IMPLEMENTACIÓN EN SOFTWARE DE UN CONVERSOR ADC DE ALGORITMO VARIABLE

Alfredo Ortega
Universidad de la Patagonia San Juan Bosco
Aortega@unp.edu.ar
Los Aromos 110, Trelew, Chubut, C.P.9100
TEL. 421437

Resumen:

En este documento se tratan los dispositivos de conversión Analógico-Digital. Se propone un método para bajar los costos y algoritmos de mayor eficiencia que los comunes. También se describe la realización práctica de este método, junto con la implementación tanto en hardware como en software de un sistema de muestreo de señales de muy bajo costo.

Palabras Clave:

conversor, analógico-digital, señal , algoritmos, embebido , aproximaciones sucesivas

1. INTRODUCCIÓN:

Para que una computadora pueda trabajar e interactuar con el medio ambiente que la rodea, necesita entender las señales propias de este mundo, por ejemplo, temperatura, sonido, imágenes. Pero estas señales son analógicas, o sea, son continuas en el tiempo y no pueden ser procesadas por una computadora digital, que necesita de una señal Digital, cuyos valores a través del tiempo son discretos. El dispositivo de conversión necesario en este caso se llama Conversor Analógico-Digital.

Existen varios dispositivos hardware para realizar la conversión de una señal analógica a una digital, procesable por una computadora digital. Pueden encontrarse varios ejemplos en [1]. Hoy en día se usan ampliamente en el procesamiento de señales, en usos como la grabación de sonidos, digitalización de vídeo y otros. Sin embargo, implementar estos dispositivos eficientemente en hardware requiere de una lógica complicada que los hace muchas veces la parte mas cara del equipo en cuestión.

El objetivo de este documento es presentar un dispositivo de este tipo implementado en su mayoría en software (con una mínima parte de hardware) con el objetivo de abaratar los costes de producción, y utilizar algoritmos mas eficientes que los utilizados actualmente.

2. REGISTRO DE APROXIMACIONES SUCESIVAS

Existen muchos métodos de realizar la conversión de una señal analógica a una digital [1]. Para ejemplificar nos centraremos en el método SAR (Succesive Aproximations Register) Registro de Aproximaciones sucesivas, dada su amplia utilización y mediana complejidad.

Este método bien conocido en el mundo de la electrónica consta, como el diagrama lo indica, en una señal de prueba, aquí marcada en línea negra llena que se compara constantemente con la señal de entrada cuyo valor es desconocido, aquí representada por una línea punteada azul.

Ejemplificaremos un conversor SAR de 8 bits, o sea, su salida es una octeto o Byte digital, llamado "Sample" que puede representar valores de 0 a 255. Utilizamos la siguiente escala: el valor de 0 Volts analógicos se corresponderá con el valor 127 en nuestro Byte, y los valores de 5 y –5 Volts se corresponderán con los valores de 255 y 0 respectivamente.

La señal de prueba es fruto de una conversión Digital-Analógica, ya que es generada por un circuito digital y debe ser comparada con la señal de entrada, que es analógica.

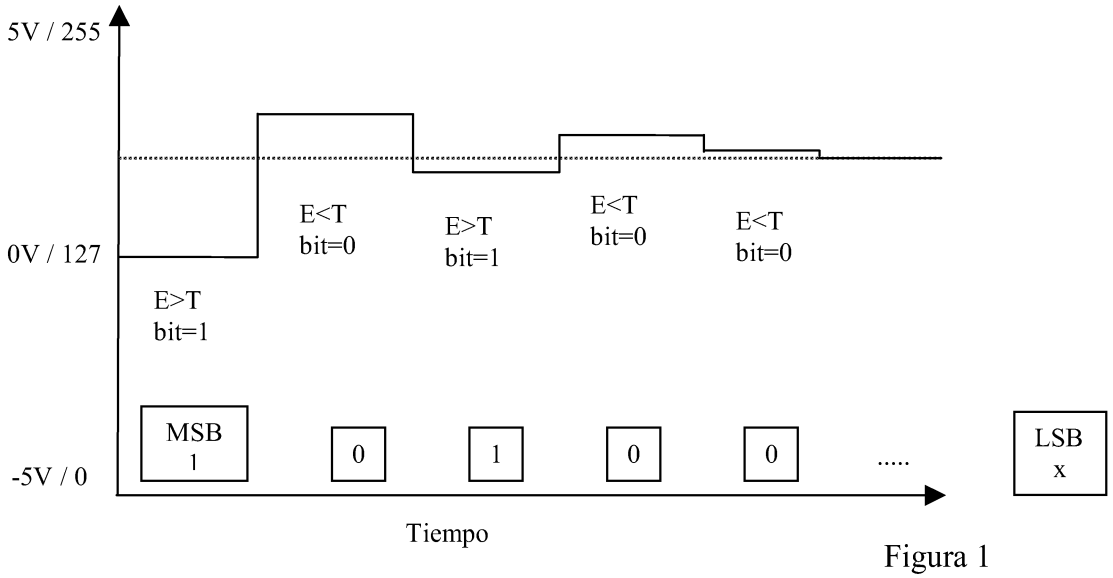
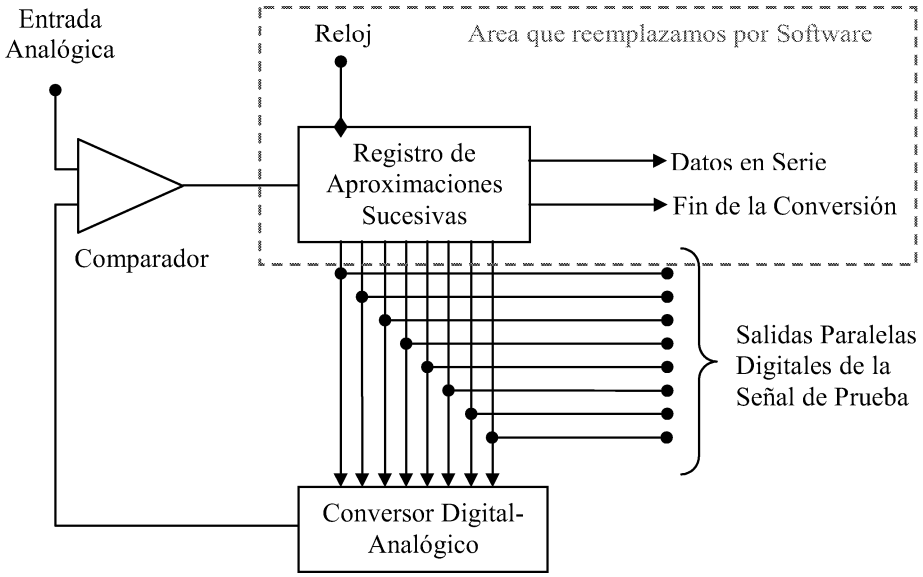


Figura 1

Esta técnica obtiene en n pasos una conversión analógica de n bits. El diagrama siguiente muestra la configuración de hardware necesaria y la parte que nuestro algoritmo suplanta:



2.1 Una breve explicación del diagrama superior:

- Comparador: Dispositivo que ante dos señales analógicas, genera una salida digital dependiendo de cual señal es la mas alta. Su salida es siempre digital, 0 o 1, no importa si sus entradas sean señales iguales. Es un componente muy sencillo y de bajo costo.
- Conversor Digital-Analogico: Este dispositivo cumple la función inversa al que estamos construyendo, o sea, convierte un número digital en una señal analógica representativa. Existen algunos muy complicados y con muchas funciones extra, pero el que necesitaremos puede ser construido simplemente con una red R-2R de resistores, cuyo costo es muy bajo. Para una explicación de la red R-2R remítase al final de este documento.
- Registro de Aproximaciones Sucesivas: Este Registro es la porción que genera la señal de prueba sobre la base de la información del Comparador, y emite los bits de la conversión en forma serial. Cuando la aproximación es suficientemente buena, emite una señal de Fin de la Conversión y puede leerse la palabra binaria representativa de la señal analógica.

Aquí mostramos que el área reemplazada por Software es el Registro de Aproximaciones Sucesivas, que es en realidad el eje central de todo el dispositivo y también la parte mas complicada ya que internamente posee una lógica de funcionamiento que no poseen los demás componentes. De esta forma, el sistema completo quedaría configurado como se ve en la Figura 3, con el procesador ocupando el lugar de la lógica del Registro SAR, ahora nombremos los pros y contras de esta configuración respecto a la anterior:

Pros:

- Se reduce el costo del sistema en total, ya que el costo del conversor en si es disminuido en un 70% o más, dependiendo de la implementación. En un sistema embebido, la diferencia entre un

microcontrolador con un conversor Analógico-Digital incluido y otro usando este sistema es muy grande.

- Se tiene un control total sobre la resolución y velocidad de conversión. Ejemplo: si para determinada aplicación se necesita una resolución de 4 bits en un conversor que tiene 8 bits cono resolución máxima, el software puede convertir tan solo los bits necesarios, dando por resultado un aumento en la velocidad de conversión, ya que se necesitan menos pasos. En este ejemplo la cantidad de conversiones por segundo aumentaría al doble exactamente, ya que en lugar de 8 pasos, se realizaría cada conversión en 4.
- La resolución y velocidad pueden ajustarse en tiempo real y de una conversión a otra. Esta característica es muy útil ya que el sistema puede adecuarse al tipo de señal que se mide.
- Se pueden utilizar una variedad de algoritmos de conversión, o una combinación de estos, abriendo el campo para investigar algoritmos mas eficientes con solo modificar el software de control. Por ejemplo, si la señal no tiene saltos o cambios bruscos de nivel, puede cambiarse a un algoritmo mas adecuado que el de Aproximaciones sucesivas, como el de seguimiento, aumentando la velocidad de sampleo.

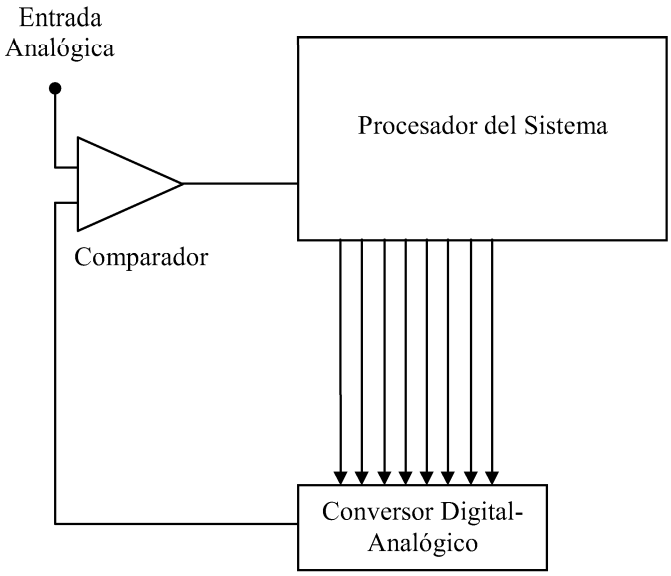


Figura 3

Contras:

- El procesador debe dedicar parte de sus ciclos a realizar la conversión, como consecuencia no se puede realizar otros cálculos mientras se esta realizando la conversión, si se usa un procesador lento.
- El programa de control debe tener una manera exacta de medir el tiempo de conversión para una digitalización correcta.
- La velocidad de sampleo o conversión esta limitada por la velocidad del procesador (O la velocidad de I/O de este)

Como características de este sistema, puede decirse que la resolución esta limitada solamente por la resolución del conversor Digital-Analogico (DAC) en conjunto con la cantidad de salidas disponibles del procesador y la velocidad, seguramente será limitada por el I/O de la interface con el procesador, ya que como norma, todos los componentes deben poder trabajar como máximo a la velocidad de esta interface.
Para una conversión de n bits y m conversiones/segundo, se requerirán de 1 entrada digital al procesador, n salidas digitales del procesador, un conversor DAC de n bits, y la frecuencia de trabajo de los componentes será como máximo ***m*(n+1)*** hertz. El segundo termino tiene en cuenta el tiempo necesario para leer la comparación.

3. IMPLEMENTACIÓN

3.1 Hardware

Se realizo la implementación de un dispositivo de este tipo utilizando como procesador a una computadora personal. Se realizo un conversor de 8 bits utilizando la Paralelo (puerto de impresora) estándar en todas las PCs [3]. Este puerto tiene una velocidad máxima de unos 500 khz, lo que nos dio una velocidad de conversión máxima de 40-44 khz a 8 bits. Como conversor Digital-Analógico se utilizo una red R-2R con resistencias calibradas y como comparador se utilizo un comparador de vídeo LM-311. El circuito completo es un poco mas complejo porque contiene varios amplificadores y filtros para obtener un instrumento utilizable. El costo de los componentes adquiridos a un minorista fue menor a los 20 Pesos/Dólares, el costo del conversor mismo fue de menos de 4 Pesos/Dolares. Se supone que en grandes cantidades el costo podría reducirse a unos centavos de Dólar.

Los requerimientos mínimos de hardware para esta implementación de prueba son un procesador 386 de 40 mhz y 4 Mb de RAM y vídeo VGA, aunque para utilizar la función de espectroscopio tridimensional (En los tres ejes se representa frecuencia, energía y tiempo) es recomendable un Pentium de 166 Mhz o mayor.

3.2 Software

El software fue preparado como un instrumento de medición, específicamente como un espectroscopio digital, procesando la señal recibida por el conversor con un algoritmo FFT de aritmética entera [2]. El software fue realizado en c++ y assembler (386) . Presenta una conveniente interfaz de ventanas con un manejador propio, ya que el sistema corre bajo el sistema operativo DOS usando un emulador de memoria EMS. Con la emulación adecuada corre también bajo Windows 95-98.

La aplicación posee mas de 5000 líneas de código pero el corazón del software es una función de conversión, que contiene el algoritmo de aproximaciones sucesivas con resolución variable, implementada en código de ensamblador x86. La función se lista en el Anexo I por ser de corta longitud y quizás ayude a comprender el funcionamiento del algoritmo.

4. ALGORITMO PROPUESTO

4.1 Coversor por Seguimiento:

Daremos una breve reseña del funcionamiento de este conversor antes de presentar uno de los algoritmos propuestos en este trabajo. El conversor por seguimiento en lugar de "aproximar

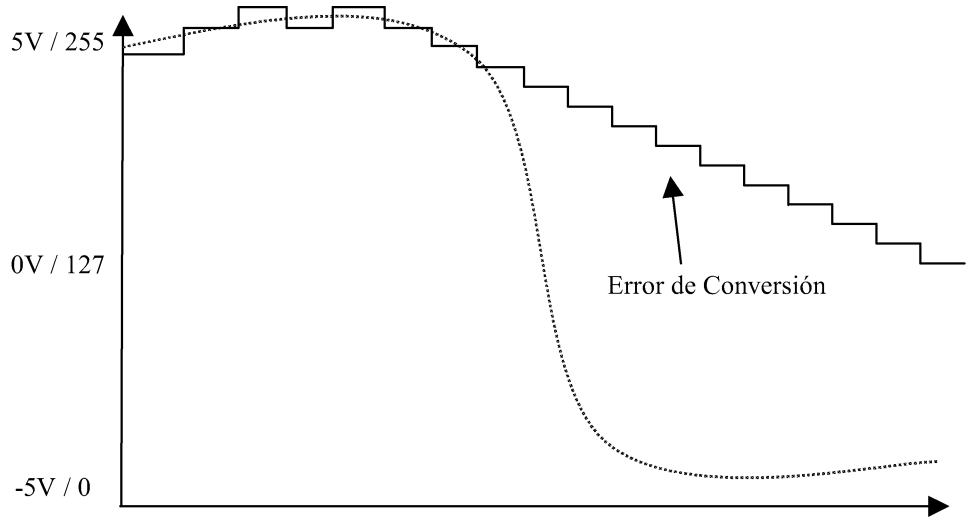


Figura 4

sucesivamente" la señal de prueba a la señal de entrada, hace la presunción de que la señal de entrada habrá cambiado muy poco desde la ultima conversión, por eso al recibir el resultado de la comparación entre ambas señales, simplemente incrementa o decrementa en 1 el sample anterior y lo envía como salida. Esta técnica funciona muy bien con señales analógicas suaves, como la senoidal, si la razón de cambio es pequeña comparada con la velocidad de las conversiones, sin embargo, como puede verse en la figura 4, si la señal cambia abruptamente este algoritmo produce errores significativos por lo que no es muy utilizado.

4.2 Algoritmo Híbrido:

Como se vio anteriormente, en la realización de este dispositivo ya se implementaron con éxito los algoritmos de Aproximaciones Sucesivas con resolución variable y el de Seguimiento, pero esta aún en fase de investigación un algoritmo de tipo Híbrido, que podría aumentar la cantidad de conversiones, sin disminuir la resolución. Para esto hagamos un breve repaso de las características de los dos primeros algoritmos:

- Aproximaciones Sucesivas: Realiza la conversión sin hacer ninguna suposición acerca del valor, ni toma en cuenta conversiones anteriores. Puede realizar conversiones sobre cualquier tipo de señal con iguales resultados. Un conversor de n bits de resolución realiza una conversión por cada n comparaciones.
- Conversor por Seguimiento: Supone que la señal tiene un comportamiento "suave" y que por lo tanto la conversión actual será un incremento o decremento de la anterior. Solamente sirve con ondas que no posean cambios bruscos como las senoidales o triangulares, su desempeño es pésimo con ondas cuadradas o con cambios súbitos de voltaje. Realiza una conversión por cada comparación, siendo mucho mas rápido que el algoritmo anterior.

La propuesta de un conversor Híbrido es combinar estos dos últimos métodos para lograr un conversor que tome en cuenta sus conversiones anteriores pero que acepte cambios bruscos en la señal.

Se propone lo siguiente: Una conversión de n bits se divide en dos etapas de n_1 y n_2 bits, siendo $\boldsymbol{n_1 + n_2 = n}$.

En la primer etapa, un conversor de seguimiento de n_1 bits aproxima la señal teniendo en cuenta el valor de la conversión anterior, quedando los primeros n_1 bits mas significativos del Sample seteados de esta manera. Este paso no llevara tiempo ya que es solamente el copiado de los n_1 bits más significativos del sample anterior sobre el actual.
En la segunda etapa, se aproximan sucesivamente los n_2 bits restantes de la conversión, partiendo no de un valor inicial arbitrario, sino del valor estimado en la primer etapa. Este paso nos llevara n_2 comparaciones.
El algoritmo resultante posee una buena respuesta a los cambios súbitos del nivel de la señal, gracias a su etapa de Aproximaciones Sucesivas, pero requiere de n_1 comparaciones menos que un conversor convencional.

Siendo A la amplitud máxima de pico a pico de la señal y n la resolución de conversión, el mayor cambio súbito en la amplitud de la señal que admite un conversor de seguimiento entro dos conversiones sin perder la señal es igual a: $A/2^n$ que es un valor muy restrictivo.
En contraposición, el Algoritmo mixto puede realizar conversiones exactas en señales con un cambio súbito de $A/2^{n1}$ que será un rango siempre mayor, ya que n_1 siempre es menor a n.

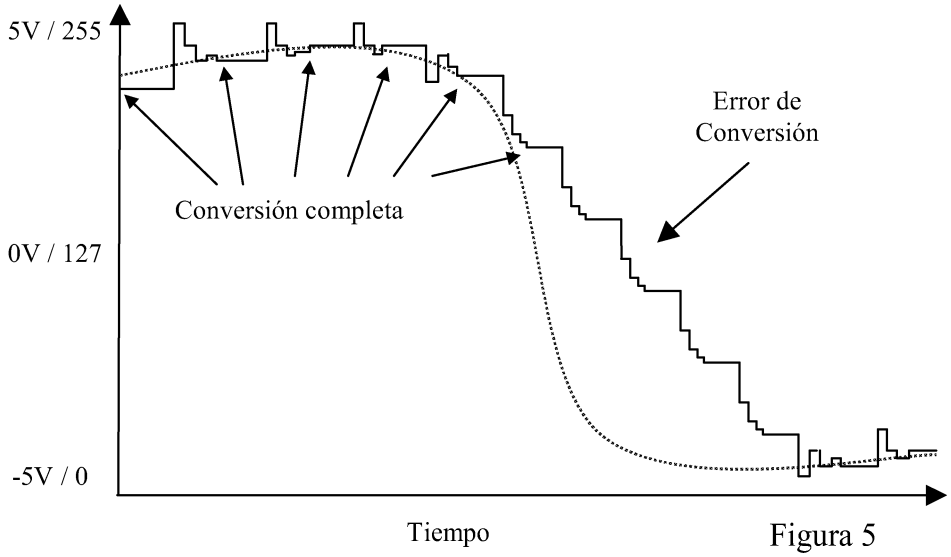


Figura 5

En la Figura 5 puede apreciarse que la respuesta a cambios abruptos de este algoritmo es mucho mejor al de seguimiento.
4.3 Conclusión: variando los parámetros n_1 y n_2 se puede adaptar el algoritmo a la señal de entrada obteniendo la máxima eficiencia. Algoritmos como este o mas complejos, que efectúen una predicción sobre la señal o efectúen algún tipo de estudio previo, pueden ser desarrollados e implementados sin costo alguno.
NOTA: Los mecanismos de “sample and hold” para el mejoramiento de la calidad de digitalización no han sido tratados en este libro pero son totalmente compatibles con este diseño en particular.

5. RED R-2R

A lo largo de este documento se ha utilizado este termino para referirse a un tipo de conversor Digital-Analogico muy simple pero apto para muchas funciones. Abajo se ilustra la estructura de dicho circuito, cada resistencia, de valor R, debe ser exactamente igual a las demás. La disposición y caídas de voltaje que esta red logra hacen posible que el voltaje en la salida sea aproximadamente $0.7\ (V_1/2+V_2/4+V_3/8...Vn/2^n)$ donde V_x es el voltaje en la entrada x.
Este circuito puede tener tanta resolución como ramas se agreguen a la red, con la precaución de que las resistencias tengan exactamente el mismo valor. La velocidad de este circuito es muy alta, ya que se trata solamente de componentes pasivos.

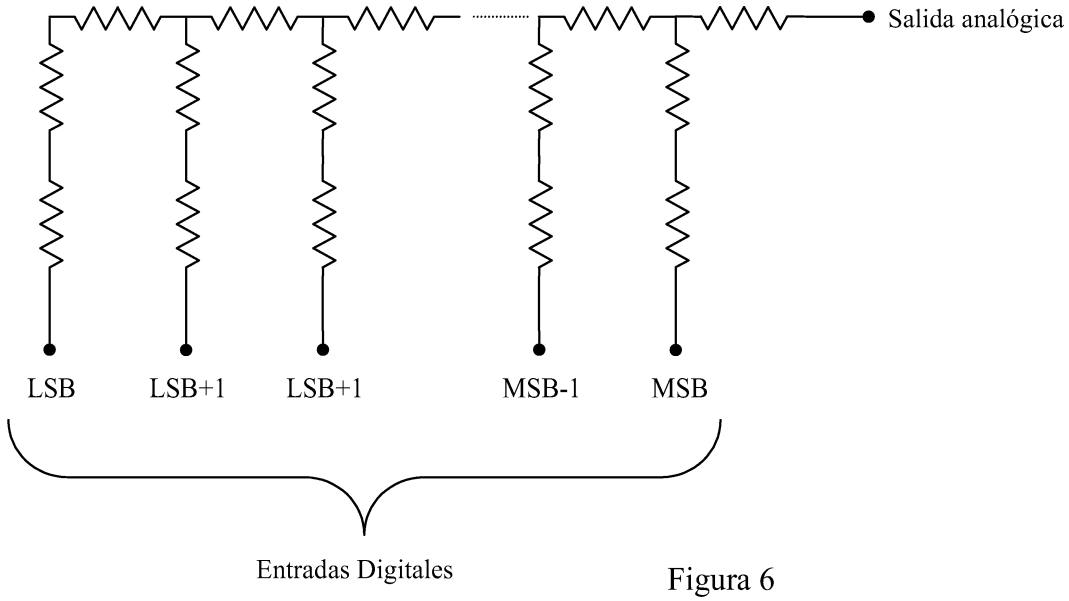


Figura 6

6. CONCLUSIÓN

Estos métodos de conversión presentan una alternativa viable a los métodos convencionales en el área de conversión Analogica-Digital de baja frecuencia y bajo costo. En el ámbito de las computadoras personales, estos dispositivos no presentan tanta ventaja debido a la baja de precio en periféricos y conversores de todo tipo. El uso de estos métodos hoy en día esta más indicado en aplicaciones embebidas, las que son muy sensibles al costo de hardware, proveyendo a cualquier dispositivo con capacidades de procesamiento y un aumento mínimo en el hardware, de un conversor Analógico-Digital y otro Digital-Analógico, brindando una interfaz analógica muy útil para interactuar con el medio ambiente.

7. VOCABULARIO

ADC: (Analogic to Digital Converter) _ Conversor Digital a Analógico

DAC: (Digital to Analogic Converter) _ Conversor Analógico a Digital

EMS: (Expanded Memory Simulator): Método antiguo para acceder a mas de 640 KB en el sistema operativo DOS sin usar un extendedor del sistema.

FFT: (Fast Fourier Tranform) Transoformada Rápida de Fourier

LSB: (Least Significant Bit) Bit Ménos Significativo

MSB: (Most Significant Bit) Bit Más Significativo

Sample: Muestra, salida de la conversión

SAR: (Succesive Aproximations Register) Registro de Aproximaciones sucesivas

Velocidad de Sampleo: Cantidad de Samples o muestras en un periodo de tiempo

8. REFERENCIAS:

[1] Jonathan A. Titus **"**Microcomputer Analog Converter, Software and Hardware Interfacing" ASIN: 0672215403

[2] Anders E. Zonst "Understanding the Fft : A Tutorial on the Algorithm & Software for Laymen, Students, Technicians & Working Engineers **"**
Citrus Press; ISBN: 0964568187

[3] Michael Tischer & Bruno Jennrich "PC Intern 6th Edition**"**
Abacus, ISBN: 1557553041

ANEXO I: Listado de la Función

La función requiere de 4 parámetros:

buf: Puntero al buffer donde almacenar las conversiones
len: Cantidad de conversiones requeridas
bits: Resolución de la conversión deseada
sincr: valor de sincronía, nivel de la señal a partir del cual se desea que comience la conversión

```
void leeLPT(char *buf,unsigned int len,unsigned char bits,unsigned char sincr)
{
asm     {
        pusha
        pushf
        push ds
        push es

        les di,buf
        mov bx,0
        }

SINCR:
asm     {
        call LEEBITS //;*********
        inc bx       //;*
        cmp bx,300   //;*
        je SIN       //;*
        cmp al,sincr //;*Intenta comenzar el barrido con
        jne SINCR    //;*la onda sincronizada.
        mov ch,al    //;*Espera 300 conversiones a la sincronización
        call LEEBITS //;*
        cmp al,ch    //;*
        jl SINCR     //;*********
        };
SIN:
asm     {
        mov cx,len
        }
inicia:
asm     {
        call LEEBITS //*******
        stosb        //*Llena buffer con los datos
        loop inicia  //*
        jmp sale     //*******
        }
LEEBITS:
asm     {
LEEBIT PROC NEAR        //***** Este procedimiento es el que realiza la conversión.
           push bx      //*
```

```
            push cx             //*Inicializa los registros para
            mov bl,128          //*empezar a revisar por el bit 7 (128).
            mov al,bl           //*
            mov cx,bits         //***** aqui se carga cx con la cantidad de bits requeridos
            };
s1:
asm         {
            or al,bl            //*******
            mov dx,0x378        //*0x378 es la dirección de comienzo de
            out dx,al           //* I/O del puerto LPT1 en las PCs estándar.
            push ax             //*
            mov dx,0x37a        //*Revisa uno a uno los bits
            in al,dx            //*del byte a convertir, mediante
            mov dx,ax           //*operaciones lógicas.
            pop ax              //*
            test dl,1           //*
            jnz MAYOR           //*Luego devuelve el byte convertido
            xor al,bl           //*por AL.
            shr bl,1            //*
            loop s1             //*
            jmp FIN             //*
            };                  //*
MAYOR:                          //*
asm         {                   //*
            shr bl,1            //*
            loop s1             //*
            };                  //*
FIN:                            //*
asm         {                   //*
            or al,bl            //*
            pop cx              //*******
            pop bx
            ret
LEEBITS ENDP
        }

sale:
asm     {
        pop es
        pop ds
        popf
        popa
        };
};
```