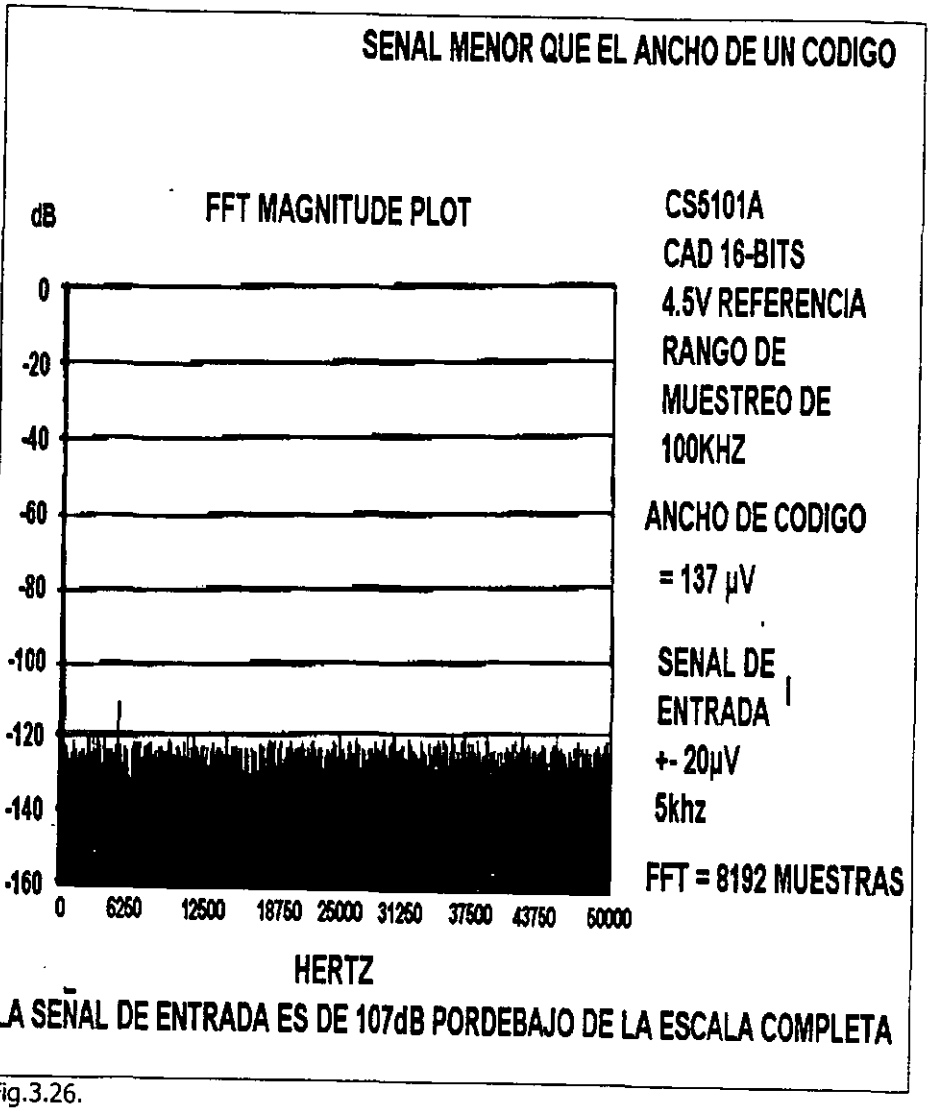

SENAL MENOR QUE EL ANCHO DE UN CODIGO
dB
FFT MAGNITUDE PLOT
0
-20
-40
-60
-80
-100
-120
-140
-160
0
6250
12500
18750
25000
31250
37500
43750
50000
HERTZ
CS5101A
CAD 16-BITS
4.5V REFERENCIA
RANGO DE
MUESTREO DE
100KHZ
ANCHO DE CODIGO
= 137 µV
SENAL DE
ENTRADA
+- 20µV
5khz
FFT = 8192 MUESTRAS
LA SEÑAL DE ENTRADA ES DE 107dB PORDEBAJO DE LA ESCALA COMPLETA

Fig.3.26.

Una señal a 20 uV, 5 kHz se aplica a la entrada analógica. La señal de entrada de 20 uV es 107 dB por debajo del *Full Scale*. Utilizando muestreo *FFT* de 8192, La señal de bajo nivel a 5kHz se detecta fácilmente.

III.14.0. CAD´s como Componentes de Sistemas más grandes

Los CAD son componentes de grandes sistemas que usualmente incluyen circuitos digitales y analógicos, fuentes de potencia, partes electromecánicas, cables, conecctores y alambres. El desarrollo puede ser degradado por la interacción entre los subsitemas.

Pruebas para cuantificar el desarrollo y los problemas en un sistema puede ser en diferentes puntos de este y bajo varias condiciones. Fig.3.27.

Pruebas aplicadas a un sistema específico o a un componente se usa para aislar los problemas hacia un componente o evento específico.

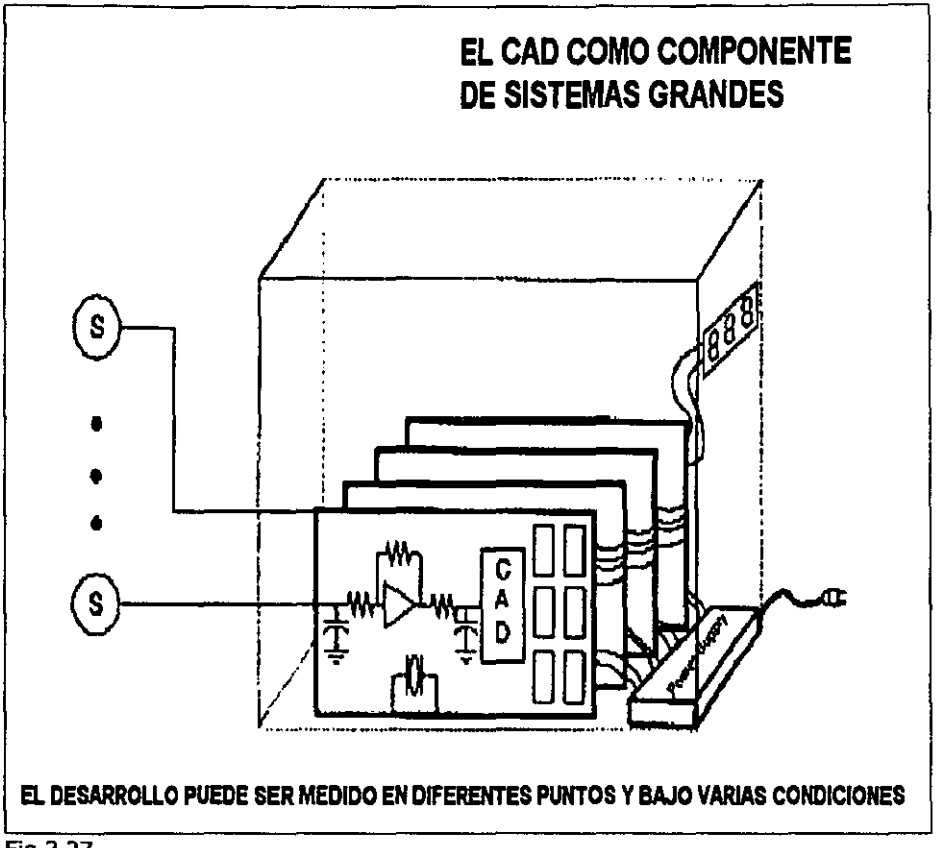


Fig.3.27.

Un método para aislar problemas es el medir el desarrollo de los sistemas básicos para establecer una línea de base. El siguiente paso es integrar el subsistema y medir el desarrollo como un proceso de integración y así hasta llegar al sistema completo.

III.14.1. Entrada del CDA aterrizada para Estabilizar la línea en Base

Para establecer una línea de base para el desarrollo, el CAD es probado con una entrada a tierra. Una de las dos pruebas en la colección de datos es un histograma y otra es un *FFT.* Cada prueba provee diferentes perspicacias dentro de la operación del circuito.

Con la entrada aterrizada, la FFT muestra un ruido de piso plano, el cual indica que el ruido es aleatorio y eventualmente esparcido a través del espectro de frecuencia.

El histograma muestra que la salida media de un CAD es −1.59 cuentas, indicando un *offset* de −1.59, y una desviación estándar es 0.54, indicando un nivel de ruido *rms* de 0.54 cuentas. Fig.3.28.

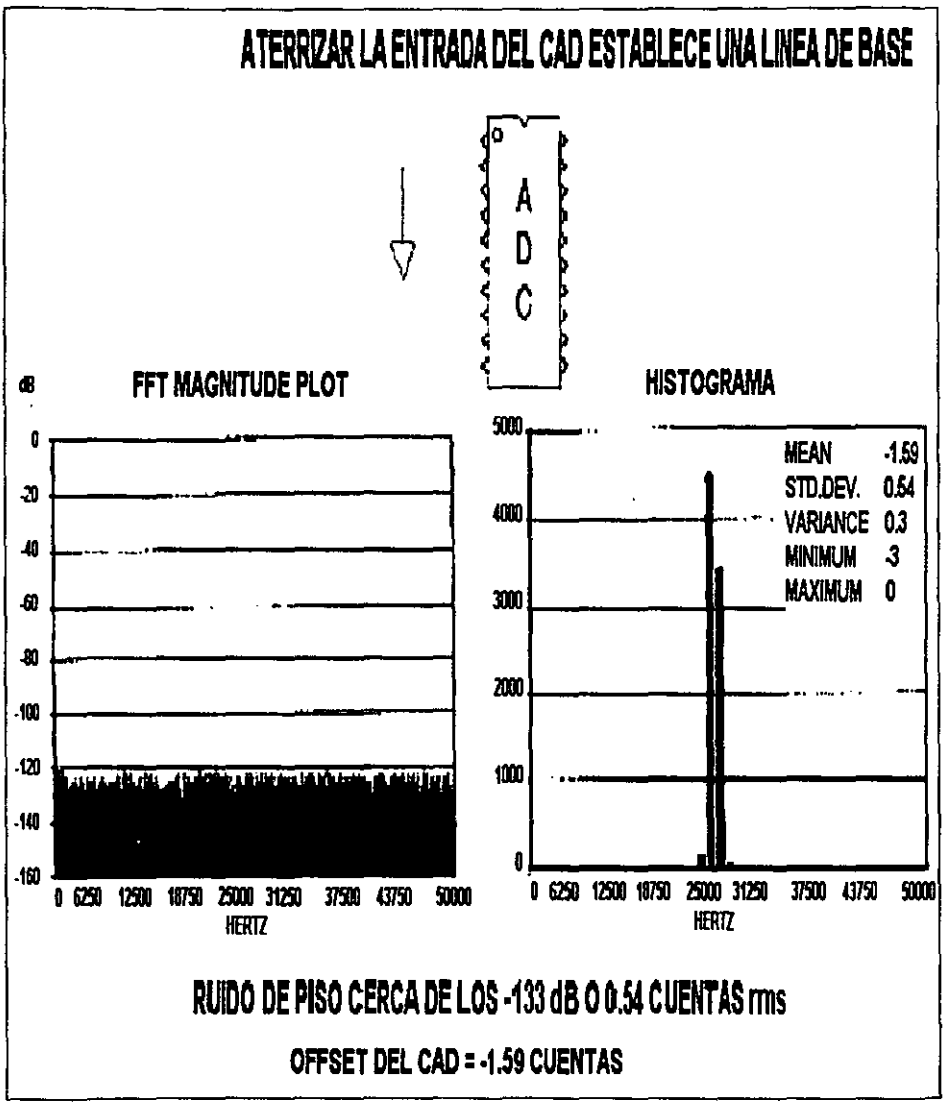

ATERRIZAR LA ENTRADA DEL CAD ESTABLECE UNA LINEA DE BASE
A
D
C
dB
FFT MAGNITUDE PLOT
0
-20
-40
-60
-80
-100
-120
-140
-160
0 6250 12500 18750 25000 31250 37500 43750 50000
HERTZ
HISTOGRAMA
5000
4000
3000
2000
1000
0
MEAN -1.59
STD.DEV. 0.54
VARIANCE 0.3
MINIMUM -3
MAXIMUM 0
0 6250 12500 18750 25000 31250 37500 43750 50000
HERTZ
RUIDO DE PISO CERCA DE LOS -133 dB O 0.54 CUENTAS rms
OFFSET DEL CAD = -1.59 CUENTAS

Fig.3.28.

Entre mayor sea el número de componentes adheridos al CAD podemos esperar que el *offset* cambie y el ruido *rms* se incremente. Sí los cambios son muy severos, las modificaciones del circuito y las mejoras tendrán que ser advertidas e implementadas.

III.14.2. Circuito Analógico de Entrada

Después que algunos circuitos analógicos (Comparadores analógico) de *front-end* son adheridos, el *FFT* y el histograma son repetidos y comparados a las pruebas previas. Fig.3.29.

En la función de densidad de potencia se muestra, pequeños picos de ruido que llegan a estar aparentemente debajo de los -120 dB. Esta es una indicación que el ruido de algún reloj puede ser acoplado dentro de un circuito analógico. El nivel de ruido coherente continua siendo bajo. Nota que el ruido de piso se ha incrementado hasta los 130 dB.

La media cambia y la desviación estándar se incrementa en el histograma, indicando un cambio en el *offset* y sumando ruido debido a los componentes.

Recordando los voltajes de ruido aleatorios se suman teniendo la raíz cuadrada de la suma de los cuadrados de los voltajes de ruido individuales.

$$\text{Ruido Total} = \sqrt{V1^2 + V2^2 + V3^2}$$

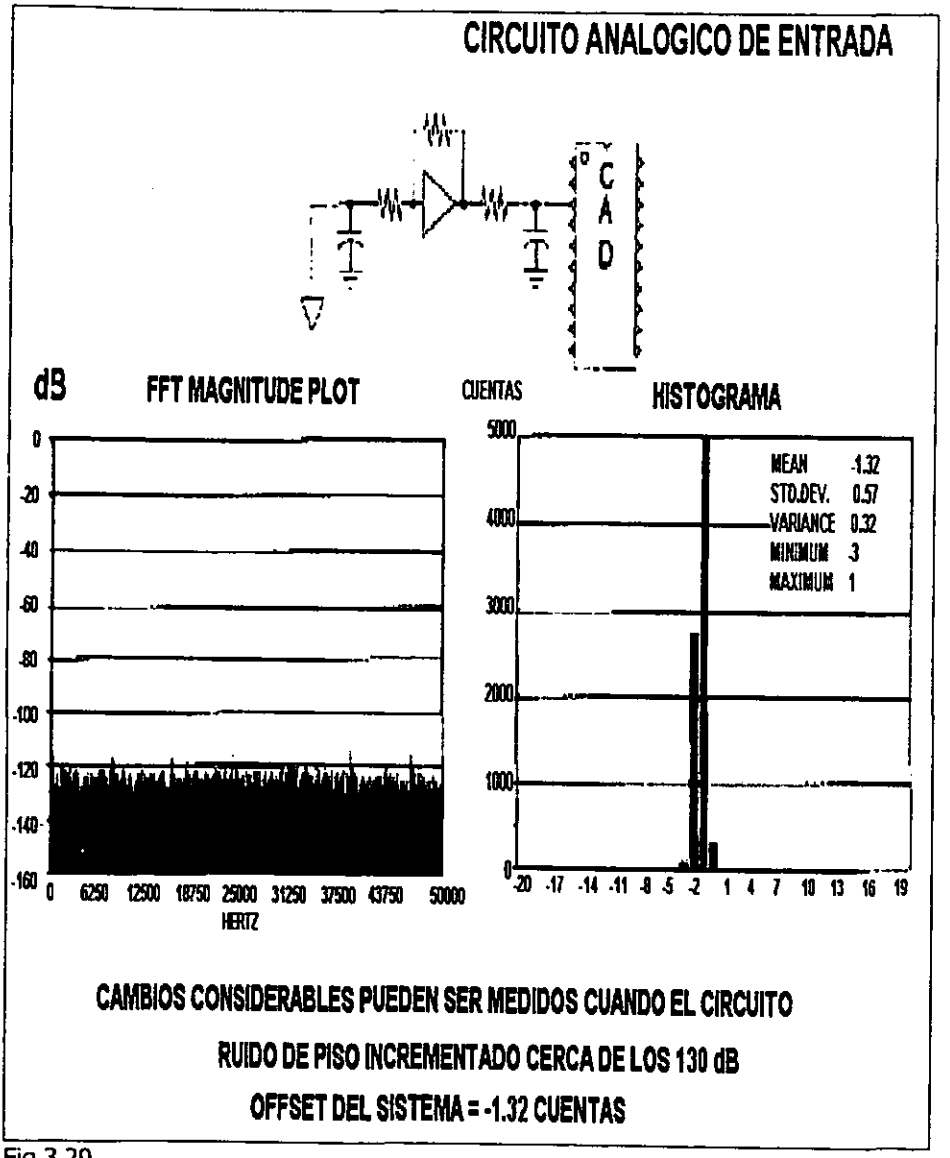

CIRCUITO ANALOGICO DE ENTRADA
C
A
D
dB
FFT MAGNITUDE PLOT
0
-20
-40
-60
-80
-100
-120
-140
-160
0 6250 12500 18750 25000 31250 37500 43750 50000
HERTZ
CUENTAS
HISTOGRAMA
5000
4000
3000
2000
1000
0
MEAN -1.32
STD.DEV. 0.57
VARIANCE 0.32
MINIMUM -3
MAXIMUM 1
-20 -17 -14 -11 -8 -5 -2 1 4 7 10 13 16 19
CAMBIOS CONSIDERABLES PUEDEN SER MEDIDOS CUANDO EL CIRCUITO
RUIDO DE PISO INCREMENTADO CERCA DE LOS 130 dB
OFFSET DEL SISTEMA = -1.32 CUENTAS

Fig.3.29.

La mayoría de los cambios en el *offset* y el ruido son relativamente pequeños.

3.14.3. Canal en Prueba

En esta prueba el *front-end* analógico del circuito completamente entero es adherido al CAD. El sensor es configurado al valor de su cero y los datos son coleccionados para la FFT y el histograma. Fig.3.30.

En el histograma, la media de muestreo es de −7.09 cuentas. Este es el *offset* para el *front-end* entero. El valor medio puede ser almacenado y restado de las lecturas del CAD para corregir el *offset* del sistema.

La FFT ilustra algunos resultados sorprendentes. Si solo la información del histograma se utilizara, se puede asumir que el ruido fue completamente aleatorio. La distribución del histograma parece poseer la familiar campana de Gauss. La función de densidad de potencia indica que no hay ruido aleatorio a 25 kHz. Conociendo la frecuencia del ruido, la fuente puede ser más fácilmente localizable y el problema arreglado.

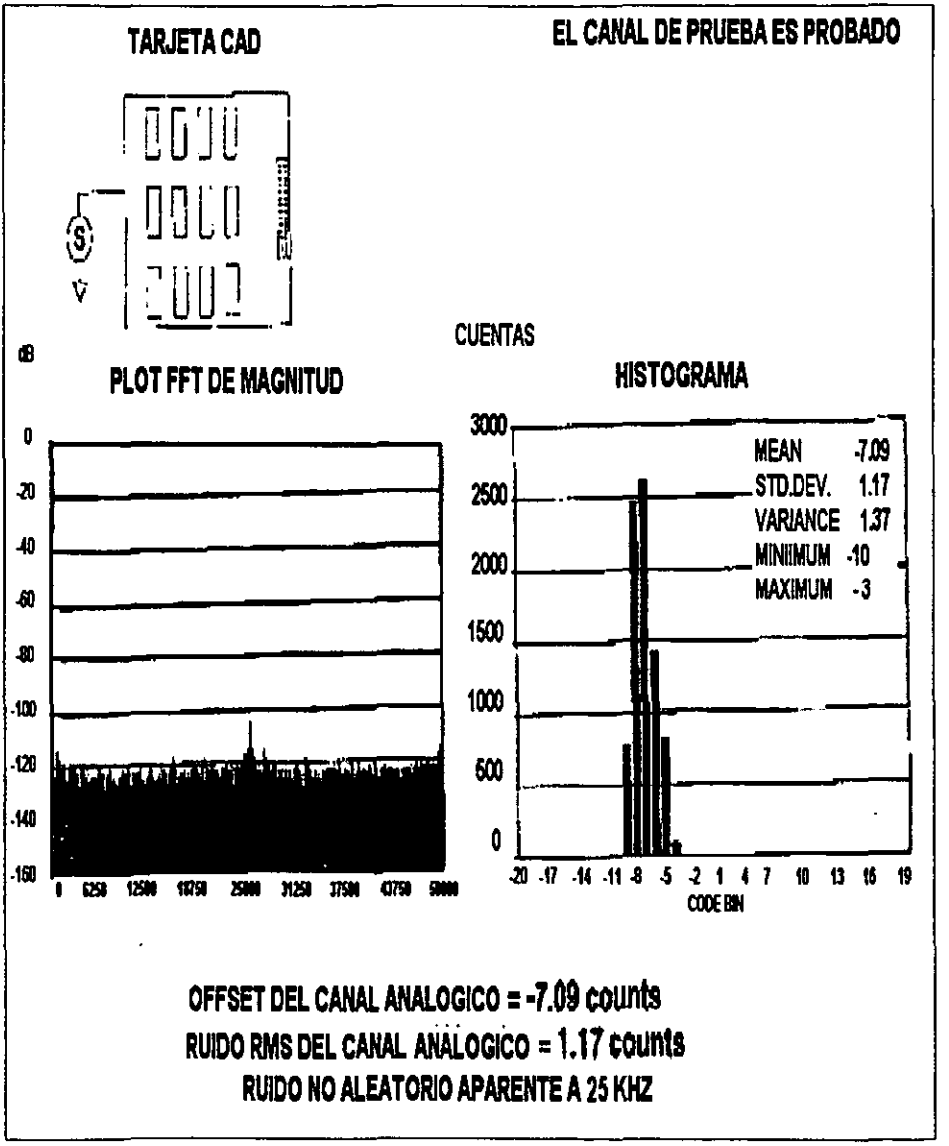

TARJETA CAD
EL CANAL DE PRUEBA ES PROBADO
dB
PLOT FFT DE MAGNITUD
0
-20
-40
-60
-80
-100
-120
-140
-160
0 6250 12500 18750 25000 31250 37500 43750 50000
CUENTAS
HISTOGRAMA
3000
2500
2000
1500
1000
500
0
MEAN -7.09
STD.DEV. 1.17
VARIANCE 1.37
MINIMUM -10
MAXIMUM -3
-20 -17 -14 -11 -8 -5 -2 1 4 7 10 13 16 19
CODE BIN
OFFSET DEL CANAL ANALOGICO = -7.09 counts
RUIDO RMS DEL CANAL ANALOGICO = 1.17 counts
RUIDO NO ALEATORIO APARENTE A 25 KHZ

Fig.3.30.

3.14.4. Desarrollo del sistema

La prueba final incorpora el sistema entero. Nuevamente el sensor es colocado al valor cero y el *set* de datos se colecta. La prueba ahora revela que el ruido dominante es no más grande que el aleatorio.

La función de densidad de potencia indica que hay un ruido significante cerca de los 25 kHz, 50 kHz, y la CD. Este ruido puede ser de la fuente de poder switcheada que opera alrededor de los 25kHz. Esto producirá ruido a 25 kHz y múltiplos de 25kHz. El ruido de alta frecuencia puede tener el efecto *aliasing* en las frecuencias más bajas.

Una interesante prueba para realizar sería cambiar la carga en la fuente de poder. La FFT puede ser utilizada para ver si la frecuencia y la magnitud del ruido cambia bajo diferentes condiciones de carga. Fig.3.31.

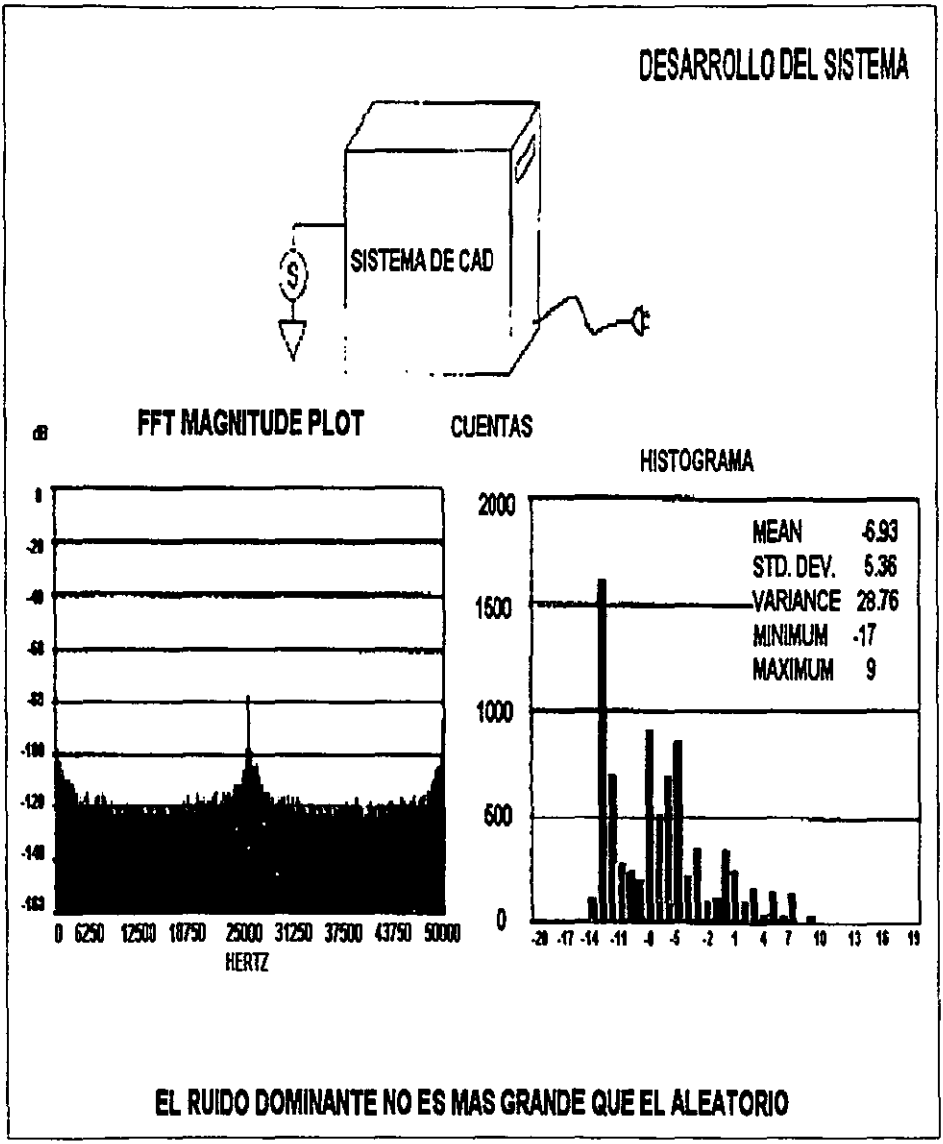

DESARROLLO DEL SISTEMA
SISTEMA DE CAD
FFT MAGNITUDE PLOT
CUENTAS
HISTOGRAMA
MEAN -6.93
STD. DEV. 5.36
VARIANCE 28.76
MINIMUM -17
MAXIMUM 9
HERTZ
EL RUIDO DOMINANTE NO ES MAS GRANDE QUE EL ALEATORIO

Fig.3.31.

3.14.5. Relación entre las *FFT´s* y el Histograma

Cuando el ruido es aleatorio, la *FFT* y la prueba del Histograma deberán correlacionarse una con otra. Cuando se mira a la función de densidad de potencia, un ruido plano de piso consiste de un ruido aleatorio. El ruido en todo el espectro puede ser calculado asignando la potencia del ruido a cada *bin.*

La función de densidad de potencia ilustrada es creada utilizando 1024 muestras. Debido al *FFT* tiene 512 *bins* (1024/2). Cada *bin* esta a –120 dB debajo de la entrada del *Full Scale.* La potencia del ruido para 512 *bins* es igual a –120dB + 10 log (512) o –92.9 dB. En este ejemplo los cálculos son fáciles con todos los 512 *bins* al mismo valor, sin embargo este método puede ser utilizado para estimar el piso de ruido estimando la función de densidad de potencia. Fig.3.32.

El valor *rms* de la señal a *Full Scale* es:

$$Full\ Scale\ Signal = 9V / 2\sqrt{2}\ Vrms$$

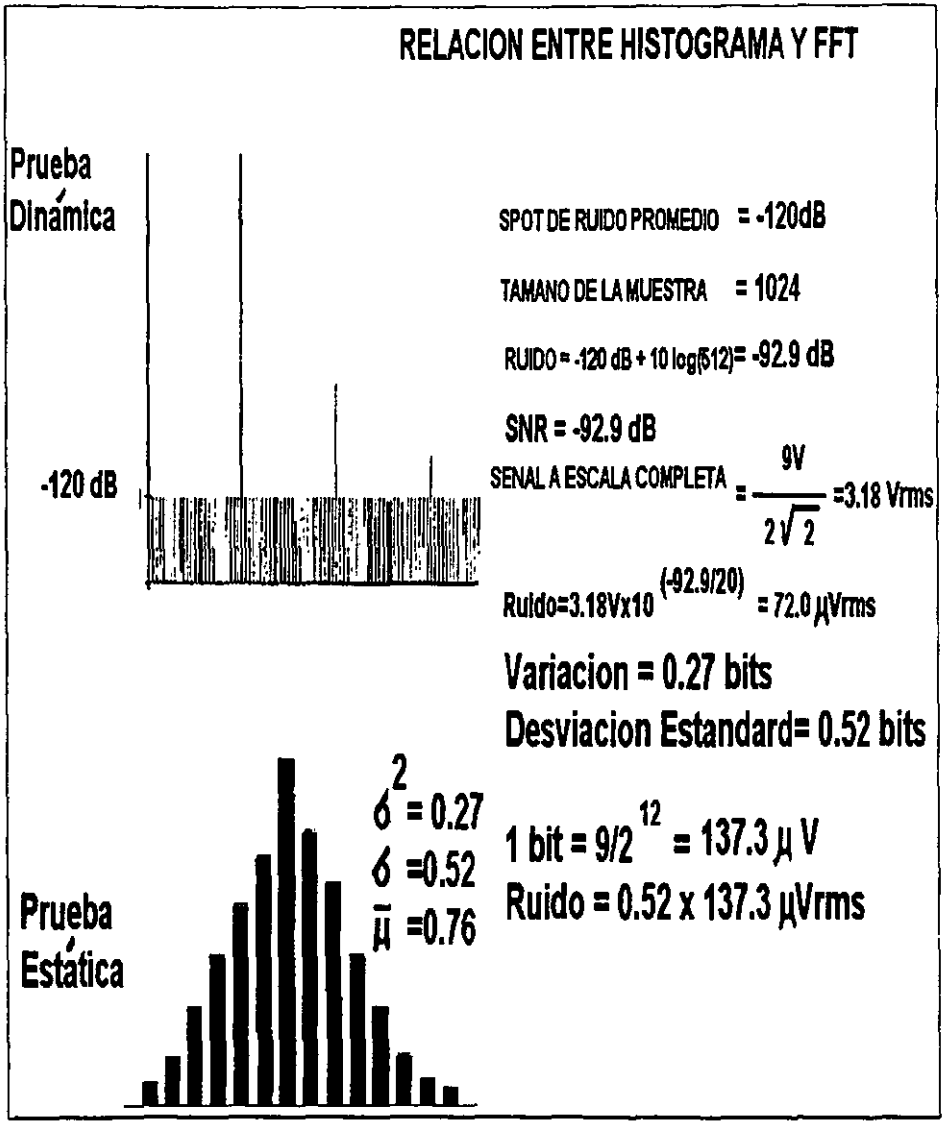

RELACION ENTRE HISTOGRAMA Y FFT
Prueba Dinámica
-120 dB
SPOT DE RUIDO PROMEDIO = -120dB
TAMANO DE LA MUESTRA = 1024
RUIDO = -120 dB + 10 log(512)= -92.9 dB
SNR = -92.9 dB
SENAL A ESCALA COMPLETA = 9V / 2√2 =3.18 Vrms
Ruido=3.18Vx10 (-92.9/20) = 72.0 µVrms
Variacion = 0.27 bits
Desviacion Estandard= 0.52 bits
σ2 = 0.27
σ =0.52
μ̄ =0.76
1 bit = 9/2 12 = 137.3 µ V
Ruido = 0.52 x 137.3 µVrms
Prueba Estática

Fig.3.32.

El ruido es 92.9 dB menor a 3.18 volts o 72 uV *rms*. La desviación de un histograma es de 0.27 cuentas las cuales se transladan a la desviación estándar de 0.52 cuentas. El tamaño del *bin* esta determinado por el tramo de entrada y la resolución del ADC con 9V de tramo de entrada, un bit es equivalente a 137,3 uV. Del histograma, el ruido es estimado para ser 0.52 cuentas o 71.4 uV *rms*. Debido a esto el ruido *rms* calculado de la función de densidad de potencia es de 72 uV utilizando el histograma.

CAPITULO IV

IV ESQUEMAS DE CONVERSIÓN UTILIZADOS EN CONVERTIDORES ANALÓGICOS DIGITALES Y PARAMETROS DE SELECCION

Con la reciente proliferación de los Convertidores Analógicos Digital (CAD´s), los Convertidores Digital Analógico (CDA´s), y la variación en los esquemas de código que utilizan estos convertidores, se tiene la necesidad de analizar estos esquemas de código que son utilizados para hablar con el "Mundo Digital". En este capítulo haremos referencia principalmente a los Convertidores de BURR-BROWN.

Siguiendo este capítulo se hará una lista de abreviaciones y definiciones que intentarán clarificar cualquier pregunta acerca de la nomenclatura que es utilizada.

En este capítulo, ejemplos y tablas serán utilizadas como si fuese un convertidor de datos de 4 bits. En ambos ejemplos (unipolares y bipolares) el Full Scale Range (*FSR*) es de 10V creando un V_{LSB} de 0.625V. Para ejemplos

unipolares, menos Full Scale (*-FS*) es 0V y más Full Scale (*+FS*) es 10V; para ejemplos bipolares, *-FS* es -5V y *+FS* es 5V.

IV.1.0. UBD - Binario Unipolar Directo

USB - Unipolar Straight Binary

La codificación BUD - binaria unipolar directa *USB - Unipolar Straight Binary*, es el esquema más simple de codificación para entender. Como el nombre lo dice este es un esquema de código el cual es utilizado por voltajes unipolares.

Cuando se usa código USB, las cuentas empiezan todas en cero (0000) para un V_{CODE} de 0V (V_{t+} = 0V +1/2V_{LSB} y no hay V_{t-}). Como el código digital se incrementa, el voltaje analógico se incrementa (un V_{LSB}) a la vez, y la cuenta digital termina en (1111) a un valor positivo de Full Scale. La tabla I muestra como los códigos USB corresponden a los voltajes analógicos para un sistema digital de 4-bit. Tabla 4.1.

MNEMONICO	CODIGO DIGITAL	V t-	VCODE	Vt+
Cero	´0000		0.000	0.3125
´+1 VLSB	´0001	0.3125	0.625	0.9375
	´0010	0.9375	1.250	1.5625
	´0011	1.5625	1.875	2.1875
1/4 FSR	´0100	2.1875	2.500	2.8125
	´0101	2.8125	3.125	3.4375
	´0110	3.4375	3.750	4.0625
	´0111	4.6875	4.375	4.6875
1/2 FSR	´1000	4.6875	5.000	5.3125
	´1001	5.3125	5.625	5.9375
	´1010	5.9375	6.250	6.5625
	´1011	6.5625	6.875	7.1875
3/4 FSR	´1100	7.1875	7.500	7.8125
	´1101	7.8125	8.125	8.4375
	´1110	8.4375	8.750	9.0625
´+FS	´1111	9.0625	9.375	

Tabla. 4.1.

La codificación PBD - unipolar binario directo es utilizada por muchos convertidores; por ejemplo los convertidores ADC7802 y ADC7803 de Burr Brown.

IV.2.0. BCD - Binario Complemento Directo

CSB - Complementary Straight Binary

El esquema de codificación BCD binario complemento directo es exactamente el complemento digital opuesto (complemento a unos) del UBD - *USB*. El código CSB, como el código USB, es también restringido para sistemas unipolares.

Cuando se usa código CSB con un sistema digital, el sistema digital empieza en ceros en el valor positivo del Full Scale. Como se va incrementando el código digital, el voltaje analógico se decrementa un VLSB a la vez, hasta que se alcanza el voltaje 0V es alcanzado en el código digital 1111. La relación entre el código CSB y sus voltajes analógicos correspondientes los podemos observar en la Tabla 4.2.

MNEMONICO	CODIGO DIGITAL	V t-	VCODE	Vt+
Cero			0.000	0.3125
´+1 VLSB	1111	0.3125	0.625	0.9375
	1110	0.9375	1.250	1.5625
	1101	1.5625	1.875	2.1875
1/4 FSR	1011	2.1875	2.500	2.8125
	1010	2.8125	3.125	3.4375
	1001	3.4375	3.750	4.0625
	1000	4.6875	4.375	4.6875
1/2 FSR	´0111	4.6875	5.000	5.3125
	´0110	5.3125	5.625	5.9375
	´0101	5.9375	6.250	6.5625
	´1000	6.5625	6.875	7.1875
3/4 FSR	´0011	7.1875	7.500	7.8125
	´0010	7.8125	8.125	8.4375
	´0001	8.4375	8.750	9.0625
´+FS	´0000	9.0625	9.375	

Tabla. 4.2.

IV.3.0. BBO - Binario Bipolar con Offset

BOB - Bipolar Offset Binary

La codificación BBO - Binario Bipolar con Offset, como el nombre lo indica, es para ser utilizado en sistemas bipolares (donde el voltaje analógico puede ser positivo o negativo). Este sistema es muy similar al esquema USB, las cuentas digitales se incrementan al aumentar el valor analógico.

El esquema BOB empieza con un cero digital (0000) en el full scale negativo (**-FS**). De tal forma que al ir aumentando la cuenta digital, el valor analógico correspondiente se aproximará al **+FS** en pasos con valor V_{LSB}, pasando en el camino a través del cero bipolar. Esta es la zona "zero crossing" y ocurre en el código digital 1000. Las cuentas digitales continua aumentando proporcionalmente respecto a la entrada hasta alcanzar el valor +FS en la cuenta digital (1111). Tabla 4.3.

MNEMONICO	CODIGO DIGITAL	V t-	VCODE	Vt+
´-FS	´0000		-5.000	-4.6875
	´0001	-406875	-4.375	-4.0625
	´0010	-4.0625	-3.750	-3.4375
	´0011	-3.4375	-3.125	-2.1875
1/2 -FS	´0100	-2.1875	-2.500	-2.8125
	´0101	-2.1875	-1.875	-1.5625
	´0110	-1.5625	-1.250	-0.9375
BPZ-1VLSB	´0111	-0.9375	-0.625	-0.3125
BPZ	´1000	-0.3125	0.000	0.3125
BPZ+1VLSB	´1001	0.3125	0.625	0.9375
	´1010	0.9375	1.250	1.5625
	´1011	1.5625	1.875	2.1875
1/2 +FS	´1100	2.1875	2.500	2.8125
	´1101	2.8125	3.125	3.4375
	´1110	3.4375	3.750	4.0625
´+FS	´1111	4.0625	4.375	

Tabla. 4.3.

Con la codificación *BOB*, el *MSB* puede ser considerado un indicador de signo donde un cero lógico indica un valor analógico negativo, y un uno indica un valor analógico mayor o igual a *BPZ*.

Cuando el *BPZ* muestra la transición de V_{t+} a *BZP* de un valor negativo (0111 a 1000) actualmente ocurre a - 0.3125V causando que el *MSB* vaya a positivo en un valor negativo.

El convertidor ADS7800 de Burr Brown, 12 bits, muestreo a 333khz, utiliza una decodificación de salida *BOB* para implementar sus rangos de entrada de +-5V a +-10V. Las series DAC780x de Burr Brown también utiliza este tipo de esquema de codificación en cada una de sus tres diferentes formatos de interfase (serial, 8-bits + 4 bits paralelo, 12 bits paralelo).

IV.4.0. COB - Binario Complementario con Offset

COB - Complementary Offset Binary

Codificación complemento del offset binario, como el esquema de codificación *BOB*, este es utilizado para sistemas donde la señal analógica es bipolar. La relación existente entre *COB y BOB* es que cada cuenta codificada es el complemento a unos del otro (todos los bits son invertidos por su complemento de uno a otro).

La codificación *COB* empieza en el cero digital (0000) en *+FS*. Cada vez que se incrementa la cuenta, el valor analógico correspondiente se aproximará al *-FS* con pasos de valor V_{LSB} pasando a través del cero bipolar durante el camino. Este cruce por cero ocurre en el código digital 0111. Figura. 4.4.

MNEMONICO	CODIGO DIGITAL	V t-	VCODE	Vt+
´-FS	1111		-5.000	-4.6875
	1110	-406875	-4.375	-4.0625
	1101	-4.0625	-3.750	-3.4375
	1100	-3.4375	-3.125	-2.1875
1/2 -FS	1011	-2.1875	-2.500	-2.8125
	1010	-2.1875	-1.875	-1.5625
	1001	-1.5625	-1.250	-0.9375
BPZ-1VLSB	1000	-0.9375	-0.625	-0.3125
BPZ	´0111	-0.3125	0.000	0.3125
BPZ+1VLSB	´0110	0.3125	0.625	0.9375
	´0101	0.9375	1.250	1.5625
	´1000	1.5625	1.875	2.1875
1/2 +FS	´0011	2.1875	2.500	2.8125
	´0010	2.8125	3.125	3.4375
	´0001	3.4375	3.750	4.0625
´+FS	´0000	4.0625	4.375	

Tabla. 4.4.

Con la codificación *COB*, así como con la *BOB*, el *MSB* puede ser utilizado como indicador de signo en cualquier lugar donde haya un "1" lógico indica un valor analógico negativo, y un "0" lógico indica un valor analógico mayor o igual al *BPZ*.

Cuando el *BPZ* muestra la transición V_{t+} a *BPZ* de un valor negativo (1000 al 0111) de hecho ocurre al voltaje -0.3125V causando que el *MSB* vaya a positivo en un valor negativo.

IV.5.0. CDB - Complemento A Dos Binario

BTC - Binary Two´s Complement

La codificación complemento binario a dos es un tipo de codificación utilizada por la mayoría de los sistemas basados en microprocesadores y procesadores matemáticos para algoritmos matemáticos, y también es el esquema

de codificación en el cual la industria del audio digital ha decidido utilizarlo como estándar.

La codificación complemento binario a dos es también un esquema diseñado para señales bipolares analógicas. Es muy similar a *BOB*, pero no luce como este. La única diferencia entre *BOB* y *BTC* es que el *MSB* ha sido invertido.

Desafortunadamente *BTC* no es tan directo como los esquemas anteriormente mencionados. Los códigos no son continuos de un final del espectro analógico a otro debido a la discontinuidad la cual ocurre a *BPZ*.

El cero digital (0000) corresponde al *BPZ*, y la cuenta digital se incrementa a su máximo código positivo de 0111 tal como el valor del voltaje se aproxime o alcance el valor *+FS*. El código después se reanuda hacia el valor *-FS* en el código digital 1000, y después se aproxima a *BPZ* hasta el valor digital 1111 sea alcanzado en un valor *LSB* por debajo del *BPZ*. Fig.4.5.

MNEMONICO	CODIGO DIGITAL	V t-	VCODE	Vt+
´-FS	1000		-5.000	-4.6875
	1001	-406875	-4.375	-4.0625
	1010	-4.0625	-3.750	-3.4375
	1011	-3.4375	-3.125	-2.1875
1/2 -FS	1100	-2.1875	-2.500	-2.8125
	1101	-2.1875	-1.875	-1.5625
	1110	-1.5625	-1.250	-0.9375
BPZ-1VLSB	1111	-0.9375	-0.625	-0.3125
BPZ	´0000	-0.3125	0.000	0.3125
BPZ+1VLSB	´0001	0.3125	0.625	0.9375
	´0010	0.9375	1.250	1.5625
	´0011	1.5625	1.875	2.1875
1/2 +FS	´0100	2.1875	2.500	2.8125
	´0101	2.8125	3.125	3.4375
	´0110	3.4375	3.750	4.0625
´+FS	´0111	4.0625	4.375	

Tabla 4.5.

Con el esquema *BTC*, el *MSB* puede considerarse nuevamente como un indicador de signo. Cuando el *MSB* es un "0" lógico un valor positivo es indicado, y cuando el *MSB* es un "1" lógico un valor negativo es indicado.

La transición *Vt+* al *BPZ* de un valor negativo (1111 a 0000) de hecho ocurre a -0.3125V causando que el *MSB* vaya a positivo en un valor negativo.

Este es el esquema del código el cual es utilizado por Burr-Brown en los chips de interfase para *DSP* (DSP101/DSP102 entrada analógica y DSP201/DSP202 salida analógica) diseñada para una "interfase con cero chips" compatibles con la mayoría de *DSP* existentes actualmente en el mercado (1999).

El esquema complemento binario a dos es uno de los más utilizados en los convertidores analógicos digitales de alta velocidad, como lo son el ADC603, el ADC 614, y por supuesto es utilizado por los convertidores digitales de audio de Burr-Brown.

IV.6.0. CTC - Complemento a Dos del Complemento

CDC – Complementary Two´s Complement

La codificación complemento del complemento a dos es diseñada también para un sistema bipolar de señales analógicas. Es el complemento de su contra parte *BTC*, y es también muy similar al *COB*, aunque esta relación no es inmediatamente lógica. La única diferencia entre el *COB* y *CTC* es que el *MSB* ha sido invertido.

Con la codificación, digital "cero" esta en un voltaje analógico el cual es ligeramente menor a 1 *LSB* que cero bipolar analógico. Así como las cuentas digitales se incrementan, los voltajes analógicos se van convirtiendo más negativos hasta están en alto excepto por el *MSB* (0111). En este punto, el código digital corresponde al valor analógico *-FS*. EL próximo paso en incrementar el código digital tendrá que tener el código digital "1", y el resto de los bits como un"0´s" lógicos (1000), y este código representa el valor analógico positivo *+FS*. Así como el código digital se incrementa, el valor del voltaje analógico correspondiente decrece hasta que se obtenga el *BPZ*. La tabla 4.6. demuestra la relación analógica/digital.

MNEMONICO	CODIGO DIGITAL	V t-	VCODE	Vt+
´-FS	1000		-5.000	-4.6875
	1001	-406875	-4.375	-4.0625
	1010	-4.0625	-3.750	-3.4375
	1011	-3.4375	-3.125	-2.1875
1/2 -FS	1100	-2.1875	-2.500	-2.8125
	1101	-2.1875	-1.875	-1.5625
	1110	-1.5625	-1.250	-0.9375
BPZ-1VLSB	1111	-0.9375	-0.625	-0.3125
BPZ	´0000	-0.3125	0.000	0.3125
BPZ+1VLSB	´0001	0.3125	0.625	0.9375
	´0010	0.9375	1.250	1.5625
	´0011	1.5625	1.875	2.1875
1/2 +FS	´0100	2.1875	2.500	2.8125
	´0101	2.8125	3.125	3.4375
	´0110	3.4375	3.750	4.0625
´+FS	´0111	4.0625	4.375	

Tabla. 4.6.

Con la codificación complementaria del complemento a dos, el *MSB* es también un indicador de signo con los estados "0" y "1" representan los valores negativos y positivos del voltaje, respectivamente.

Este código es utilizado también por los convertidores de alta velocidad de Burr-Brown ADC603 y ADC614. Estos convertidores acompañan esta tarea dual de codificación configurado con una entrada para seleccionar el código.

Los códigos de entrada y salida utilizados con *ADC´s y DAC´s* son variados un convertidor individual puede ser capaz de utilizar una o más esquemas de codificación. Sin embargo, con todos estos esquemas viables, el esquema deseado no es siempre viablemente de lectura para una conversión particular de interés. La conversión de códigos es sencilla, al igual que es de fácil el manipular un sistema digital de uno a otro (lo único que necesitas es un inversor) también lo es el manipular la polaridad de un voltaje monopolar a un bipolar (utilizando componentes analógicos).

IV.8.0 Definiciones del Capítulo

.n = Número de bits es un sistema digital particular.

LSB = Least Significant Bit - Bit Menos Significativo.

MSB = Most Significant Bit - Bit menos Significativo.

FSR = Full Scale Range - Rango a Escala Completa.

Rango Dinámico de una señal analógica.

BPZ = Bipolar Zero - Cero Bipolar.

V_{LSB} = Least Significant Bit Voltage - Voltaje del Bit Menos Significativo.

V_{LSB} = FSR/2^n

V_{CODE} = Code Voltage - Código de Voltaje

V_{CODE} = Código Digital en Base 10 * V_{LSB}

El V_{CODE} es el voltaje correspondiente a un código digital en un convertidor analógico digital. Para un convertidor analógico digital que provee de una corriente de salida como modo de operación, *VCODE* se refiere al voltaje después de haber realizado la conversión de corriente a voltaje.

Para un convertidor Analógico Digital, el código de voltaje es un rango de voltajes analógicos compuestos por V_{CODE} +- 1/2V_{LSB} Esto es debido al inherente

error de cuantización de *+- 1/2 VLSB* que se encuentra presente en la salida digital finita de un Convertidor Analógico Digital.

+FS = Positive Full Scale - Escala Completa positiva.

-FS = Negative Full Scale - Escala Completa Negativa.

Vt = Transition Voltage - Voltaje de Transición.

El voltaje de transición que corresponde al cambio actual del código en un convertidor ideal analógico digital. Estos voltajes son los voltajes en cada final del rango de *VCODE +- 1/2 VLSB.*

Vt+ = *VCODE + 1/2 VLSB*

Vt- = *VCODE - 1/2 VLSB*

Para un convertidor Digital - Analógico, será exactamente *VCODE*, y la transición del voltaje puede ser ignorado.

IV.9.0. Parámetros de Selección

Al pretender utilizar un CAD se debe de analizar un par de criterios que el tesista recomienda continuación. Teniendo con fin el saber que tipo de arquitectura se debe de utilizar y que características principales deberá de tener el CAD.

1er. Criterio. Se tiene que determinar la clasificación del diseño en la aplicación.

1. Medición
   - Típico para frecuencia baja, alta resolución.
   - Utilizado para medir cambios en entidades físicas
   - Muchos de estos diseños emplean la arquitectura delta-sigma o soluciones *SAR*

2. Video
   - Típicamente CAD de 8 bits y velocidad de 7200 MHZ

- Principalmente utiliza convertidores *Flash y Pipeline*

3. Audio

- Utilizado para grabadoras de voz y sonido para audio digital
- Ancho de Banda usualmente limitado a los 20khz.
- Se utiliza Principalmente convertidores delta-sigma
- Principales fabricantes: Phillips, Crystal, ADI, Burr Brown

4. Alta velocidad

- Generalmente son utilizados para alta resolución a una velocidad > 1Mhz.
- La arquitectura más popular es la *SAR y PIPELINE*
- Aplicaciones principales son radares, comunicaciones y procesamiento de imágenes
- Principales fabricantes: ADI, Harris, Phillips, SPI, Burr Brown, Datel,.

5. Propósitos Generales

- Para velocidades en 10 Khz a 1Mhz, 10 bits a 16 bits
- Burr Brown es el líder en este mercado
- Otros fabricantes: ADI, Crystal, Maxim, LTC, National, Micropower
- Burr Brown es el líder en 16 bits

(a) Adquisición de Datos

- Control de Datos
- Control de Procesos Industriales
- *Data logging*
- Medición y Prueba
- Medición de Fenómenos Físicos

(b) Instrumentación Médica

- Procesamiento de Imágenes
- Telemetría, Monitoreo Remoto
- Digitalización a través de Fotodiodos

(c) Comunicación

- Estación de Bases para Telecom
- Recepción Digital

2do. Criterio. Se deben de definir los siguientes parámetros al utilizar un Convertidor Analógico Digital.

- Resolución requerida
- Números de Canales
- Rango de Muestreo
- Formato de la Salida (Serial, Paralela o ambas)
- Rango de entrada (Sí es Unipolar y Bipolar)
- Requerimientos de polarización
- Necesidad de alimentación (Portátil, Batería; Fuente bipolar, etc.)
- Tipo de encapsulado - *package number*
- Otras características importantes del sistema dependiendo de la aplicación.

CAPITULO V

V ALGORITMO DE LA TRANSFORMADA RAPIDA DE FOURIER TRF – *FAST FOURIER TRANSFORM FFT*

La discretización FFT es un algoritmo el cual convierte una función de muestreo de valor complejo de tiempo en una función de muestreo de valor complejo de frecuencia.

La mayoría de las veces, nosotros queremos operar en funciones de valor real, así que nosotros configuramos todas las partes imaginarias de la entrada a cero.

- Todos los arreglos de entrada y salida deberán deben de tener un tamaño común, el cual podemos llamar ***n***.
- El valor de ***n*** deberá ser un entero positivo de ***2***. Por ejemplo, un arreglo de ***1024*** es permitido, pero un arreglo de ***1000*** no es. El arreglo de menor tamaño es ***2***. No hay limite superior para el valor ***n*** mayor que el inherente limitado por el tamaño de la memoria de un arreglo de cuatro

(entrada ***{real, imaginario}, salida{real, imaginario}***) y el limite en el tiempo es inherente en ejecutar este algoritmo ***O (n * log(n)***.

- El arreglo `realOut` sostiene los coeficientes de las ondas del coseno en la fórmula de Fourier.
- El arreglo `imagOut` sostiene los coeficientes de las ondas del seno en la fórmula de Fourier.
- El orden de las frecuencias en los arreglos de las salidas (`realOut` y `imagOut`) son un poco extraños, por que ellas contienen ambas frecuencias negativas y positivas. Ambas frecuencias son necesarias por que son utilizas en el cálculo matemático para trabajar cuando las entradas son de valores complejos (cuando al menos alguno de las entradas no tiene ningún componente imaginario diferente de cero). La mayoría de las veces, el FFT es utilizada para entradas estrictamente de valores reales, y este es el caso especial del análisis digital audio. El FFT, cuando se alimentan valores reales de entrada, da una salida de las cuales las frecuencias positivas y negativas son redundantes. Este se reinvierte cuando ellos son *complejos conjugados* de uno otro, significando que la parte real son iguales y la parte imaginaria son opuesta una de la otro. Si las entradas son valores reales, podemos conseguir toda la información de la frecuencia que necesites solo mirando la primer parte del arreglo.

- El primer bloque de salida, **`realOut[0]` y `imag[0]`**, contiene el valor promedio de todas las muestras de entrada. Para los índices de salida *i=1,2,3,....,n/2*, el valor de la frecuencia expresada en Hz es ***f = sampling Rate - rango de frecuencia * i/n.*** La contra parte negativa de la frecuencia de cada índice de frecuencia positiva *i=1,2,3...n/2-1, es i'=n-i.* Aquí hay un ejemplo. Supón que el rango de frecuencia es de *44,100Hz,* y se esta utilizando buffers de *1024* números complejos para ambas entradas y salidas. La frecuencia a *i=1* será *(44,100Hz)*1/1024 = 43.07 Hz.* La contraparte de la frecuencia negativa será a *i=1024-1 = 1023 (el ultimo bloque en el arreglo),* con una frecuencia de *43.07 Hz.* De esta forma, *i=17* corresponderá a una frecuencia de *(43.07) * 17 = 732.13 Hz,* mientras *i=1024 –17* corresponderá a *–732 Hz,* etc.
- El índice *n/2* será un caso especial. Este corresponderá a la frecuencia de Nyquist, la cual es siempre es la mitad de la frecuencia de muestreo en cualquier grabación digital *PCM.* Por ejemplo, *la frecuencia de Nyquist* en una grabación de audio de *CD* es *(44,100 Hz)/2 = 22,050 Hz.* La frecuencia de Nyquist es importante por que esta es su frecuencia más alta que una grabación digital *PCM* puede reproducir. Nada por arriba puede ser representado por el *PCM.* Si tu tratas de hacer un muestreo a

una señal la cual tiene una frecuencia mayor que la frecuencia de Nyquist, ocurrirá una distorsión severa llamada *aliasing*, Los ingenieros de grabación saben que se tienen que filtrar las salidas de cualquier señal que esta por arriba de la frecuencia de Nyquist en el equipo análogo antes de que la señal se hecha un muestreo digitalmente, para así eliminar los problemas de *aliasing*. Nota que la limitación de la limitación en la frecuencia de Nyquist es inherente de la grabación digital misma, de cualquier forma la FFT siempre es realizada en la grabación.

- Si la entrada del FFT es real, el índice de la frecuencia de Nyquist *n/2* en la salida tendrá siempre un valor real (significa que la parte imaginaria será cero, o algo realmente cerca a cero debido a los errores de redondeo por punto flotante). Esto es porque, utilizando la formula abajo mostrada, *i=n-n/2=n/2*, así que la frecuencia de Nyquist es su contraparte negativa. Sin embargo, esta deberá ser igual a su complejo conjugado, la cual cambiara lo fuerza a un numero real.
- Algunas veces se estará interesado solo en la magnitud de cada componente de la frecuencia, o su ángulo de fase. Aquí esta como se puede calcular:

```
#include <math.h>
double magnitude  = sqrt ( realOut[I]*realOut[I] +
imangOut[I]*imagOut[I]);
```

```
double angle = atan2 (imagOut[I], realOut[I]);
```

Si se esta interesado en hacer una conversión a la inversa, de una magnitud, ángulo real e imaginario, utiliza el siguiente código:

```
#include <math.h>

double real = magnitude *  cos(angle);
double imag = magnitude  *  sin(angle);
```

- Si se necesita tener un riguroso entendimiento matemático del FFT, aquí hay una ecuación la cual te dirá la relación exacta entre las entradas y las salidas. En esta ecuación, x_k es el k-*ésima* de entrada del valor complejo muestreado en el dominio del tiempo, y_p es el p-*ésima* de la salida del valor complejo muestreado en el dominio de la frecuencia, y $n=2^N$ es el numero total de muestras. Nota que la k y la p se encuentran en el rango de *0...n-1.*

$$y_p = \sum_{k=0}^{n-1} x_k \left[\cos\left(2\Pi \frac{kp}{n} \right) + isen\left(2\Pi \frac{kp}{n} \right) \right]$$

- A pesar de que esta fórmula te dice a que es equivalente la *FFT*, esta fórmula no es exactamente como el algoritmo *FFT* es implementado. Esta fórmula requiere $O(n^2)$ operaciones, de cualquier manera el *FFT* mismo es $O\ (n*log2(n))$. En otras palabras, si utilizas la fórmula anteriormente mencionada, será mas lento que al utilizar que al utilizar el algoritmo del *FFT*. Sin embargo, sí tu sólo necesitas un *set* pequeño del espectro de frecuencias (diremos que solo dos o tres muestras de la frecuencia), o se tiene un número de muestras que no son potencia de **2**, esta fórmula combinada con algunas parámetros trigonométricas pueden ser de uso práctico.

V.1.0. Una aplicacion de la FFT

El algoritmo del *FFT* intenta ser mejor para analizar las grabaciones de audio digital, que para el filtrado o la sintetización de sonidos. Un ejemplo común es cuando se quiere hacer software equivalente para un aparato llamado analizador de espectros, el cual los ingenieros eléctricos utilizan para desplegar graficas de frecuencia de una señal eléctrica. Se utiliza también para aplicaciones que no tienen nada que ver con el audio, como el procesamiento de imágenes (utilizando la versión bidimensional de la *FFT*). La *FFT* tiene aplicaciones científicas, como tratando de detectar fluctuaciones periódicas en los precios de la Casas de Bolsa, poblaciones de animales, etc. Las *FFT's* son también utilizadas en los análisis seísmo gráficos, en la información para sonogramas del interior de la tierra. También se están utilizando para analizar secuencias del *ADN*.

El problema principal al utilizar el *FFT* para el procesamiento de sonidos es que las grabaciones digitales deben de estar rotas en pedazos de n muestras, donde *n* siempre tiene que ser un entero potencia de *2*. Uno, primero se tomara un bloque de *FFT*, procesara el arreglo de *FFT* de salida (la salida de ceros todas las frecuencias de muestras fuera de un cierto rango de frecuencias), después se realiza la inversa del *FFT* para conseguir una señal de regreso filtrada en el dominio del tiempo. Cuando el audio esta roto dentro de

discontinuidades como estas y procesados con el FFT, el resultado filtrado tendrá discontinuidades las cuales causaran un sonido *clicking* en la salida de cada frontera de la discontinuidad. Por ejemplo, si la grabación tiene un rango de muestreo de 44,100 Hz, y los bloques tienen un tamaño *n=1024,* habrá entonces un *click* audible cada *1024/(44,100 Hz) = 0.0232* segundos, por ser este tiempo de *click* tan pequeño esta casi no se nota.

Uno de los problemas más graves que se tiene es el cómo manejar las discontinuidades, se puede utilizar el siguiente método de la sombra de buffers con registros de corrimiento o con filtros digitales. Brevemente se pueden describir los dos.

Sombra de *buffers* con registros de corrimiento, Asumiendo que el tamaño del buffer son $n=2^N$. Sobre la primera iteración, se lee n muestras de la entrada de audio, haciendo la *FFT,* procesando, e integrando las *FFT,* guardando el dato resultante del tiempo en la segunda mitad del buffer. Después, desplaza la segunda mitad del buffer original a la primera mitad (recordando que *n* es una potencia de *2,* si esta es divisible por *2*), y lee las *n/2* muestras dentro de la segunda mitad del buffer. Nuevamente haz la *FFT,* procésala, obtén la integración de las *FFT.* Ahora haz una sombra lineal de la segunda mitad del buffer viejo que fue salvado del primer (*FFT,* proceso, Integración de las *FFTs*) triplícalo multiplicando cada muestra por un valor que varía de *1* (para una

muestra con número *n/2*) a 0 (para un número de muestras *n-1*). Haz una sombra lineal en la primera mitad del nuevo buffer de salida (ir linealmente de *0 a 1*), y suma dos mitades juntas para conseguir la salida lo cual será una transición suave. Nota que áreas alrededor de cada discontinuidad son virtualmente borradas de la salida, aunque un nivel consistente de volumen es mantenido. Esta técnica es la mejor cuando el proceso no altera la fase de información del espectro de frecuencia. Un filtro pasabandas trabajará muy bien, pero podrías encontrar distorsión con el registro de corrimiento de la marca de la señal. Si tu realmente quieres limpiar el sonido producido por estas discontinuidades en el audio deberás revisar los estudios sobre los filtros digitales lineales en el dominio del tiempo – *Time Domain Filters*, los cuales procesan las muestras de la entrada de audio en muestras una a la vez en lugar de procesarlas en un paquete de muestras.

A continuación se muestra un ejemplo de C++ del presente método. Se debe tener cuidado especial como son llamadas las funciones *FadeMix y ShiftData* desde *main*.

Si tu realmente quieres hacer limpieza al hacer aplicaciones de sonido, debes de utilizar los algoritmo de filtros digitales, entonces tienes que estudiar Time Domain Filtres –Filtros en el Dominio de Tiempo.

V.2.0. Paquetes para resolver las Transformada Rápida de Fourier FFT´s de funciones seno y coseno

Antes que todo haremos notar que el simple hecho de hacer un programa para resolvar Transformadas Rapidas de Fourier es motivo para otra tesis. Las Versiones Simples no usan área de trabajo, pero las Versiones Rápidas utilizan áreas de trabajo. Las Versiones Rápidas tienen la mismas especificaciones.

Comentarios de uso :

Bloques de cometarios de explicación en cada Programa.

Los ejemplos de las subrutinas tienen programas ejemplos que llama a las subrutinas para demostrar su implementación.

La tabla de coseno/seno no son necesarias si tu utilizas la "fft2f.*".

Archivos:

En la página "FFT Links" del servidor de Yahoo podrás estudiar la página de Don Cross – dcross@intersrv.com. Don te presentará todo un estudio acerca de la transformada rápida de Fourier.

fft2f.c : FFT Paquete en C - Versión simple (radix 2)

fft2ft.c : Test Program for "fft2f.c"

fft4g.c : FFT Package in C - Versión Rápida (radix 4, 2)

fft8g.c : FFT Package in C - Versión Rápida (radix 8, 4, 2)

fftxgt.c : Test Program for "fft*g.c"

fftxx.doc : Documento de la "fft*.*"

pi_fft.c : PI(= 3.14159265358979323846...) Programa para calcular un banco de prueba para las librerias "fft4g.c", "fft8g.c"

Ejemplo de la compilacion:

```
gcc -O6 -ffast-math pi_fft.c fft4g.c -lm -o pi_fft4g
cc -fast -dalign -xO5 pi_fft.c fft8g.c -lm -o pi_fft8g
cl /O2 /G6 pi_fft.c fft4g.c /Fepi_fft4g.exe
etc...
```

Autor:

-------- Referencia --------

Masatake MORI, Makoto NATORI, Tatuo TORII: Suchikeisan,

Iwanamikouzajyouhoukagaku18, Iwanami, 1982 (Japones)

Henri J. Nussbaumer: Fast Fourier Transform and Convolution

Algorithms, Springer Verlag, 1982

-------- Copyright --------

Takuya OOURA (email: ooura@mmm.t.u-tokyo.ac.jp)

-------- Historial --------

...

Dec. 1995 : Edición de la FFT de propósitos generales

Mar. 1996 : Cambio de especificación

Jun. 1996 : Cambio del método trigonométrico por una tabla de función

Sep. 1996 : Modificación de los documentos

Feb. 1997 : Cambio en los bucles de mariposa

Dec. 1997 : modificación de los documentos

Dec. 1997 : Aumento del programa "fft4g.*"

Jul. 1998 : Arreglo de algunos bugs en los documentos

Jul. 1998 : Aumento del "fft8g.*" y borra el "fft4f.*"

Jul. 1998 : Aumenta el programa de prueba "pi_fft.c"

V.3.0. Programa fft2f.c - Paquete FFT en C - Versión simple (radio 2)

```
/*
Fast Fourier/Cosine/Sine Transform
    dimension   :one
    data length :power of 2
    decimation  :frequency
    radix       :2
    data        :inplace
    table       :not use
functions
    cdft: Complex Discrete Fourier Transform
    rdft: Real Discrete Fourier Transform
    ddct: Discrete Cosine Transform
    ddst: Discrete Sine Transform
    dfct: Cosine Transform of RDFT (Real Symmetric DFT)
    dfst: Sine Transform of RDFT (Real Anti-symmetric DFT)
function prototypes
    void cdft(int, double, double, double *);
    void rdft(int, double, double, double *);
    void ddct(int, double, double, double *);
    void ddst(int, double, double, double *);
    void dfct(int, double, double, double *);
    void dfst(int, double, double, double *);


-------- Complex DFT (Discrete Fourier Transform) --------
    [definition]
        <case1>
            X[k] = sum_j=0^n-1 x[j]*exp(2*pi*i*j*k/n), 0<=k<n
        <case2>
            X[k] = sum_j=0^n-1 x[j]*exp(-2*pi*i*j*k/n), 0<=k<n
        (notes: sum_j=0^n-1 is a summation from j=0 to n-1)
    [usage]
        <case1>
            cdft(2*n, cos(M_PI/n), sin(M_PI/n), a);
        <case2>
            cdft(2*n, cos(M_PI/n), -sin(M_PI/n), a);
    [parameters]
        2*n             :data length (int)
                         n >= 1, n = power of 2
        a[0...2*n-1]    :input/output data (double *)
                         input data
                             a[2*j] = Re(x[j]), a[2*j+1] = Im(x[j]), 0<=j<n
                         output data
                             a[2*k] = Re(X[k]), a[2*k+1] = Im(X[k]), 0<=k<n
    [remark]
        Inverse of
            cdft(2*n, cos(M_PI/n), -sin(M_PI/n), a);
        is
            cdft(2*n, cos(M_PI/n), sin(M_PI/n), a);
            for (j = 0; j <= 2 * n - 1; j++) {
                a[j] *= 1.0 / n;
            }
        .
```

```
-------- Real DFT / Inverse of Real DFT --------
    [definition]
        <case1> RDFT
            R[k] = sum_j=0^n-1 a[j]*cos(2*pi*j*k/n), 0<=k<=n/2
            I[k] = sum_j=0^n-1 a[j]*sin(2*pi*j*k/n), 0<k<n/2
        <case2> IRDFT (excluding scale)
            a[k] = R[0]/2 + R[n/2]/2 +
                   sum_j=1^n/2-1 R[j]*cos(2*pi*j*k/n) +
                   sum_j=1^n/2-1 I[j]*sin(2*pi*j*k/n), 0<=k<n
    [usage]
        <case1>
            rdft(n, cos(M_PI/n), sin(M_PI/n), a);
        <case2>
            rdft(n, cos(M_PI/n), -sin(M_PI/n), a);
    [parameters]
        n             :data length (int)
                       n >= 2, n = power of 2
        a[0...n-1]    :input/output data (double *)
                       <case1>
                           output data
                               a[2*k] = R[k], 0<=k<n/2
                               a[2*k+1] = I[k], 0<k<n/2
                               a[1] = R[n/2]
                       <case2>
                           input data
                               a[2*j] = R[j], 0<=j<n/2
                               a[2*j+1] = I[j], 0<j<n/2
                               a[1] = R[n/2]
    [remark]
        Inverse of
            rdft(n, cos(M_PI/n), sin(M_PI/n), a);
        is
            rdft(n, cos(M_PI/n), -sin(M_PI/n), a);
            for (j = 0; j <= n - 1; j++) {
                a[j] *= 2.0 / n;
            }
        .


-------- DCT (Discrete Cosine Transform) / Inverse of DCT --------
    [definition]
        <case1> IDCT (excluding scale)
            C[k] = sum_j=0^n-1 a[j]*cos(pi*j*(k+1/2)/n), 0<=k<n
        <case2> DCT
            C[k] = sum_j=0^n-1 a[j]*cos(pi*(j+1/2)*k/n), 0<=k<n
    [usage]
        <case1>
            ddct(n, cos(M_PI/n/2), sin(M_PI/n/2), a);
        <case2>
            ddct(n, cos(M_PI/n/2), -sin(M_PI/n/2), a);
    [parameters]
        n             :data length (int)
                       n >= 2, n = power of 2
        a[0...n-1]    :input/output data (double *)
                       output data
                           a[k] = C[k], 0<=k<n
    [remark]
        Inverse of
            ddct(n, cos(M_PI/n/2), -sin(M_PI/n/2), a);
        is
```

```
        a[0] *= 0.5;
        ddct(n, cos(M_PI/n/2), sin(M_PI/n/2), a);
        for (j = 0; j <= n - 1; j++) {
            a[j] *= 2.0 / n;
        }
    .


-------- DST (Discrete Sine Transform) / Inverse of DST --------
    [definition]
        <case1> IDST (excluding scale)
            S[k] = sum_j=1^n A[j]*sin(pi*j*(k+1/2)/n), 0<=k<n
        <case2> DST
            S[k] = sum_j=0^n-1 a[j]*sin(pi*(j+1/2)*k/n), 0<k<=n
    [usage]
        <case1>
            ddst(n, cos(M_PI/n/2), sin(M_PI/n/2), a);
        <case2>
            ddst(n, cos(M_PI/n/2), -sin(M_PI/n/2), a);
    [parameters]
        n               :data length (int)
                         n >= 2, n = power of 2
        a[0...n-1]      :input/output data (double *)
                         <case1>
                             input data
                                 a[j] = A[j], 0<j<n
                                 a[0] = A[n]
                             output data
                                 a[k] = S[k], 0<=k<n
                         <case2>
                             output data
                                 a[k] = S[k], 0<k<n
                                 a[0] = S[n]
    [remark]
        Inverse of
            ddst(n, cos(M_PI/n/2), -sin(M_PI/n/2), a);
        is
            a[0] *= 0.5;
            ddst(n, cos(M_PI/n/2), sin(M_PI/n/2), a);
            for (j = 0; j <= n - 1; j++) {
                a[j] *= 2.0 / n;
            }
        .


-------- Cosine Transform of RDFT (Real Symmetric DFT) --------
    [definition]
        C[k] = sum_j=0^n a[j]*cos(pi*j*k/n), 0<=k<=n
    [usage]
        dfct(n, cos(M_PI/n), sin(M_PI/n), a);
    [parameters]
        n               :data length - 1 (int)
                         n >= 2, n = power of 2
        a[0...n]        :input/output data (double *)
                         output data
                             a[k] = C[k], 0<=k<=n
    [remark]
        Inverse of
            a[0] *= 0.5;
            a[n] *= 0.5;
            dfct(n, cos(M_PI/n), sin(M_PI/n), a);
```

```
        is
            a[0] *= 0.5;
            a[n] *= 0.5;
            dfct(n, cos(M_PI/n), sin(M_PI/n), a);
            for (j = 0; j <= n; j++) {
                a[j] *= 2.0 / n;
            }
        .


-------- Sine Transform of RDFT (Real Anti-symmetric DFT) --------
    [definition]
        S[k] = sum_j=1^n-1 a[j]*sin(pi*j*k/n), 0<k<n
    [usage]
        dfst(n, cos(M_PI/n), sin(M_PI/n), a);
    [parameters]
        n             :data length + 1 (int)
                       n >= 2, n = power of 2
        a[0...n-1]    :input/output data (double *)
                       output data
                           a[k] = S[k], 0<k<n
                       (a[0] is used for work area)
    [remark]
        Inverse of
            dfst(n, cos(M_PI/n), sin(M_PI/n), a);
        is
            dfst(n, cos(M_PI/n), sin(M_PI/n), a);
            for (j = 1; j <= n - 1; j++) {
                a[j] *= 2.0 / n;
            }
        .
*/


void bitrv2(int n, double *a)
{
    int j, j1, k, k1, l, m, m2, n2;
    double xr, xi;

    m = n >> 2;
    m2 = m << 1;
    n2 = n - 2;
    k = 0;
    for (j = 0; j <= m2 - 4; j += 4) {
        if (j < k) {
            xr = a[j];
            xi = a[j + 1];
            a[j] = a[k];
            a[j + 1] = a[k + 1];
            a[k] = xr;
            a[k + 1] = xi;
        } else if (j > k) {
            j1 = n2 - j;
            k1 = n2 - k;
            xr = a[j1];
            xi = a[j1 + 1];
            a[j1] = a[k1];
            a[j1 + 1] = a[k1 + 1];
            a[k1] = xr;
            a[k1 + 1] = xi;
        }
```

```
            kl = m2 + k;
            xr = a[j + 2];
            xi = a[j + 3];
            a[j + 2] = a[kl];
            a[j + 3] = a[kl + 1];
            a[kl] = xr;
            a[kl + 1] = xi;
            l = m;
            while (k >= l) {
                k -= l;
                l >>= 1;
            }
            k += l;
        }
}

void cdft(int n, double wr, double wi, double *a)
{
    void bitrv2(int n, double *a);
    int i, j, k, l, m;
    double wkr, wki, wdr, wdi, ss, xr, xi;

    m = n;
    while (m > 4) {
        l = m >> 1;
        wkr = 1;
        wki = 0;
        wdr = 1 - 2 * wi * wi;
        wdi = 2 * wi * wr;
        ss = 2 * wdi;
        wr = wdr;
        wi = wdi;
        for (j = 0; j <= n - m; j += m) {
            i = j + l;
            xr = a[j] - a[i];
            xi = a[j + 1] - a[i + 1];
            a[j] += a[i];
            a[j + 1] += a[i + 1];
            a[i] = xr;
            a[i + 1] = xi;
            xr = a[j + 2] - a[i + 2];
            xi = a[j + 3] - a[i + 3];
            a[j + 2] += a[i + 2];
            a[j + 3] += a[i + 3];
            a[i + 2] = wdr * xr - wdi * xi;
            a[i + 3] = wdr * xi + wdi * xr;
        }
        for (k = 4; k <= l - 4; k += 4) {
            wkr -= ss * wdi;
            wki += ss * wdr;
            wdr -= ss * wki;
            wdi += ss * wkr;
            for (j = k; j <= n - m + k; j += m) {
                i = j + l;
                xr = a[j] - a[i];
                xi = a[j + 1] - a[i + 1];
                a[j] += a[i];
                a[j + 1] += a[i + 1];
                a[i] = wkr * xr - wki * xi;
                a[i + 1] = wkr * xi + wki * xr;
```

```
                xr = a[j + 2] - a[i + 2];
                xi = a[j + 3] - a[i + 3];
                a[j + 2] += a[i + 2];
                a[j + 3] += a[i + 3];
                a[i + 2] = wdr * xr - wdi * xi;
                a[i + 3] = wdr * xi + wdi * xr;
            }
        }
        m = l;
    }
    if (m > 2) {
        for (j = 0; j <= n - 4; j += 4) {
            xr = a[j] - a[j + 2];
            xi = a[j + 1] - a[j + 3];
            a[j] += a[j + 2];
            a[j + 1] += a[j + 3];
            a[j + 2] = xr;
            a[j + 3] = xi;
        }
    }
    if (n > 4) {
        bitrv2(n, a);
    }
}


void rdft(int n, double wr, double wi, double *a)
{
    void cdft(int n, double wr, double wi, double *a);
    int j, k;
    double wkr, wki, wdr, wdi, ss, xr, xi, yr, yi;

    if (n > 4) {
        wkr = 0;
        wki = 0;
        wdr = wi * wi;
        wdi = wi * wr;
        ss = 4 * wdi;
        wr = 1 - 2 * wdr;
        wi = 2 * wdi;
        if (wi >= 0) {
            cdft(n, wr, wi, a);
            xi = a[0] - a[1];
            a[0] += a[1];
            a[1] = xi;
        }
        for (k = (n >> 1) - 4; k >= 4; k -= 4) {
            j = n - k;
            xr = a[k + 2] - a[j - 2];
            xi = a[k + 3] + a[j - 1];
            yr = wdr * xr - wdi * xi;
            yi = wdr * xi + wdi * xr;
            a[k + 2] -= yr;
            a[k + 3] -= yi;
            a[j - 2] += yr;
            a[j - 1] -= yi;
            wkr += ss * wdi;
            wki += ss * (0.5 - wdr);
            xr = a[k] - a[j];
            xi = a[k + 1] + a[j + 1];
            yr = wkr * xr - wki * xi;
```

```
            yi = wkr * xi + wki * xr;
            a[k] -= yr;
            a[k + 1] -= yi;
            a[j] += yr;
            a[j + 1] -= yi;
            wdr += ss * wki;
            wdi += ss * (0.5 - wkr);
        }
        j = n - 2;
        xr = a[2] - a[j];
        xi = a[3] + a[j + 1];
        yr = wdr * xr - wdi * xi;
        yi = wdr * xi + wdi * xr;
        a[2] -= yr;
        a[3] -= yi;
        a[j] += yr;
        a[j + 1] -= yi;
        if (wi < 0) {
            a[1] = 0.5 * (a[0] - a[1]);
            a[0] -= a[1];
            cdft(n, wr, wi, a);
        }
    } else {
        if (wi < 0) {
            a[1] = 0.5 * (a[0] - a[1]);
            a[0] -= a[1];
        }
        if (n > 2) {
            xr = a[0] - a[2];
            xi = a[1] - a[3];
            a[0] += a[2];
            a[1] += a[3];
            a[2] = xr;
            a[3] = xi;
        }
        if (wi >= 0) {
            xi = a[0] - a[1];
            a[0] += a[1];
            a[1] = xi;
        }
    }
}


void ddct(int n, double wr, double wi, double *a)
{
    void rdft(int n, double wr, double wi, double *a);
    int j, k, m;
    double wkr, wki, wdr, wdi, ss, xr;

    if (n > 2) {
        wkr = 0.5;
        wki = 0.5;
        wdr = 0.5 * (wr - wi);
        wdi = 0.5 * (wr + wi);
        ss = 2 * wi;
        if (wi < 0) {
            xr = a[n - 1];
            for (k = n - 2; k >= 2; k -= 2) {
                a[k + 1] = a[k] - a[k - 1];
                a[k] += a[k - 1];
```

```
            }
            a[1] = 2 * xr;
            a[0] *= 2;
            rdft(n, 1 - ss * wi, ss * wr, a);
            xr = wdr;
            wdr = wdi;
            wdi = xr;
            ss = -ss;
        }
        m = n >> 1;
        for (k = 1; k <= m - 3; k += 2) {
            j = n - k;
            xr = wdi * a[k] - wdr * a[j];
            a[k] = wdr * a[k] + wdi * a[j];
            a[j] = xr;
            wkr -= ss * wdi;
            wki += ss * wdr;
            xr = wki * a[k + 1] - wkr * a[j - 1];
            a[k + 1] = wkr * a[k + 1] + wki * a[j - 1];
            a[j - 1] = xr;
            wdr -= ss * wki;
            wdi += ss * wkr;
        }
        k = m - 1;
        j = n - k;
        xr = wdi * a[k] - wdr * a[j];
        a[k] = wdr * a[k] + wdi * a[j];
        a[j] = xr;
        a[m] *= wki + ss * wdr;
        if (wi >= 0) {
            rdft(n, 1 - ss * wi, ss * wr, a);
            xr = a[1];
            for (k = 2; k <= n - 2; k += 2) {
                a[k - 1] = a[k] - a[k + 1];
                a[k] += a[k + 1];
            }
            a[n - 1] = xr;
        }
    } else {
        if (wi >= 0) {
            xr = 0.5 * (wr + wi) * a[1];
            a[1] = a[0] - xr;
            a[0] += xr;
        } else {
            xr = a[0] - a[1];
            a[0] += a[1];
            a[1] = 0.5 * (wr - wi) * xr;
        }
    }
}


void ddst(int n, double wr, double wi, double *a)
{
    void rdft(int n, double wr, double wi, double *a);
    int j, k, m;
    double wkr, wki, wdr, wdi, ss, xr;

    if (n > 2) {
        wkr = 0.5;
        wki = 0.5;
```

```
        wdr = 0.5 * (wr - wi);
        wdi = 0.5 * (wr + wi);
        ss = 2 * wi;
        if (wi < 0) {
            xr = a[n - 1];
            for (k = n - 2; k >= 2; k -= 2) {
                a[k + 1] = a[k] + a[k - 1];
                a[k] -= a[k - 1];
            }
            a[1] = -2 * xr;
            a[0] *= 2;
            rdft(n, 1 - ss * wi, ss * wr, a);
            xr = wdr;
            wdr = -wdi;
            wdi = xr;
            wkr = -wkr;
        }
        m = n >> 1;
        for (k = 1; k <= m - 3; k += 2) {
            j = n - k;
            xr = wdi * a[j] - wdr * a[k];
            a[k] = wdr * a[j] + wdi * a[k];
            a[j] = xr;
            wkr -= ss * wdi;
            wki += ss * wdr;
            xr = wki * a[j - 1] - wkr * a[k + 1];
            a[k + 1] = wkr * a[j - 1] + wki * a[k + 1];
            a[j - 1] = xr;
            wdr -= ss * wki;
            wdi += ss * wkr;
        }
        k = m - 1;
        j = n - k;
        xr = wdi * a[j] - wdr * a[k];
        a[k] = wdr * a[j] + wdi * a[k];
        a[j] = xr;
        a[m] *= wki + ss * wdr;
        if (wi >= 0) {
            rdft(n, 1 - ss * wi, ss * wr, a);
            xr = a[1];
            for (k = 2; k <= n - 2; k += 2) {
                a[k - 1] = a[k + 1] - a[k];
                a[k] += a[k + 1];
            }
            a[n - 1] = -xr;
        }
    } else {
        if (wi >= 0) {
            xr = 0.5 * (wr + wi) * a[1];
            a[1] = xr - a[0];
            a[0] += xr;
        } else {
            xr = a[0] + a[1];
            a[0] -= a[1];
            a[1] = 0.5 * (wr - wi) * xr;
        }
    }
}


void bitrv(int n, double *a)
```

```
{
    int j, k, l, m, m2, n1;
    double xr;

    if (n > 2) {
        m = n >> 2;
        m2 = m << 1;
        n1 = n - 1;
        k = 0;
        for (j = 0; j <= m2 - 2; j += 2) {
            if (j < k) {
                xr = a[j];
                a[j] = a[k];
                a[k] = xr;
            } else if (j > k) {
                xr = a[n1 - j];
                a[n1 - j] = a[n1 - k];
                a[n1 - k] = xr;
            }
            xr = a[j + 1];
            a[j + 1] = a[m2 + k];
            a[m2 + k] = xr;
            l = m;
            while (k >= l) {
                k -= l;
                l >>= 1;
            }
            k += l;
        }
    }
}


void dfct(int n, double wr, double wi, double *a)
{
    void ddct(int n, double wr, double wi, double *a);
    void bitrv(int n, double *a);
    int j, k, m, mh;
    double xr, xi, an;

    m = n >> 1;
    for (j = 0; j <= m - 1; j++) {
        k = n - j;
        xr = a[j] + a[k];
        a[j] -= a[k];
        a[k] = xr;
    }
    an = a[n];
    while (m >= 2) {
        ddct(m, wr, wi, a);
        xr = 1 - 2 * wi * wi;
        wi *= 2 * wr;
        wr = xr;
        bitrv(m, a);
        mh = m >> 1;
        xi = a[m];
        a[m] = a[0];
        a[0] = an - xi;
        an += xi;
        for (j = 1; j <= mh - 1; j++) {
            k = m - j;
```

```
            xr = a[m + k];
            xi = a[m + j];
            a[m + j] = a[j];
            a[m + k] = a[k];
            a[j] = xr - xi;
            a[k] = xr + xi;
        }
        xr = a[mh];
        a[mh] = a[m + mh];
        a[m + mh] = xr;
        m = mh;
    }
    xi = a[1];
    a[1] = a[0];
    a[0] = an + xi;
    a[n] = an - xi;
    bitrv(n, a);
}


void dfst(int n, double wr, double wi, double *a)
{
    void ddst(int n, double wr, double wi, double *a);
    void bitrv(int n, double *a);
    int j, k, m, mh;
    double xr, xi;

    m = n >> 1;
    for (j = 1; j <= m - 1; j++) {
        k = n - j;
        xr = a[j] - a[k];
        a[j] += a[k];
        a[k] = xr;
    }
    a[0] = a[m];
    while (m >= 2) {
        ddst(m, wr, wi, a);
        xr = 1 - 2 * wi * wi;
        wi *= 2 * wr;
        wr = xr;
        bitrv(m, a);
        mh = m >> 1;
        for (j = 1; j <= mh - 1; j++) {
            k = m - j;
            xr = a[m + k];
            xi = a[m + j];
            a[m + j] = a[j];
            a[m + k] = a[k];
            a[j] = xr + xi;
            a[k] = xr - xi;
        }
        a[m] = a[0];
        a[0] = a[m + mh];
        a[m + mh] = a[mh];
        m = mh;
    }
    a[1] = a[0];
    a[0] = 0;
    bitrv(n, a);
}
```

V.4.0. Prueba del Programa fft2f.c

```
/* Prueba del fft2f.c */

#include <math.h>
#include <stdio.h>
#include "fft2f.c"
#ifndef M_PI
#define M_PI 3.14159265358979323846
#endif
#define MAX(x,y) ((x) > (y) ? (x) : (y))

/* random number generator, 0 <= RND < 1 */
#define RND(p) ((*(p) = (*(p) * 7141 + 54773) % 259200) * (1.0 / 259200))

#define NMAX 8192

main()
{
    void cdft(int, double, double, double *);
    void rdft(int, double, double, double *);
    void ddct(int, double, double, double *);
    void ddst(int, double, double, double *);
    void dfct(int, double, double, double *);
    void dfst(int, double, double, double *);
    void putdata(int nini, int nend, double *a);
    double errorcheck(int nini, int nend, double scale, double *a);
    int n;
    double a[NMAX + 1], err;

    printf("data length n=? \n");
    scanf("%d", &n);

    /* check of CDFT */
    putdata(0, n - 1, a);
    cdft(n, cos(2 * M_PI / n), sin(2 * M_PI / n), a);
    cdft(n, cos(2 * M_PI / n), -sin(2 * M_PI / n), a);
    err = errorcheck(0, n - 1, 2.0 / n, a);
    printf("cdft err= %lg \n", err);

    /* check of RDFT */
    putdata(0, n - 1, a);
    rdft(n, cos(M_PI / n), sin(M_PI / n), a);
    rdft(n, cos(M_PI / n), -sin(M_PI / n), a);
    err = errorcheck(0, n - 1, 2.0 / n, a);
    printf("rdft err= %lg \n", err);

    /* check of DDCT */
    putdata(0, n - 1, a);
    ddct(n, cos(M_PI / (2 * n)), sin(M_PI / (2 * n)), a);
    ddct(n, cos(M_PI / (2 * n)), -sin(M_PI / (2 * n)), a);
    a[0] *= 0.5;
    err = errorcheck(0, n - 1, 2.0 / n, a);
    printf("ddct err= %lg \n", err);

    /* check of DDST */
    putdata(0, n - 1, a);
    ddst(n, cos(M_PI / (2 * n)), sin(M_PI / (2 * n)), a);
    ddst(n, cos(M_PI / (2 * n)), -sin(M_PI / (2 * n)), a);
    a[0] *= 0.5;
    err = errorcheck(0, n - 1, 2.0 / n, a);
    printf("ddst err= %lg \n", err);
```

```
    /* check of DFCT */
    putdata(0, n, a);
    a[0] *= 0.5;
    a[n] *= 0.5;
    dfct(n, cos(M_PI / n), sin(M_PI / n), a);
    a[0] *= 0.5;
    a[n] *= 0.5;
    dfct(n, cos(M_PI / n), sin(M_PI / n), a);
    err = errorcheck(0, n, 2.0 / n, a);
    printf("dfct err= %lg \n", err);

    /* check of DFST */
    putdata(1, n - 1, a);
    dfst(n, cos(M_PI / n), sin(M_PI / n), a);
    dfst(n, cos(M_PI / n), sin(M_PI / n), a);
    err = errorcheck(1, n - 1, 2.0 / n, a);
    printf("dfst err= %lg \n", err);
}


void putdata(int nini, int nend, double *a)
{
    int j, seed = 0;

    for (j = nini; j <= nend; j++) {
        a[j] = RND(&seed);
    }
}


double errorcheck(int nini, int nend, double scale, double *a)
{
    int j, seed = 0;
    double err = 0, e;

    for (j = nini; j <= nend; j++) {
        e = RND(&seed) - a[j] * scale;
        err = MAX(err, fabs(e));
    }
    return err;
}
```

CONCLUSIONES

CONCLUSIONES

Un Convertidor Analógico Digital genera una palabra digital la cual tiene que ser representativa de la magnitud de una señal analógica. Por ser el CAD un dispositivo cuantizador o contador, entonces, habilita el error de cuantizacion. El CAD es un dispositivo no lineal.

Al utilizar un CAD existe diferentes tipos de errores y efectos, tanto estáticos como dinámicos como son: DNL, INL, offset, Gain, Jitter, Aliasing, de Apertura, Termal Drift.

El error DNL existe cuando el tamaño del escalón es diferente al ideal.

El error INL es la medida de la distancia a la línea recta dibujada entre, los códigos en cada fin de la función de transferencia y origen, y la línea mas alejada será errónea usualmente se expresa de la forma señal/ (ruido + distorsión).

El error por offset, es el punto cero de la función de transferencia. El error de ganancia es un cambio en la pendiente de la función de transferencia del convertidor relativo al estado del voltaje de referencia. Jitter son las variaciones de tiempo en el periodo de muestreo respecto a la frecuencia de base o fundamental.

Si se muestrea una señal a una frecuencia menor que el doble de su frecuencia máxima, causa aliasing.

El sobremuestreo mejora el Rango Dinámico del Convertidor Analógico Digital y reduce el numero de polos necesarios para diseñar un filtro digital. El error de apertura es definido como los errores de amplitud y tiempo de los puntos de datos muestreados debido a la incertidumbre de los cambios dinámicos de los datos durante el muestreo.

Los filtros Digitales son utilizados para prevenir el aliasing, son de menor costo; además, pueden alcanzar una mayor complejidad, reproducirse con gran exactitud, y tener mejor estabilidad sobre tiempo y temperatura.

En los convertidores de Audio principalmente se utilizan los filtros digitales internos en el C.I. Se confirma el teorema de Nyquist, Un mínimo de dos muestras por ciclo en el ancho de banda de la señal de datos, son requeridas en un sistema ideal de muestreo de datos, para producir datos muestreados sin perdida de información.

Existen 3 diferentes tipos de arquitectura de Convertidores Analógicos Digital, el Flash, el de Aproximaciones Sucesivas y el $\Lambda\Sigma$, pero algunos fabricantes dividen la arquitectura flash en dos.

El convertidor Flash sirve para digitalizar las altas frecuencias, y así lo hace pero con muy poca resolución. Se puede hablar de convertidor Flash pipelined o subranging. La arquitectura *pipelined* es similar a la aproximación *subranging*, pero utiliza un solo bit comparador para cada estado. N comparadores son usados para obtener un convertidor analógico-digital. La arquitectura *pipelined* utiliza un número menor de comparadores, pero tiene que utilizar al menos un ciclo de reloj para acompañar cada bit de la conversión. Por esta razón, este tiene un mayor tiempo de retardo que la arquitectura en paralelo.

El convertidor de Registro de Aproximaciones Sucesivas (RAS) - *Successive approximation register's (SAR)* sirven para un rango de frecuencia intermedia, regularmente utilizado en aplicaciones con multiplexores para digitalizar muchos canales con información de baja frecuencia.

El Convertidor $\Lambda\Sigma$ es un convertidor de sobremuestreo y se desarrollan efectivamente con frecuencias por debajo de los 200khz del ancho de banda.

Si la frecuencia de muestreo es el doble tanto como el ancho de banda, el ruido debido a la cuantización se extiende tanto como el doble del ancho de banda. Bajo esta condición la potencia del ruido por unidad de frecuencia será reducida a la mitad.

Los histogramas son usados para medir la Corriente Directa con exactitud o las características del desarrollo estático tales como el *offset*. Las *FFT´s* son utilizadas para medir las características del desarrollo dinámico como la linealidad. Ciertas características tales como la linealidad y el ruido pueden ser medidas con cualquier técnica. La exactitud de la estimación es dependiente del número de muestras coleccionadas. El número de muestras que necesitan ser coleccionadas depende de algunos factores incluyendo el intervalo confidencial, grado de exactitud y la distribución de las variables. Intervalos confiables son el grado de certeza donde la estimación esta dentro de un rango específico. Más altos grados de exactitud o de distribuciones con una gran variación requerirán más muestras para estimar. En este punto el promedio es un método utilizado para incrementar la resolución de un convertidor analógico digital CAD y disminuir la incertidumbre debida al ruido. Como discusión, en el histograma donde se analiza el ruido, la exactitud del promedio y la incertidumbre asociada con los datos muestreados, depende del *set* de muestras de la desviación. Una pobre DNL distorsiona la distribución de ruido aleatoreamente. Códigos con mayor amplitud tienen una alta probabilidad de ocurrencia que los códigos angostos. El error DNL afecta la distribución del histograma. Si la distribución esta severamente distorsionada, la estadística puede ser no confiable. Es importante escoger un CAD de buenas características si la aplicación requiere promediarse. Los histogramas pueden ser utilizados para medir la Diferencia no lineal (DNL) de un convertidor analógico digital sobre un pequeño tramo. Una señal triángulo es aplicada a la entrada del

CAD y un set de muestras de datos es colectada. Si el DAC tiene un DNL perfecto, cada código en la función de transferencia tiene la misma probabilidad de ocurrencia. Una señal triángulo perfecta producirá un histograma perfectamente plano.

En la medición del desarrollo dinámico de un CAD, el principal objetivo es el investigar como se altera la función de conversión del espectro de la señal trasmitida a través del sistema. El análisis de espectros se utiliza para encontrar el contenido de frecuencia de las señales. Aquí el dominio de las señales en el tiempo es transformado al dominio de la frecuencia. Las porciones de magnitud de los elementos de frecuencia son utilizadas para determinar la distribución con respecto a la frecuencia, construyendo una gráfica espectral de potencias. La potencia del espectro es utilizada para medir la distorsión del sistema, el rango dinámico, la atenuación, y el ruido inducido. El proceso de la transformada rápida de Fourier *(FFT)* empieza en el CAD, el cual convierte una señal analógica en una representación digital discreta (muestreo). La amplitud de la señal analógica es convertida a una representación n-bit binaria llamada palabra digital. El CAD saca estas palabras digitales a un rango seleccionado por una frecuencia de muestreo (*Fs*). Esta frecuencia de muestreo es primeramente determinada o por el *troughput* del CAD. Una muestra del *set* de puntos de datos es colectada por el algoritmo *FFT*. El algoritmo *FFT* opera sobre un *set* de muestras N, para las cuales N deberá ser una potencia de 2, por ejemplo, N= 2^8 =256 o N= 2^9 = 512. El tamaño del *set* de muestras es determinado por la capacidad del proceso (tamaño de la memoria y procesamiento *troughtput*), por las señales de entrada dinámicas, y por el grado de resolución de la frecuencia.

El algoritmo *FFT* asume que la señal de entrada es periódica sobre un número de puntos en la *FFT*. Esto significa que copias múltiples de la señal pueden ser colocadas al final sin ninguna discontinuidad en el punto donde se unen. Una

FFT de un dato no periódico será el resultado en elementos de alta frecuencia que no existen en la señal actual. La periodicidad del algoritmo de la *FFT* requiere que el *set* de muestras contenga un número entero de ciclos de señal y no es práctico para la mayoría de las aplicaciones. Esto en efecto hace que *Fs* y N sean función de la frecuencia de entrada.

La periodicidad es más complicada cuando la señal está compuesta de múltiples elementos de frecuencia o no es solamente una señal seno. Las ventanas son utilizadas para reducir la fuga espectral que resulta de utilizar *FFT* sobre señales no dinámicas o no periódicas. *Windowing* multiplica la señal en el dominio del tiempo por una función que atenúa la amplitud al finalizar. Debido a esto se tienen las discontinuidades.

El objetivo en la selección *windows* para medidas dinámicas es el mantener la energía fundamental en el lóbulo principal y tener una fuga insignificante en los lóbulos de adyacentes. Con las funciones *windows*, el primer paso a tratar es la mancha de la señal de entrada entre varios *bins* y segundo tratar con los lóbulos adyacentes. La resolución de la frecuencia y la fuga espectral son influenciados por la selección de una función *windows*.

Escoger el *windows* apropiado depende de la aplicación, el desarrollo del sistema, y de la información requerida. Medir el desarrollo de un sistema con un CAD es una aplicación típica para el análisis con la transformada rápida de fourier *FFT*. El Objetivo es el de obtener la potencia espectral de una señal seno puro. Una señal senoidal pura reduce el número de variables y permite la medida de la ocurrencia del desarrollo del sistema. El objetivo es el mantener toda la señal de potencia en lóbulo principal. Esto requiere que una función *window* de lóbulos adyacentes sean más bajos que el ruido de piso teórico. Lo pequeño del lóbulo reduce la fuga espectral para encasillar afuera el lóbulo principal, haciendo la lectura por el procesador de la potencia de la señal más fácil. La amplitud del

lóbulo principal y un nivel más alto de lóbulos adyacentes para las 5 *windows* son cinco tipos descritos en el punto III.5.2. En la medición del desarrollo dinámico, los cálculos son más fáciles cuando una de las *windows* de los lóbulos adyacentes se encuentran por debajo del ruido de piso mínimo. Los resultados en la figura del ruido de piso están dictados por el desarrollo CAD.

El ruido de piso mínimo se configura por la resolución y por el número de muestras utilizadas en el *FFT*. El espectro de potencia ilustra la densidad espectral de la señal medida la cual puede mostrar la corrupción del seno puro por un sistema electrónico. Son cuatro las medidas de desarrollo que pueden ser calculadas de un espectro de potencia obtenido de l *FFT*: *Signal-to-Noise Ratio (SNR), Signal-to-Noise, Distortion (SINAD), Signal-to-distortion (SDR), y Signal-to-Peak-Noise (SPN).* Cada parámetro da una perspicacia a las características de linealidad y ruido del sistema. Estos son utilizados en los bancos de desarrollo o identificando problemas específicos de desarrollo. Con el espectro de potencia construido, el desarrollo estadístico dinámico puede ser calculado.

Un algoritmo es utilizado para identificar las varias subsecciones del espectro de potencia. Primero el espectro es buscado por el *bin* más alto, localizando la señal fundamental. La potencia de la señal fundamental es calculada con consideración por el tipo de *window* utilizado por el algoritmo. La siguiente sección identifica los *bins* de los *offsets* de CD. El tamaño del área depende del tipo *window* utilizado en el algoritmo. El CD no se utiliza para medir el desarrollo dinámico. Con las secciones definidas de la señal fundamental y la señal de CD, la potencia que permanece en los *bins* se suman y un radio determinado versus la señal de potencia fundamental. Este radio es el valor *SINAD (Signal-to Distortion).* El siguiente paso es para localizar el segundo bin más alto fuera de las áreas de la fundamental y del *offset de* CD. Este *bin* localiza el área del pico de ruido. La potencia en el pico de la señal de ruido se calcula y el radio del *Signal-to-Peak-*

Noise (SPN) se obtiene. El último paso es para localizar las armónicas secundarias asociadas con la señal fundamental. Aquí el algoritmo busca el espectro para las armónicas sin *aliasing* o con *aliasing*. La potencia asociada con las armónicas secundarias se considera una distorsión. Calculando el poder de la distorsión se tiene que lidiar para obtener el *SD* y los radios *SNR*. El tamaño del *set* de muestras utilizado con el algoritmo *FFT* determina la resolución de frecuencia y la mancha de ruido la cual hace el ruido de piso. Más información está contenida en un *set* grande de muestras que un *set* pequeño de muestras. Un *FFT* de un *set* grande de muestras usa la información para proveer una resolución de frecuencia más fina. La amplitud de banda es igual a la frecuencia de muestreo dividida por el número de muestras (Fs/N).

Con un menor ruido de piso, las señales de bajo nivel son más aparentes. Los niveles de energía a frecuencias específicas permanecerán descargadas con el incremento del tamaño de las muestras. Una concepción errónea es que una señal más baja que el rango dinámico de frecuencia puede no ser detectado ni medido. Utilizando un CAD ideal de 16 bits, el mejor rango dinámico es 98 dB, señales que están por debajo de los 100dB del *Full Scale* pueden detectarse.

Con la reciente proliferación de los Convertidores Analógicos Digital (CAD´s), los Convertidores Digital Analógico (CDA´s), y la variación en los esquemas de código que utilizan estos convertidores, se tiene la necesidad de analizar estos esquemas de código que son utilizados para hablar con el "Mundo Digital". Al pretender utilizar un CAD se debe de analizar un par de criterios que el tesista recomienda continuación. Teniendo con fin el saber que tipo de arquitectura se debe de utilizar y que características principales deberá de tener el CAD.

Primero se tiene que determinar la clasificación del diseño en la aplicación. Medición, Video Audio, Alta velocidad, Propósitos Generales, Adquisición de Datos, Instrumentación Médica, Comunicación.

Segundo, se deben de definir los siguientes parámetros al utilizar un Convertidor Analógico Digital. Resolución requerida, Números de Canales, Rango de Muestreo, Formato de la Salida (Serial, Paralela o ambas), Rango de entrada (Sí es Unipolar y Bipolar), Requerimientos de polarización, Necesidad de alimentación (Portátil, Batería; Fuente bipolar, etc.), Tipo de encapsulado - *package number.*

Otras características importantes del sistema dependiendo de la aplicación. Por ejemplo existen microcontroladores con convertidor analógico digital interno cuyas características serán función software/hardware del microcontrolador. .

Por ultimo la importancia de encontrar una función la cual te permita el implementar una interacción inteligente entre el CAD y la unidad de procesamiento, es necesario, por que la discretización FFT es un algoritmo el cual convierte una función de muestreo de valor complejo de tiempo en una función de muestreo de valor complejo de frecuencia, y este algoritmo tiene el fin el entregar un valor muy cercano al real.

BIBLIOGRAFIA

BIBLIOGRAFÍA

1.- Análisis de Fourier
Hwei P. Hsu.
Fondo Educativo Interamericano, S.A.
1970, U.S.A.

2.-Instrumentación Digital
AMICEE
Editorial Limusa
1975.

3.-Application Handbook,
Burr-Brown
1993, U.S.A.

4.-Data Book
Micro Linear
1995, U.S.A.

5.-Linear Products
Burr-Brown
1995, U.S.A.

6.-Crystal World Tour Aplication Seminar
Cristal Semiconductor Corporation
1995, U.S.A.

7.-Applications Handbook, Vol.1.
Micro Linear.
1995, U.S.A.

8.-Crystal Semiconductor Audio Databok
Cristal Semiconductor Corporation
MARCH 1995, USA.

9.-Mixed Signal Products,
Burr-Brown.
1997, U.S.A.

10.-Linear & Mixer Signal Products
Burr-Brown.
1998, U.S.A.

11.- 1997 Insight Training
Burr Brown,
Arizona, USA, 1997.

12.-Crystal Semiconductor Audio Training
Cristal Semiconductor Corporate,
San Jose California, 1998.

SITIOS EN EL WWW

SITIOS DE INTERNET EN EL WWW

Fabricantes de Circuitos Integrados (CAD):

www.burr-brown.com
www.microlinear.com
www.crystal.com
www.motorola.com
www.phillips.com
www.national.com
www.ti.com
www.adi.com

Distribuidores de Componentes Electrónicos:

www.avnet.com
www.insight-electronics.com
www.pioneer.com
www.wyle.com
www.future.com
www.arrow.com

Otros Sitios:

www.edn.com