CODIFICADORES DE VOZ RC, APLICACIÓN DEL CODIFICADOR RC-2 AL DESARROLLO DE UNA CALCULADORA PARLANTE

MARIANO BARRÓN RUIZ
Departamento de Ingeniería de Sistemas y Automática.
Universidad del País Vasco. España.

En este trabajo se describen dos compresores que codifican las señales de voz utilizando un bit por muestra; ambos codecs se caracterizan por: 1º) la escasa potencia de cálculo que precisan para codificar las señales de audio, y 2º) la sencillez del circuito de decodificación o reproducción de voz (una o dos salidas digitales seguidas de una red RC). El codec RC-2 proporciona una calidad de sonido aceptablemente buena con tasas de bit inferiores a los 13.500 bits al segundo, sin embargo, la calidad mejora notablemente cuando se incrementa la tasa de bit. Se demuestra que la novedosa técnica de codificación RC-2 proporciona mejores resultados que el conocido codificador CVSD (Continuously Variable Slope Delta) utilizado principalmente en aplicaciones militares durante más de 30 años. Un software de PC, desarrollado por el autor, facilita la codificación de ficheros de voz capturados con la Grabadora de Sonidos de Windows y genera ficheros BIN, ASM o C que pueden enlazarse fácilmente con el software de cualquier microcontrolador para crear nuevas aplicaciones que utilicen la voz como elemento de interfaz con el hombre. Finalmente, para verificar la utilidad del software desarrollado, se describe una calculadora parlante desarrollada con solo tres chips (un µC 8051, un amplificador de sonido y una memoria serie para almacenar la voz). La calculadora, que puede ser útil para persona ciegas, opera con números reales y dispone de las cuatro operaciones básicas y de las funciones de cambio de signo y raíz cuadrada.

1. Introducción

El micrófono es un dispositivo que transforma los cambios de presión de aire, originados por las fuentes de sonido, en un voltaje que varía continuamente en el tiempo, permitiendo obtener una representación analógica del sonido. La representación digital de un sonido se obtiene midiendo con suficiente periodicidad el voltaje generado por el micrófono, convirtiendo cada medida en un número, y escribiendo estos números en un fichero. Para transformar un voltaje en un número se utiliza un convertidor de analógico digital (ADC). La técnica que codifica una señal analógica como una secuencia de valores digitales se denomina Pulse Code Modulation (PCM). Para reproducir un sonido almacenado en formato digital se convierten los números en voltajes mediante un convertidor de digital a analógico (DAC), y se aplican a un altavoz, con la misma periodicidad que se utilizó en la captura. Las variaciones de voltaje producen movimientos en la membrana del altavoz que se transmiten por el aire como ondas de presión o sonidos.

Intuitivamente, resulta claro que la reproducción de un sonido será tanto mejor cuanto más elevado sea el número de muestras disponibles; es decir, cuanto mayor sea el tamaño del fichero de voz; sin embargo, éste razonamiento intuitivo no es totalmente cierto. El teorema de muestreo de Nyquist demuestra que, puede reconstruirse una señal analógica limitada en banda, a partir de sus muestras, si éstas se han obtenido con una frecuencia de muestreo superior al doble de la frecuencia más elevada presente en la señal. Para limitar el ancho de banda de las señales de audio se emplean filtros paso bajo.

La gama de frecuencias a las que el oído humano es capaz de responder depende de las personas y de la edad de las mismas, pero típicamente se extiende desde los 20 Hz hasta los 20.000 Hz. Por este

motivo, para digitalizar sonido de alta calidad, se utiliza una frecuencia de muestreo de 44.100 Hz, que resulta superior al doble de los 20.000 Hz. Las frecuencias generadas por la voz humana se encuentran comprendidas en un rango notablemente más bajo que la gama de frecuencias a las que responde el oído [1], de hecho, las señales de voz que se transmiten por canales telefónicos, tienen un ancho de banda inferior a los 4 KHz, y por esa razón se digitalizan utilizando velocidades de muestreo de 8 KHz. Los 4 KHz de ancho de banda de las señales de voz, proporcionan buena inteligibilidad y permiten reconocer al locutor. Cuando la fuente de sonido no es una persona, sino por ejemplo un conjunto de instrumentos musicales, el ancho de banda es más amplio, por lo que debe utilizarse una frecuencia de muestreo más alta para mantener la calidad del sonido.

2. Clases de codificadores de sonido

La codificación PCM del sonido exige una gran capacidad de almacenamiento. Para almacenar sonido de alta calidad se requieren 1,4 Mbits por segundo (2 canales •16 bits • 44.100 muestras/s); éste volumen se reduce a unos 128 kbits/s si se trata de voz humana (1 canal•16 bits•8.000 muestras/s). La necesidad de almacenar, reproducir, o transmitir sonido digital obliga a desarrollar codificadores de sonido que reduzcan el número de bits, sin que ello suponga una pérdida apreciable de calidad.

Atendiendo al tipo de codificación empleada, los codificadores de voz humana se clasifican en: **Codificadores de forma de onda, Vocoders, y Codificadores híbridos**. Los codificadores de forma de onda tratan de reproducir la forma de onda del sonido original con la mayor fidelidad posible, siendo capaces de reproducir voz de buena calidad con un volumen moderado-alto de bits (>16 Kbits/s). Los vocoders intentan conservar las propiedades espectrales del sonido, pudiendo generar voz inteligible que a veces suena algo sintética, a una tasa de bits mucho más reducida (entre 2 y 16 Kbits/s). Los codificadores híbridos utilizan técnicas de los dos anteriores y poseen características intermedias. Aunque los vocoders son los más eficientes en el tratamiento de la voz, los codificadores de forma de onda no se centran exclusivamente en el tratamiento del habla, y reproducen con fidelidad otros sonidos, tales como la música o el ruido de fondo.

La evaluación de la calidad de un codificador de voz no es una tarea sencilla, debido a que la calidad de la voz es un concepto subjetivo que está ligado a su percepción; no obstante, para medir el parecido entre el sonido original y el sonido sintetizado pueden utilizarse parámetros objetivos. Así, los vocoders utilizan medidas de distancia espectral, mientras que los codificadores de forma de onda usan la **relación señal ruido SNR** [2, 3]. Debido a que nuestros codificadores RC-1 y RC-2 son del tipo de forma de onda, utilizaremos como parámetro de comparación la SNR expresada en decibelios. Se define esta relación como:

$$\mathbf{SNR = 10 \cdot \log10 \left[\left(\Sigma (S(n))^2 \right) / \left(\Sigma (S(n) - V(n))^2 \right) \right]} \qquad \mathbf{(1)}$$

El numerador de la expresión (1) representa el cuadrado de la amplitud de las muestras de la señal original S(n), y el denominador representa el cuadrado del ruido o diferencia entre la señal original S(n) y la señal sintetizada V(n). El parecido entre la señal sintetizada y la señal original, será tanto mayor, cuanto mayor sea la relación SNR. Es conocido que la SNR pondera excesivamente las muestras de mayor amplitud con relación a las muestras más débiles; sin embargo, las muestras débiles tienen una gran importancia en la percepción del sonido, por este motivo, a veces se utiliza la **relación señal ruido segmental o SSNR** definida como la media aritmética de los valores SNR obtenidos para un frame o subconjunto de muestras. Típicamente cada frame representa de 10 a 20 ms de voz.

$$\mathbf{SSNR = (\Sigma\ SNRframe) / N} \qquad \mathbf{(2)}$$

Otros refinamientos en el cálculo de SSNR contemplan la exclusión de los valores extremos de SNR, y la eliminación de los frames de silencio. La elección de la longitud del frame y el resto de refinamientos en el cálculo de SSNR, no son otra cosa que un intento de reflejar dentro de un parámetro objetivo el concepto subjetivo de inteligibilidad de la voz.

3. Algunos codificadores de forma de onda

Los codificadores de forma de onda, o codificadores en el dominio del tiempo, intentan reproducir fielmente la forma de onda de la señal de sonido. Un ejemplo de este tipo de codificador lo constituye el codificador PCM, que puede reproducir voz humana a 128 kbits/s utilizando una frecuencia de muestreo y de reproducción de 8.000 Hz con una resolución de 16 bits. En muchos casos, una resolución de 12 bits es suficiente, con lo cual la tasa de bits se reduce a 96 kbits/s. Para lograr rebajar aún más la tasa de bits, algunos codificadores PCM usan un ADC provisto de un cuantificador no uniforme, que utiliza mayor resolución para niveles bajos de señal y menor resolución para niveles altos. Los sistemas de comunicación provistos de cuantificadores no uniformes, comprimen la señal en el lado del transmisor y la expanden en el lado del receptor para obtener una relación lineal. Existen dispositivos electrónicos, llamados companders, que se encargan de comprimir la señal a transmitir y expandir la señal recibida. Con un compander de 8-bits se obtiene aproximadamente la misma calidad de sonido que con un cuantificador uniforme de 12 bits. Así, un codificador PCM provisto de un cuantificador logarítmico, como el A-law o el μ -law, puede reproducir voz de calidad a 64 kbits/s.

Algunos codificadores de forma de onda sacan partido a la fuerte correlación existente entre muestras adyacentes de voz codificando la diferencia (delta) entre dos muestras, en lugar de la amplitud de las mismas. Llevando esta idea un paso más adelante el codificador puede utilizar como entrada la diferencia entre una muestra de señal y una estimación de su valor. La diferencia entre dos señales requiere menos niveles de cuantización y consecuentemente la tasa de bits se reduce. Estos codificadores se denominan Diferencial PCM (DPCM). Si se combina la codificación adaptativa con la DPCM se obtiene un nuevo algoritmo de compresión llamado adaptativo DPCM (ADPCM), capaz de reproducir voz de calidad a 32 kbits/s. La Delta Modulation (DM) y la Continuosly Variable Slope Delta modulation (CVSD) son técnicas de codificación diferencial que utilizan un solo bit para codificar la diferencia entre dos muestras sucesivas de voz. El codificador CSVD es básicamente un codificador DM con un cuantizador adaptativo. Los codificadores CVSD se utilizan en comunicaciones estratégicas que opcionalmente incluyen técnicas de seguridad o cifrado. Normalmente utilizan tasas de transmisión de 16 y 32 kbits/s.

Los dos codificadores presentados en este artículo también codifican la diferencia entre dos muestras sucesivas de sonido utilizando un bit. El codificador RC-1 es muy similar al DM y el codificador RC-2 es parecido al CVSD; se diferencian, entre otras cosas, en el integrador utilizado. El integrador de los codificadores RC-1 y RC-2 está formado por una resistencia (o dos resistencias) y un condensador. El codificador RC-2 proporciona una calidad de sonido aceptablemente buena a unos 13,5 kbits/s; incluso a la mitad de esta tasa, el sonido generado es perfectamente entendible.

4. Codec RC-1

Es un codec de la clase de codificadores de forma de onda que se caracteriza por reproducir el sonido usando una patilla de un microcontrolador seguida de una red RC, véase la Figura 1. Para reproducir el sonido utiliza una frecuencia de actualización de la patilla de salida del microcontrolador igual a la frecuencia de muestreo del sonido original. El codificador RC-1 consigue un factor de compresión 16, siempre que la captura del sonido se realice utilizando muestras de 16 bits, ya que en la reproducción del mismo solo se utiliza un bit. La respuesta del circuito RC de la figura 1, a una entrada en escalón aplicada en la patilla P1 del microcontrolador, viene dada por la expresión:

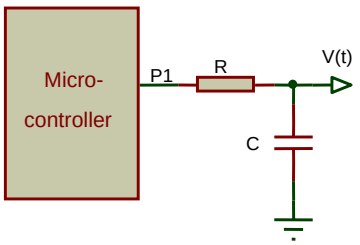

Figura 1. Decodificador RC-1.

$$V(t) = Vfinal + (Vinicial - Vfinal) \cdot e^{-t/RC} \qquad (3)$$

Donde Vfinal será 0V, si la patilla del microcontrolador se encuentra a nivel bajo, o Vcc si la misma se encuentra a nivel alto. Suponiendo que el microcontrolador actualice la patilla de salida a la misma velocidad que se utilizó para tomar las muestras de la señal original, es decir, con un periodo de

muestreo T; la relación entre dos muestras consecutivas V(n+1) y V(n) vendrá dada por:

$$V(n+1) = Vfinal + (V(n) - Vfinal) \cdot e^{-T/RC} \qquad (4)$$

Supongamos que ya se dispone del sonido que se desea codificar, y que viene representado por un conjunto de muestras de 16 bits S(n), con n=1..m, adquiridas utilizando un periodo de muestreo T. La codificación del sonido consiste en encontrar una secuencia de valores de 1 bit, P(n), con n=1..m, que actualicen la patilla P1 del microcontrolador cada T segundos, produciendo una señal en el condensador C, lo más parecida posible al sonido inicial. Para codificar el sonido S(n) supongamos que inicialmente la patilla P1 del microcontrolador se encuentra a nivel bajo, con lo que el condensador estará descargado, siendo el valor inicial de la tensión en el condensador V(0) = 0; supongamos también que se ha fijado un valor inicial para los componentes R y C, por ejemplo R = 6k y C = 0.1 µF. Se utilizará la fórmula (4), con n = 0, para calcular la tensión del condensador V(1) en el instante de tiempo T. Se realizarán dos cálculos de V(1) para contemplar las dos posibilidades: que el nivel de salida de la patilla del microcontrolador sea: bajo (Vfinal = 0 V), o alto (Vfinal = Vcc). Elegiremos el valor de V(1) que se encuentre más próximo al valor de la primera muestra de sonido S(1), obteniendo para P(1) un valor 0 si se utilizó Vfinal = 0 V, o un valor 1 si se usó Vfinal = Vcc.

Una vez conocido el valor de V(1), se utilizará de nuevo la fórmula (4), con n = 1, y se realizarán dos nuevas previsiones para V(2). Nos quedaremos con el valor de V(2) más próximo a la muestra del sonido S(2), obteniendo el dígito binario P(2). El proceso continuará hasta tratar todas las muestras del sonido a codificar. El procedimiento anterior puede repetirse para diferentes valores de la constante de tiempo RC, hasta determinar el valor de RC que maximice el parámetro utilizado (SNR o SSNR) para medir el parecido entre la señal sintetizada V(n) y el sonido original S(n). Para modificar la constante RC se puede dejar constante el valor de C y modificar sólo el valor de R.

Si se elige un valor de R demasiado grande, el incremento de tensión entre muestras consecutivas será pequeño, la forma de onda sintetizada no podrá seguir los cambios rápidos del sonido original y se tendrá **ruido de rebasamiento de pendiente**. Por el contrario si se elige un valor de R muy pequeño, el cambio entre muestras consecutivas de la señal sintetizada será grande y se tendrá un **ruido granular** excesivo en los tramos de silencio.

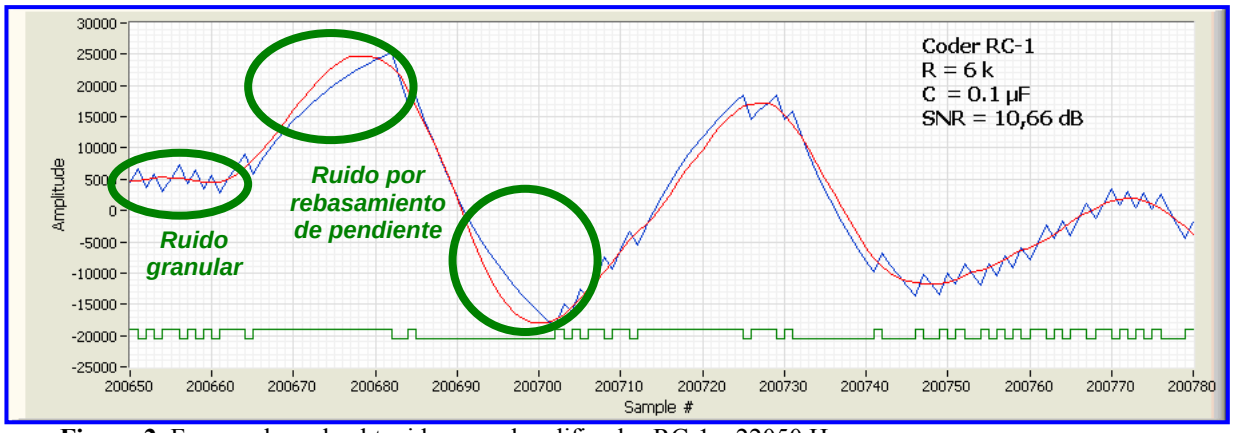

Figura 2. Formas de onda obtenidas por el codificador RC-1 a 22050 Hz.

El software de PC desarrollado, realiza el proceso descrito anteriormente de forma muy eficaz. Trabajando incluso con grandes ficheros de voz, proporciona con rapidez resultados como los mostrados en la figura 2. La onda de color rojo es la señal S(n) de sonido original; la de color azul es la señal V(n) sintetizada por el circuito de la figura 1, y en la parte inferior se muestra, en color verde, la información digital P(n) a generar por la patilla P1 del µC. Puede apreciarse que las curvas de carga y descarga del condensador tienden hacia Vcc (Amplitud = +32.767), o hacia 0 (Amplitud = -32.768), según que el nivel lógico de la patilla P1 sea alto o bajo, respectivamente. En los tramos donde la señal S(n) cambia lentamente se aprecia el ruido granular de la señal V(n), y en los tramos de variación rápida se observa el ruido debido a rebasamiento de pendiente.

5. Codec RC-2

El codec RC-2 es muy parecido al anterior, pero mejora la calidad del sonido con un incremento de costo casi nulo. Para reproducir el sonido utiliza dos patillas de un microcontrolador y un filtro RC, véase la Figura 3. La frecuencia de actualización de las dos patillas de salida del microcontrolador se mantiene igual a la frecuencia de muestreo del sonido original. Pese a utilizar dos patillas, el tamaño del fichero necesario para generar el sonido es el mismo que en el codificador anterior, debido a que el micro-controlador pone siempre su patilla P2 al mismo estado lógico que tuvo la patilla P1en la actualización anterior; es decir, el mismo flujo de datos de salida de la patilla P1, se tiene en la patilla P2, con una demora de T segundos ($T = 1/f_{MUESTREO}$).

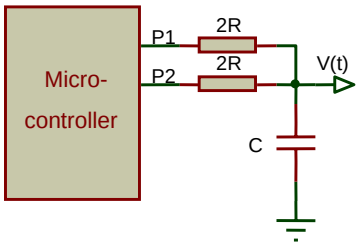

Figura 3. Decodificador RC-2.

La respuesta del circuito RC, de la figura 3 puede determinarse mediante la ecuación (3), pero ahora, Vfinal podrá tomar los valores: Vcc, 0V, o Vcc/2, dependiendo de que las patillas de microcontrolador P1 y P2 se encuentren: ambas a nivel alto, ambas a nivel bajo, o que sus niveles lógicos sean opuestos. Como la resistencia equivalente Thevenin de 2R y 2R es R, la ecuaciones (3) y (4) seguirán siendo válidas. El codificador RC-2 ofrece tres posibles valores para Vfinal, en lugar de los dos valores disponibles con el codificador RC-1, lo cual le permite adaptarse mejor a cualquier forma de onda. En particular, la representación analógica de los silencios, que consiste en un valor constante Vcc/2, puede obtenerse con el codificador RC-2 sin ningún ruido granular.

Llamaremos S(n) al valor de la muestra n del sonido original, V(n) al valor de la muestra n del sonido sintetizado, P2(n) y P1(n) a los valores lógicos de las patillas P2 y P1 en el periodo n. Para la codificación del sonido supongamos que las salidas del microcontrolador se encuentran inicialmente en niveles opuestos: P2(0) a nivel bajo y P1(0) a nivel alto, con lo que el condensador estará cargado a Vcc/2, es decir V(0) = Vcc/2; supongamos también que se ha fijado un valor inicial para los componentes R y C, por ejemplo R = 6k y C = 0.1 µF. Se utilizará la fórmula (4), con n = 0, para calcular la tensión del condensador en el instante de tiempo T. Debemos tener en cuenta que P2(1) tomará el valor que antes tenía P1 (nivel alto), mientras que P1(1) podrá tomar valor alto o bajo. Se realizarán dos cálculos de V(1), para contemplar las dos posibilidades: que el nivel de salida de la patilla P1(1) sea bajo (Vfinal = Vcc/2), o alto (Vfinal = Vcc). Nos quedaremos con el valor de V(1) que se encuentre más próximo al valor de la primera muestra de sonido S(1), obteniendo para P(1) un valor L si se utilizó Vfinal = Vcc/2, o un valor H si se utilizó Vfinal = Vcc.

Una vez conocido el valor de V(1), se utilizará de nuevo la fórmula (4), con n=1. La patilla P2 tomará el valor anterior de P1; la patilla P1 podrá tomar valor alto o bajo; por ello, se realizarán dos nuevas previsiones para V(2), y nos quedaremos con el valor más próximo a la muestra del sonido a codificar S(2), obteniendo el dígito binario P(2). El proceso continuará hasta tratar todas las muestras del sonido a codificar.

Debido a que la patilla P2 toma siempre el valor anterior de la patilla P1, es fácil deducir que si en un determinado instante de tiempo P1 tomó valor Vcc, en la siguiente previsión V(t) podrá tender hacia Vcc o hacia Vcc/2, pero nunca hacia 0. De igual forma, si P1 tomó valor 0, en la siguiente previsión V(t) podrá tender hacia Vcc/2 o hacia 0, pero nunca hacia Vcc. Así pues, el valor lógico de la patilla P1 influye en el valor de salida actual del circuito RC-2 y en el valor que la salida tomará en la siguiente muestra; por este motivo, se puede obtener una mejor aproximación si (como realiza nuestro software) se realizan previsiones teniendo en cuenta, no solo una, sino las dos siguientes muestras del sonido original. Con el algoritmo descrito, el software de PC proporciona los resultados mostrados en la Figura 4, en la que puede apreciarse en color rojo la señal S(n), correspondiente al sonido original; en azul la señal V(n) sintetizada por el codificador de 2 bits, y en verde la información binaria, P1(n) y P2(n), generada por las patillas P1 y P2 del microcontrolador.

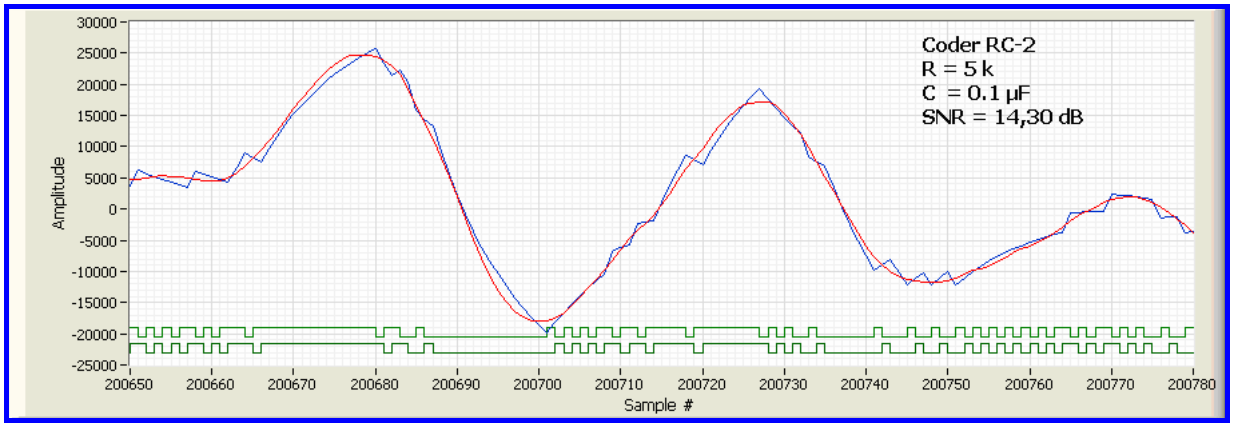

Figura 4. Formas de onda obtenidas por el codificador RC-2 a 22050 Hz.

Puede apreciarse que las curvas de carga y descarga del condensador tienden hacia Vcc (Amplitud = +32.767), hacia Vcc/2 (Amplitud = 0), o hacia 0 (Amplitud = -32.768). En este caso se obtiene una valor máximo de SNR = 14,3038 dB, con R=5k. Al comparar estas señales con las de la figura 2 se aprecia una notable disminución del ruido granular y del ruido de rebasamiento de pendiente, a la vez que se mejora la SNR en más de 3dB.

6. Compactación RLE de los ficheros de voz codificados

Los codificadores de sonido RC-1 y RC-2, producen un conjunto de valores binarios P(n) para actualizar la patilla (o las dos patillas) de salida del microcontrolador y generar el sonido sintetizado. Si el sonido original S(n) se capturó utilizando muestras de 16 bits a razón de 22.050 muestras por segundo, se tendrá una señal P(n) formada por muestras de 1 bit que servirán para generar el sonido sintetizado a la velocidad de 22.050 muestras por segundo. En el supuesto anterior, los codificadores RC-1 y RC-2 proporcionan una reducción del tamaño del fichero original de 16 veces. Para almacenar de forma efectiva la señal binaria P(n) se agruparán 8 muestras sucesivas de P(n) en un byte y se guardarán en el disco duro del PC. Si observamos detenidamente la estructura de un fichero de datos que contenga la señal P(n) se descubre rápidamente la presencia de series de bytes con los valores 0xAA y 0x55, que representan silencios o tramos en los que la amplitud de la señal apenas varía. La técnica de compresión de datos que reemplaza "n" apariciones consecutivas del byte "b" por el par "bn" se conoce como "run length encoding" o RLE [1]. Si realizamos histogramas de los ficheros de voz codificada, se confirma la presencia masiva de los bytes 0xAA y 0x55, y la ausencia casi total de bytes tales como 0xF9, 0xF3, 0xE7, …, 0x0C, 0x06. Esta información nos permite diseñar la estrategia de compactación mostrada en la tabla 1.

Situación	Cambiarlo por:	Efecto
Un solo byte AA	AA	Neutro
Un AA seguido de nn AA's	F9 nn, con nn = 01..FF	Compresión si nn >1
Un solo byte 55	55	Neutro
Un 55 seguido de nn 55's	F3 nn, con nn = 01..FF	Compresión si nn > 1
Un F9	F9 00	Expansión
Un F3	F3 00	Expansión

Tabla 1. Estrategia de compresión RLE de ficheros de voz con codificación RC-1 y RC-2.

La estrategia descrita, no comprime ni expande un solo AA o una pareja AA AA; también resulta neutra ante las apariciones de un solo 55 o una pareja 55 55. Comprime si encuentra más de dos AA seguidos, o más de dos 55 seguidos, y expande cada vez que encuentra un F9 o un F3. La experiencia práctica demuestra que al aplicar esta estrategia de compactación a ficheros de voz codificados, se logra reducir su tamaño hasta un 50-70% del tamaño inicial. Si los ficheros de voz se muestrean a 22.050 muestras al segundo, la mayoría de las veces se comprime por debajo del 62% del tamaño inicial, logrando de esta manera reducir la velocidad de reproducción a 0,62•22.050 = 13.671 bits/s.

7. Software de codificación del sonido

El software de codificación de sonido es una aplicación para PC desarrollada en LabVIEW [4], que recibe como entrada un fichero de sonido WAV, lo codifica en formato RC-1 o RC-2, opcionalmente lo comprime utilizando la técnica RLE descrita en el apartado anterior, y permite salvar el fichero codificado en formato binario, en lenguaje ensamblador o en lenguaje C, para que pueda reproducirse con cualquier microcontrolador. El fichero ejecutable "RC Sound Encoder" es de libre disposición y puede descargarse desde [5].

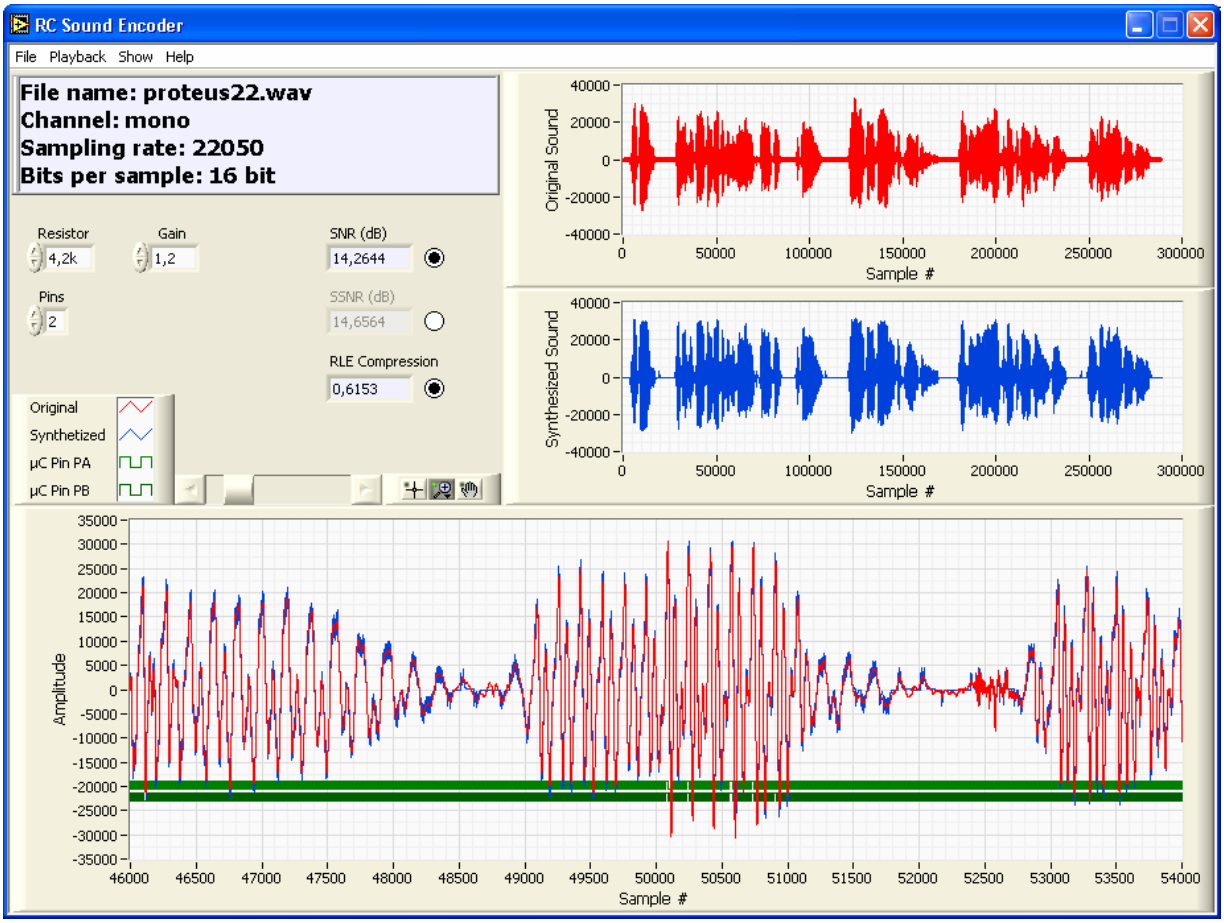

Figura 5. Formas de onda obtenidas por el codificador RC-2 a 22050 Hz.

El software, véase la Figura 5, presenta al usuario tres visualizadores: el primero, situado en la parte superior muestra la representación gráfica del sonido original; el segundo, situado en la zona central, muestra el sonido sintetizado; y el tercero, situado en la zona inferior, muestra el sonido original, el sonido sintetizado y la información digital que debe sacarse por una o dos patillas del microcontrolador para generar el sonido. El primer visualizador solo cambia cuando se abre un nuevo fichero WAV a procesar, los otros dos visualizadores cambian, cada vez que el usuario modifica los controles: Resistor, Pins y Gain que influyen en el resultado de la codificación. El tercer visualizador dispone de una barra de scroll y de herramientas para aplicar zoom sobre cualquier región. Para tratar un fichero de sonido es necesario disponer del mismo. Podemos obtener nuestros propios ficheros de sonido utilizando la Grabadora de Sonidos de Windows. Se consiguen buenos resultados salvando los ficheros WAV en formato PCM, 22.050 Hz (1/2 CD Rate), 16 bits, mono. Aún mejor resultado se obtiene con frecuencias de muestreo más elevadas, por ejemplo 44.100 Hz (Basic CD Standard), pero ello supone duplicar el tamaño de los ficheros y la tasa de bits a sacar por las patillas del µC. El programa de PC permite tratar ficheros WAV muestreados a 8.000, 11.025, 22.050 y 44.100 Hz.

Una vez que dispongamos del fichero o ficheros WAV, se lanzará la aplicación "RC Sound Encoder" y tras elegir la opción de menú "File – Open WAV File"; aparecerá una caja de diálogo para

que el usuario seleccione el fichero que desea procesar. Después de realizar la selección, se codificará el sonido utilizando los valores por omisión de los controles (Resistor = 5k, Gain = 1 y Pins = 2), o sus valores anteriores, y se refrescarán los tres visualizadores gráficos. El usuario puede interactuar con el programa modificando el valor de estos controles y observando la respuesta del codificador. Consideramos que el sonido sintetizado y el sonido original son más parecidos, cuanto mayor sea la relación señal ruido, es decir cuanto mayor sea el valor que muestra el visualizador digital SNR (dB). El programa también proporciona la relación señal ruido segmental en el visualizador SSNR (dB), sin embargo no es conveniente trabajar con los parámetros SNR y SSNR a la vez, ya que en general, presentan sus valores máximos para diferentes valores del control Resistor, y ello genera confusión en el usuario. Es preferible que la persona que utilice el programa, tras un número suficiente de pruebas, elija utilizar como parámetro de comparación uno de ellos, el que le resulte más satisfactorio, y que desmarque el Round Radio Button del parámetro desechado, para que se muestre atenuado y no le provoque distracción. El programa calcula el valor SSNR promediando los valores SNR de frames o subconjuntos de 400 muestras, lo que representa unos 18 ms a la frecuencia de muestreo de 22.050 Hz. Además en el cálculo de SSNR se rechazan los frames cuyo valor SNR es inferior a los 3 dB, eliminándose de esta forma los tramos de silencio.

Ya se ha señalado anteriormente que la inteligibilidad de la voz es un concepto subjetivo; por este motivo, el software permite reproducir en todo momento el sonido original y el sonido sintetizado, facilitando al usuario el logro de la codificación óptima. La reproducción de los sonidos se obtiene con las opciones de menú “Playback – Original Sound” y “Playback – Unfiltered Synthetized Sound”. Aunque el software permite codificar los sonidos en los formatos RC-1 y RC-2, se aconseja utilizar siempre el segundo formato, debido a la mejora de calidad de sonido que proporciona con el costo adicional de una resistencia y otra patilla del microcontrolador. La selección del tipo de codificación se realiza con el control Pins.

El control Gain, sirve para atenuar o amplificar la muestra de sonido original, con el objeto de ecualizar diferentes ficheros WAV. El valor del parámetro SNR obtenido por los codificadores RC-1 y RC-2 depende del valor de la constante de tiempo RC del circuito de reproducción y también de la amplitud de la señal tratada. Si se modifica el control Gain manteniendo el valor del control Resistor, también variará el valor de los parámetros SNR y SSNR. Si hay que reproducir varios ficheros WAV a 22050 Hz con un μC, conviene fijar para el control Resistor un valor próximo a 5k, y actuar sobre el control Gain para tratar de obtener el valor máximo de SNR para cada fichero WAV. La opción de menú “Show – Playback hardware”, calcula y muestra el circuito necesario para generar el sonido con un microcontrolador. Cuando el usuario del programa está seguro de que la codificación lograda es la mejor para su aplicación, dispone de las opciones de menú: “File – Save BIN File”, “File – Save ASM File” y “File – Save C File”, para salvar el fichero de voz en formato binario, en lenguaje ensamblador o en lenguaje C. En los tres casos, los dos primeros bytes del fichero indican el número de datos que les siguen; estos ficheros pueden salvarse con o sin compresión RLE, dependiendo del estado del Radio Button situado a la derecha del visualizador RLE Compression, véase Figura 5. Lógicamente, en la mayoría de los casos interesa aprovechar la compresión RLE para minimizar el tamaño de los ficheros de voz, ya que ello no complica excesivamente el software de reproducción de sonido.

8. Codificador RC-2 versus codificador CVSD

Los codificadores RC-1 y RC-2 pertenecen a la clase de los codificadores de forma de onda diferenciales de un bit, también llamados moduladores delta (DM). En principio puede pensarse que el codificador RC-2 no es de este tipo ya para generar sonido utiliza dos bits; sin embargo, el valor del segundo bit en cualquier instante de tiempo t, es igual al valor que tuvo el primer bit en el instante de tiempo t-T, siendo T el periodo de actualización de las patillas de salida del microcontrolador en la reproducción del sonido codificado, el cual coincide con periodo de muestreo de la señal original. La superioridad del codificador RC-2 sobre el codificador RC-1 ha quedado manifestada en el apartado 5, pero sería interesante comparar el comportamiento del primero frente a otro codificador del mismo tipo, tal como el codificador CVSD (Continuously Variable Slope Delta).

El codificador CVSD se ha venido usando principalmente en aplicaciones militares durante más 30 años; recientemente ha sido adoptado por Bluetooth [6] debido a su comportamiento robusto frente a errores de transmisión. La salida digital del codificador CSVD consiste en un bit por muestra; el bit transmitido se utiliza para indicar la pendiente de la señal, los 1's y 0's transmitidos se integran para obtener rampas ascendentes o descendentes que tratan de aproximarse a la señal original. Para que el codificador pueda seguir a señales que crecen o decrecen a distintas velocidades va equipado con una lógica de detección de pendiente, que tiene en cuenta los tres (o cuatro) últimos bits transmitidos o recibidos. Si estos tres bits son todos 1's o todos 0's, se aumenta la ganancia del integrador para que la rampa de subida o de bajada incremente su desplazamiento o paso. Si no se da la circunstancia anterior, la ganancia del integrador se reduce.

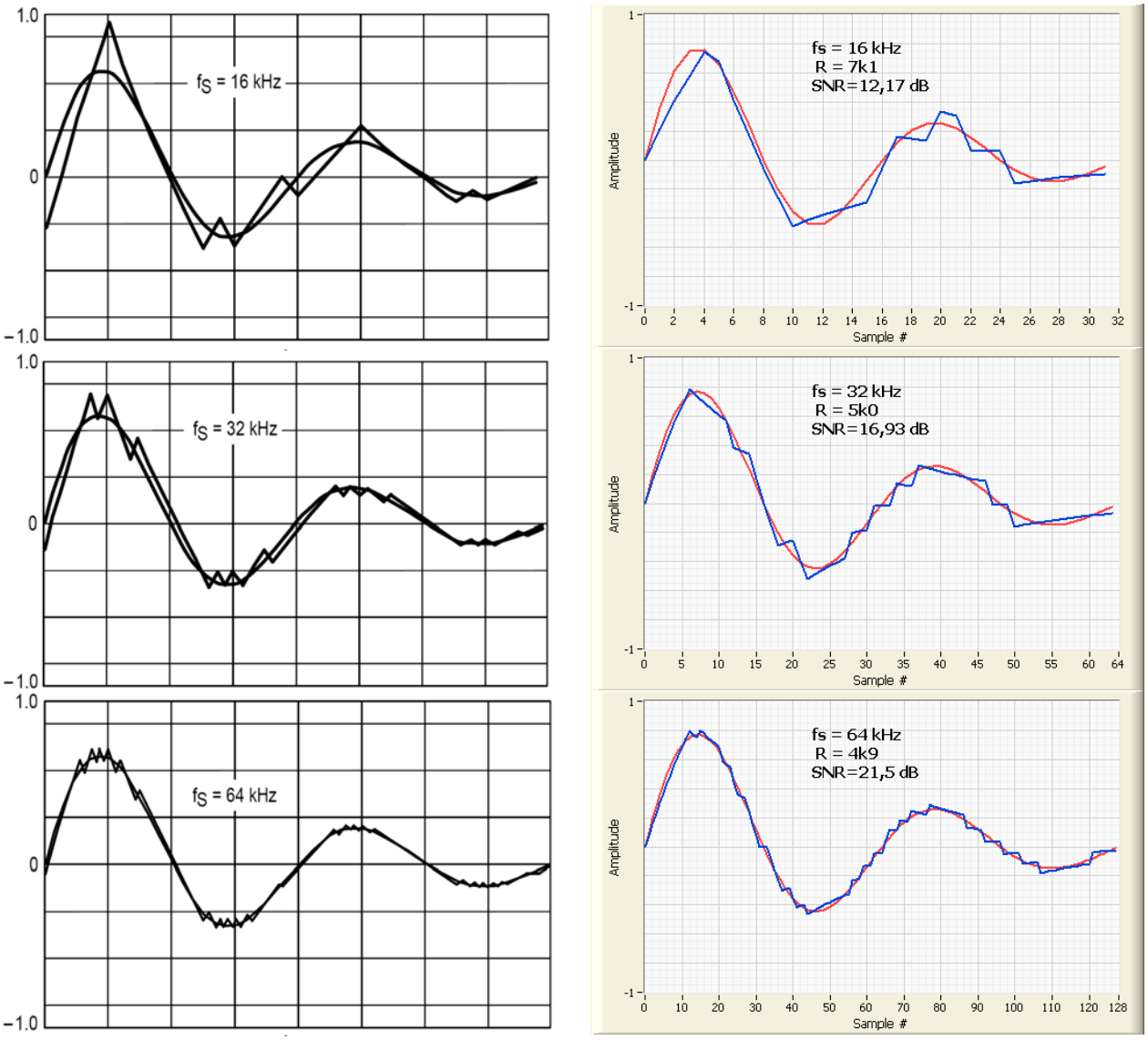

Figura 6. Señales de salida del codificador CSVD (izquierda) y del codificador RC-2 (derecha) para una señal de entrada de 1.000 Hz muestreada a 16, 32 y 64 kHz. Las gráficas de la columna de la izquierda se han obtenido de la nota de aplicación AN1544/D [7] de Motorola y corresponden a su modulador MC34115

Las figuras 6 y 7 muestran los resultados obtenidos por el modulador CVSD MC34115 de Motorola, y por el codificador RC-2 en condiciones de trabajo similares. Tras observar los resultados de ambos codificadores se puede afirmar que el comportamiento del codificador RC-2 es superior al del MC34115, y le domina claramente en los tramos de silencio.

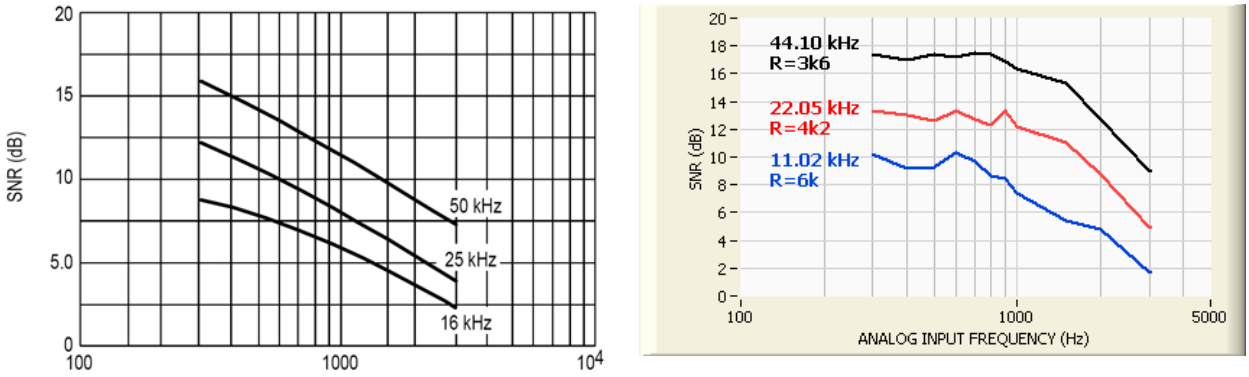

Figura 7. SNR (dB) versus input frequency (Hz) para el MC34115 (izqda) y para el codificador RC-2 (derecha).

9. Codificador RC-2 y señales de tipo no-voz

Se utiliza el término no-voz para referirse a todos los sonidos de origen natural o artificial distintos a la voz humana. Son señales no-voz los ruidos producidos por el viento o el agua, la música, los sonidos generados por: máquinas, animales, colisiones de objetos, etc. También pueden considerarse del tipo no-voz las señales Dual Tone Multi Frequency (DTMF), cuya importancia es bien conocida en telefonía. Las señales DTMF utilizan ocho frecuencias reunidas en dos grupos. El grupo de frecuencias bajas está formado por las frecuencias de: 697, 770, 852 y 941 Hz. El grupo de frecuencias altas lo forman las frecuencias de: 1209, 1336, 1447 y 1633 Hz. Un tono DTMF válido, está formado por una frecuencia de cada grupo, existiendo por lo tanto 16 tonos DTMF.

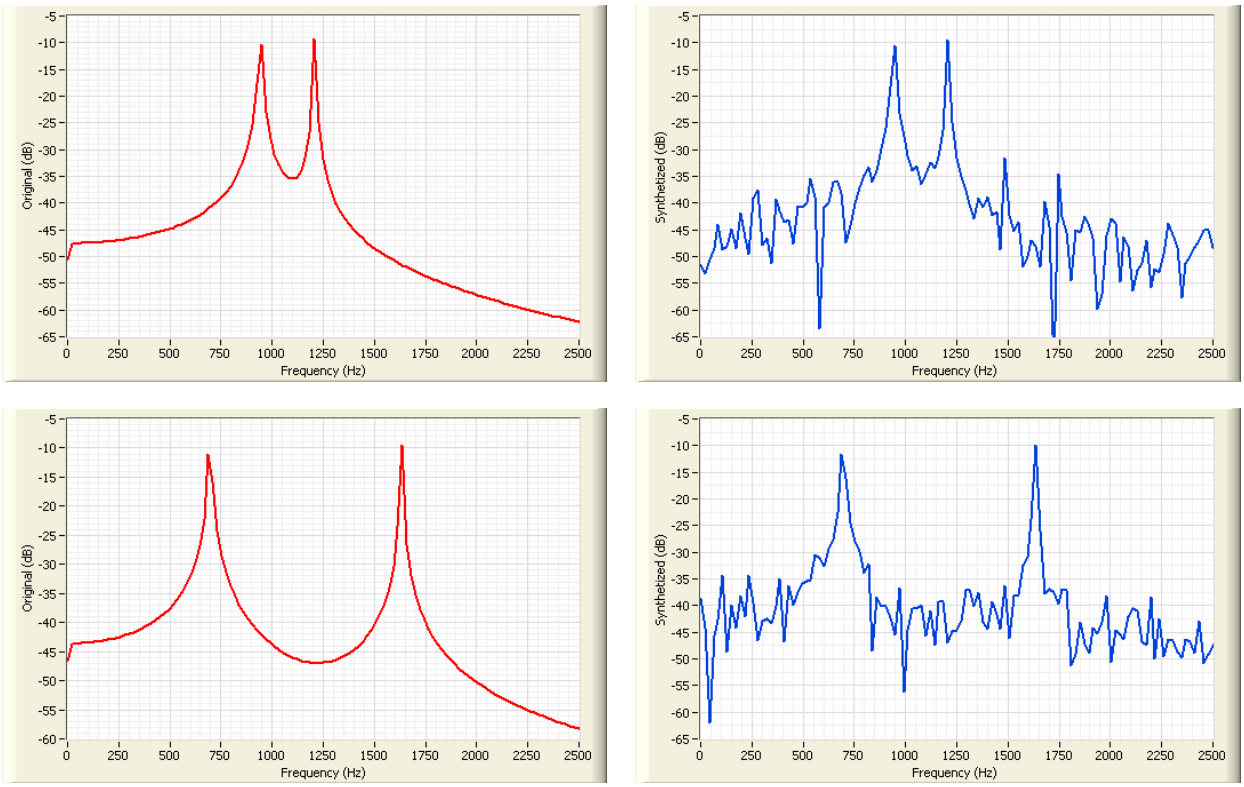

Figura 8. Espectro de las señales de entrada y de salida del codificador RC-2 (R=4k2, C=0.1µF, fs =22.050 Hz) para los tonos DTMF de frecuencias más próximas (941 y 1.209 Hz), y más alejadas (697 y 1.633 Hz).

Los codificadores de forma de onda, tales como el codificador RC-2, se caracterizan por reproducir con fidelidad los sonidos no-voz. La figura 8 representa el comportamiento del codificador RC-2 a 22.050 Hz frente a señales DTMF, mostrando el espectro de la señal original y de la señal sintetizada para los tonos DTMF de frecuencias más próximas (941 y 1.209 Hz), y más alejadas (697 y 1.633 Hz).

10. Análisis del ruido introducido por el codificador RC-2

En las formas de onda recogidas en las gráficas anteriores puede observarse que la señal de salida del codificador RC-2 se parece a la señal de entrada, pero no coincide exactamente con ella. La diferencia entre la señal de entrada y la señal de salida del codificador representa el ruido de cuantización. Puede observarse el efecto de la distorsión realizando el análisis de Fourier de las señales de entrada y salida al codificador y observando los resultados. En la figura 9 se muestra el espectro de la señal de entrada y de la señal de salida de un codificador RC-2, con frecuencia de muestreo de 22.050 Hz, cuando la señal de entrada es una función senoidal de 1.000 Hz. El efecto de la distorsión se traduce en la aparición en la señal de salida de los armónicos impares de la señal de entrada.

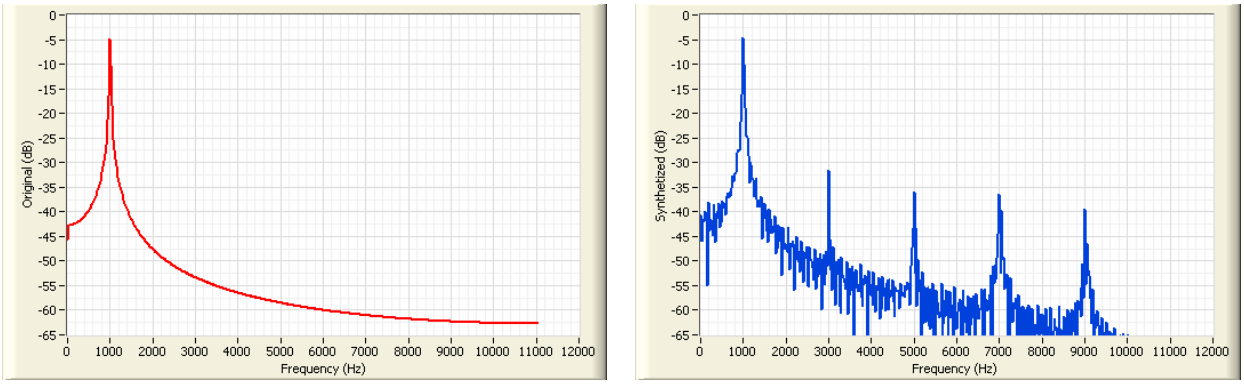

Figura 9. Espectro de las señales de entrada y de salida del codificador RC-2 (R=4k2, C=0.1µF, fs =22.050 Hz) correspondientes a la señal V0•sin(2•π•f•t), con V0 = 1V, f = 1.000Hz. Puede apreciarse la aparición de los armónicos impares representada por líneas verticales para frecuencias 3.000, 5.000, 7.000 y 9.000 Hz.

Los armónicos impares representan el ruido de cuantización o distorsión introducida por el codificador RC-2. Una sencilla red RC, como la que aparece en la figura 10 con R8=10k y C11=10nF, reduce el ruido de cuantización mejorando los resultados de salida del codificador RC-2. Para obtener mejores resultados debe utilizarse un filtro paso bajo de mayor orden. Téngase en cuenta que todas las gráficas que aparecen en este trabajo se han obtenido sin la ayuda de ningún filtro, y reflejan por lo tanto la salida no filtrada del codificador RC-2.

11. Descripción del hardware de la calculadora parlante.

Como ejemplo de uso del software descrito en apartados anteriores se ha realizado una calculadora parlante con un microcontrolador de prestaciones muy modestas pero que resultan suficientes para sintetizar la voz y para proporcionar los resultados de las operaciones aritméticas sin demoras. En los sucesivos apartados se describe el hardware y software de la calculadora así como el método utilizado para obtener los ficheros de voz y para codificarlos con el software de PC del apartado 7.

El principal circuito de la calculadora lo constituye el microcontrolador AT89C52 de la familia 8051 provisto de un cristal de 11,0592 MHz, véase figura 10. Como elemento de visualización dispone de un LCD alfanumérico de una línea de 16 caracteres, aunque no se trata de un elemento imprescindible ya que la calculadora puede hablar e informar al usuario de la entrada de los operandos, de los operadores, y de la salida de los resultados. Se ha dispuesto un teclado matricial de 20 teclas para: los 10 dígitos decimales; los cuatro operadores básicos “+”, “-”, “*”, y “/”; la coma decimal “,”, la tecla de cambio de signo “+/-”, la tecla “=” para obtener los resultados; la tecla de borrado; la tecla de función raíz cuadrada; y la tecla para seleccionar el modo de trabajo silencioso o con voz. El modo de trabajo silencioso es más rápido que el modo de trabajo con voz, ya que en este último cada vez que se obtiene un resultado se visualiza en el LCD dígito a dígito, dando tiempo a que la calculadora pronuncie cada carácter antes de mostrar el siguiente. Los ficheros de voz utilizados por la calculadora se encuentran almacenados en una memoria serie EEPROM codificados en formato RC-2 y se reproducen a razón de 22.050 muestras (de un solo bit) por cada segundo de voz.

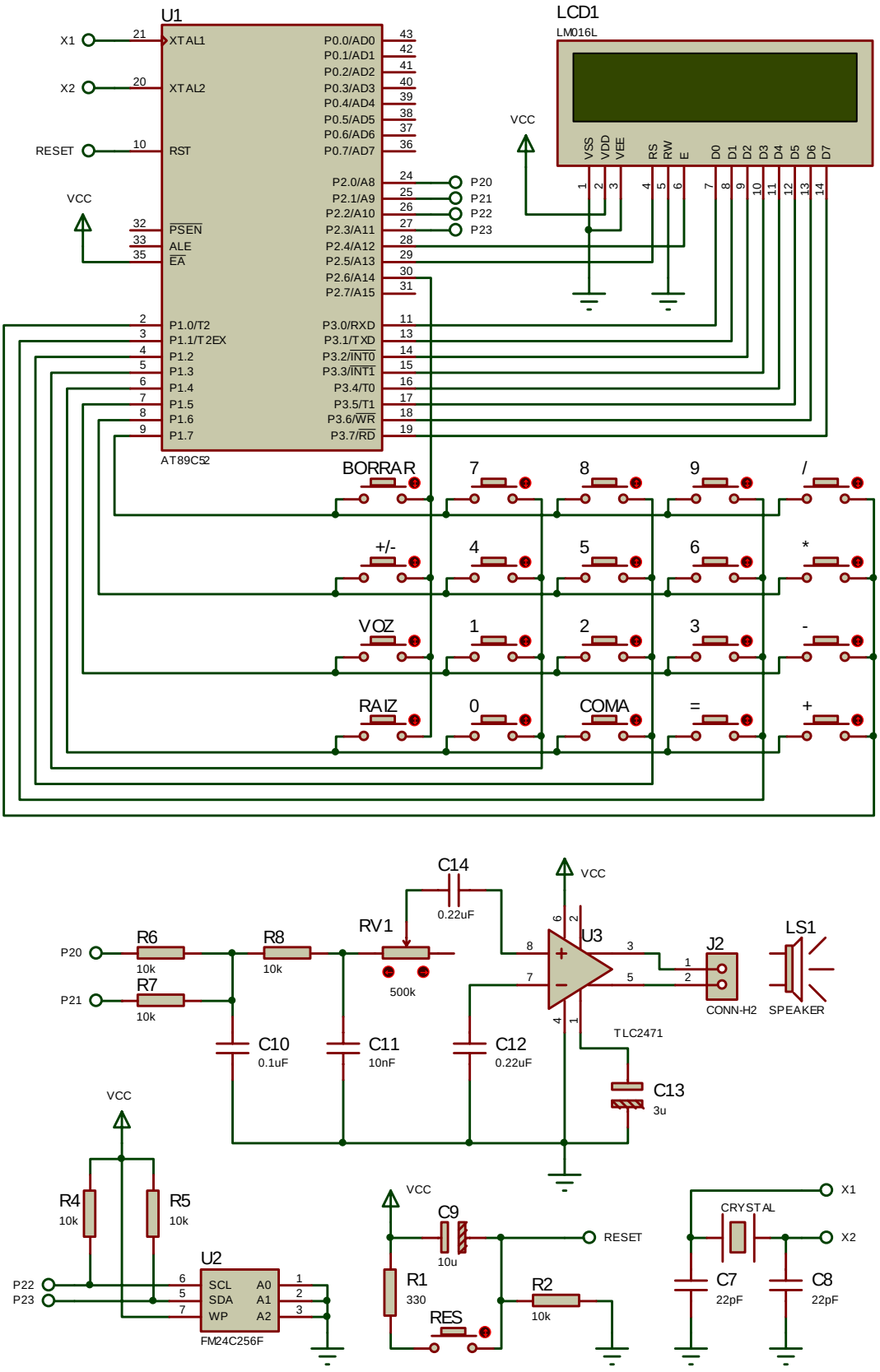

Figura 10. Hardware de la calculadora parlante: Microcontrolador, LCD, Teclado, Circuito decodificador de voz con amplificador y altavoz, Memoria serie para la voz, Circuito de Reset y Circuito del cristal.

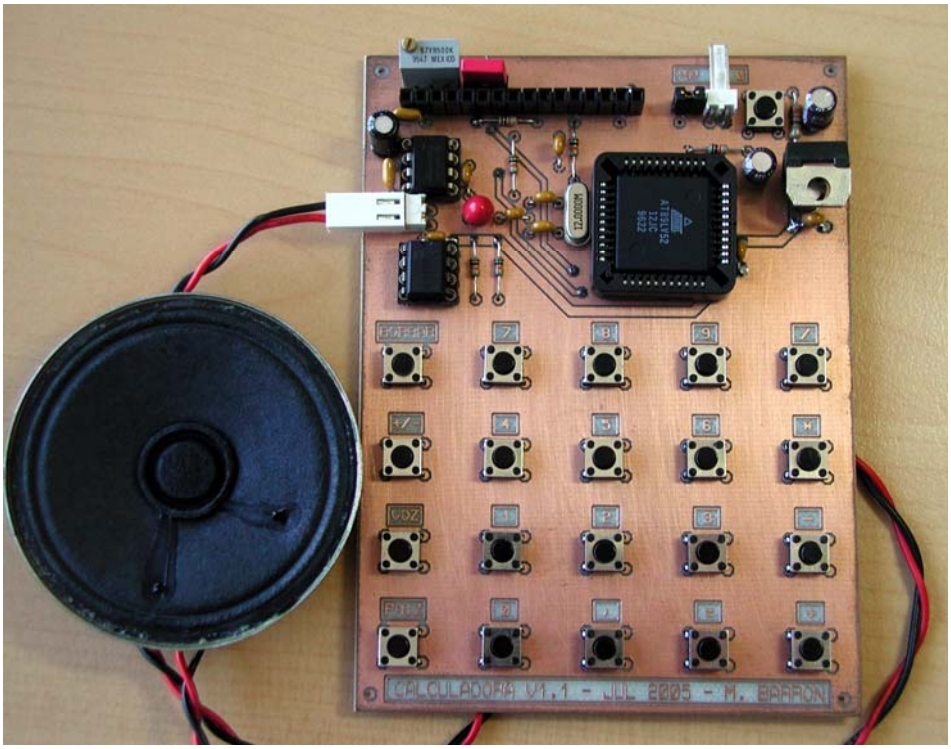

Figura 11. Fotografía de la calculadora desprovista del LCD.

12. Software para la reproducción de voz.

Para generar sonido se utilizan las patillas P2.0 y P2.1 del microcontrolador; una rutina de interrupción se encarga de actualizar estas patillas con la periodicidad requerida. Debemos tener en cuenta que un 8051 clásico trabajando a la frecuencia indicada, tiene unas prestaciones inferiores a 1 MIP. Para ser más precisos, solo puede ejecutar 921.600 ciclos máquina al segundo, lo cual implica que la rutina de interrupción no puede demorarse más de 41 ciclos máquina (921.600 / 22.050). Además el programa debe disponer de tiempo para leer desde memoria el fichero de voz y para realizar la descompresión RLE al mismo tiempo que se genera el sonido. El diseño del software ha tenido en cuenta esta limitación, y se ha conseguido una rutina de interrupción con una duración promedio de 21 ciclos máquina, y una duración máxima de 32 ciclos máquina. Aunque en el hardware mostrado en la figura 10 el fichero de voz reside en una memoria EEPROM serie, en el listado que sigue se muestra el software para la reproducción del fichero de voz FRASE1 residente en la memoria FLASH del microcontrolador, ya que resulta más sencillo de entender.

El decodificador RC-2 utiliza dos patillas de salida digitales para generar sonido pero ya se ha comentado que se usan muestras de un bit, lo que sucede es que la muestra de un bit actualiza una patilla a razón de 22050 muestras por segundo, y la segunda patilla toma siempre el valor lógico que tenía la primera patilla en el refresco anterior. Con objeto de obtener un código más rápido, en el listado que sigue no se procede así, en su lugar las sucesivas muestras de 1 bit que constituyen el fichero de voz se van sacando de forma alternada por las patillas P2.0 y P2.1.

```
#include <reg51.h>
typedef  unsigned char  U8;
typedef  unsigned int   U16;
#define fXTAL 11059200.0

U8  ACTUAL;               // Byte que se está serializando
U8  RACTUAL;              // Número de repeticiones del carácter ACTUAL
U8  REP;                  // Copia de ACTUAL

U8  bdata BITS=8;         // Contador del número de bits a sacar de un byte de voz
sbit BI=BITS^0;           // El bit BITS.0 indica si se actualiza P1.0 o P1.1
```

```
sbit P20 = P2^0;           // Patilla para la generación del sonido
sbit P21 = P2^1;           // Patilla para la generación del sonido

bit FU0;                   // User flag para indicar dato procesado
bit FU1;                   // User flag para indicar fichero procesado

void iniciaVoz(void)
   {
   TR0 = 0;                // Asegura que el timer 0 esté parado
   TF0 = 0;                // Asegura que el flag de interrupción esté borrado
   TMOD &= 0xf0;           // Sin modificar al T&C1
   TMOD |= 2;              // T&C0 en modo autorrecarga de 8 bits, GATE = 0
   TH0 = -(((fXTAL/12)/22050)+0.5); // Para reproducir sonido a 22050 bits/s
   ET0 = 1;                // Permitir la interrupcion del timer T0
   EA  = 1;                // Permitir las interrupciones
   }                       // Fin de iniciaVoz()

void isrT0(void) interrupt 1
   {
   ACTUAL>>= 1;            // Copia ACTUAL.0 a CY
   if (BI==1)              // Si BI == 1 actualiza P20
      { P20 = CY; }
   else                    // Si BI == 0 actualiza P21
      { P21 = CY; }
   if(--BITS == 0)
      {
      BITS = 8;            // Preparación para el siguiente byte de voz
      if (RACTUAL !=0)     // Si contador de repeticiones distinto de 0
         {
         ACTUAL=REP;       // Repetir el valor anterior y
         RACTUAL--;        // decrementar el contador de repeticiones
         }
      else
         { FU0 = 1; }      // Flag que indica dato (o pareja) procesado
      }
   }                       // Fin de isrT0()

void habla(U8 code *pvoz)
   {
   U8 REP2, CONT_REP=0;
   U16 NBYTES;
   P20 = 1;   P21=0;       // P2.0=1 & P2.1=0, inicialización para silencio (Vcc/2)
   ACTUAL = 0xAA;          // Para reproducir un primer byte de silencio
   FU0 = 0;                // Flag que indica byte de voz no terminado de procesar
   FU1 = 0;                // Flag que indica fichero de voz no terminado de procesar
   TR0 = 1;                // Timer T0 corriendo
   NBYTES= *pvoz<<8;       // Deja en NBYTES el número de datos del fichero de voz
   pvoz++;
   NBYTES |= *pvoz;
   while(FU1==0)           // Mientras no se haya procesado el fichero de voz
      {
      pvoz++;              // Apunta al byte de voz
      REP2    = *pvoz;     // Deja el byte de voz en la variable global REP2
      NBYTES--;            // Un byte menos a procesar
      if((REP2==0xf3)||(REP2==0xf9))
         {                 // Si REP2 == prefijo repetición de 0x55 o de 0xAA
         pvoz++;
         CONT_REP = *pvoz; // Leer nº de repeticiones
         NBYTES--;
         if(CONT_REP != 0) // Si CON_REP == 0 se trata de 0xf3 o 0xf9
            {
            if(REP2 == 0xf3)  { REP2 = 0x55;}
            else              { REP2 = 0xAA;}
            }
         }
      else                 // Si no prefijo de repetición
         { CONT_REP = 0; }
      if(NBYTES==0)
         { FU1=1; }        // Flag que indica fichero procesado
       while(FU0==0)       // Esperar hasta que se haya procesado el dato
         {;}               // o pareja (Prefijo, Repetición) anterior
      REP = ACTUAL = REP2;
      RACTUAL = CONT_REP;
      FU0 = 0;             // Indica que el siguiente dato no se ha procesado
      }                    // Fin de bucle while(FU1==0)
   TR0 = 0;                // Timer T0 parado
   }                       // Fin de habla()

extern U8 code FRASE1[];   // Fichero de voz
```

```
void main(void)
   {
   iniciaVoz();
   habla(FRASE1);          // Reproducir un fichero de voz
   while(1) { ; }          // Bucle sin final
   }                       // Fin de main()
```

El listado anterior contiene cuatro funciones:

1. Función **iniciaVoz(),** se encarga de iniciar un temporizador para que pida 22050 interrupciones al segundo que se utilizarán para actualizar las patillas encargadas de generar la voz
2. La función de atención a la interrupción del temporizador **isrT0()** actualiza las patillas encargadas de generar la voz con la periodicidad requerida. Está muy optimizada y en promedio dura 21 ciclos máquina.
3. La función **habla()** lee de memoria de código el fichero de voz con codificación RC-2, realiza la descompactación RLE de los tramos de silencio y deja preparadas unas variables globales para facilitar el trabajo a la función de interrupción isrT0(). Los dos primeros bytes del fichero de voz informan a la función habla() del número de bytes que tiene que reproducir.
4. La función **main()** llama a las funciones iniciaVoz() y habla() para generar la voz.

13. Otras posibles soluciones a la generación de la voz.

El listado anterior aún siendo totalmente funcional no representa la mejor solución para la generación de voz debido a que somete a la CPU a una elevada carga de trabajo. En efecto, para generar voz, a razón de 22050 muestras por segundo, se generan 22050 interrupciones al segundo lo que deja tan solo 41 ciclos máquina para la rutina de interrupción. Si se deseara generar voz de mayor calidad utilizando 44100 muestras por segundo, la rutina de interrupción dispondría de la mitad del tiempo (20,5 ciclos máquina) y un 8051 trabajando a la frecuencia de 11.0592MHz no serviría. La solución pasa por elegir un microcontrolador más potente, por ejemplo un 8051 más moderno o un AVR, o bien buscar otro tipo de solución que utilice los periféricos que acompañan a muchos microcontroladores para descargar de trabajo a la CPU. Cuando se opta por la última solución, los periféricos susceptibles de ser utilizados son los SPI (Serial Peripheral Interface) o las USART trabajando en modo síncrono. Las típicas UART no sirven debido a que a cada byte transmitido se le añade un bit de Start y un bit de Stop. Cuando se utiliza el SPI o la USART, la CPU escribe un byte en el registro transmisor y el hardware realiza la conversión de paralelo a serie sin añadir carga de trabajo a la CPU; de esta forma se puede generar voz a 44100 muestras por segundo escribiendo en los registros de transmisión unos 5500 bytes por segundo (es decir el cociente de la división de 44100 entre 8). El hardware necesario para decodificar el sonido consistiría en un registro de desplazamiento de dos bits (o un par de flip-flops dispuestos como un registro de desplazamiento de dos bits), dos resistencias y un condensador, véase la figura 12.

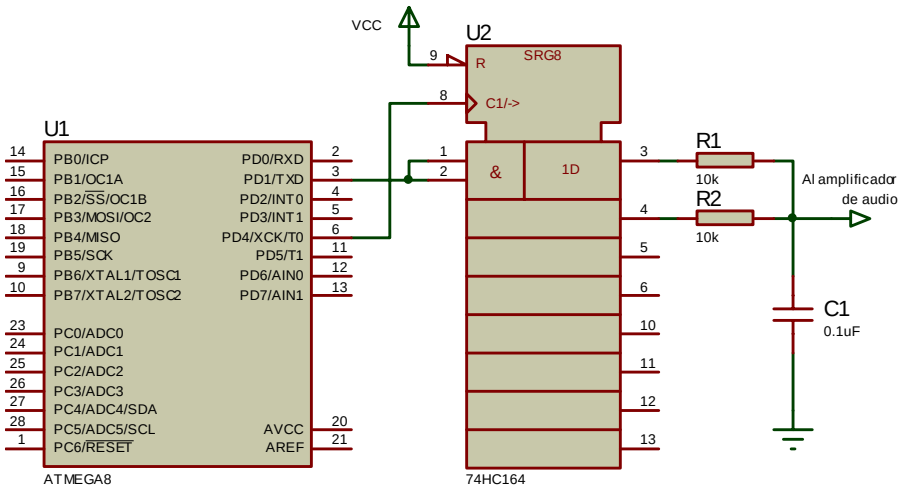

Figura 12. Síntesis de voz con codificación RC-2 usando la USART y un registro de desplazamiento de 2 bits.

La figura 12 muestra un uC ATmega88 que utiliza la USART en modo síncrono. Los datos salen en serie por la patilla 3 (TXD) y el clock sale por la patilla 6 (XCK); del registro de desplazamiento sólo se usan dos bits (patillas de salida 3 y 4). En esta figura se aprecia con claridad que el sonido se genera utilizando muestras de un bit (la información que sale del pin TXD), aunque luego se utiliza el valor de la muestra actual y el de la muestra anterior para reconstruir la señal de voz.

De las dos soluciones propuestas: SPI o USART, la más adecuada es la de la USART debido a que los microcontroladores generalmente disponen de pocas opciones para la frecuencia de clock en el bus SPI y de muchas opciones en el caso de la USART. Por este motivo cuando se usa la USART se puede reproducir el sonido con una frecuencia idéntica a la frecuencia de muestreo, o modificarla ligeramente para reproducir el sonido un poco más rápido o un poco más lento.

14. Obtención de los ficheros de voz con codificación RC-2.

Para que la calculadora pueda reproducir sonidos, éstos deben capturarse previamente, codificarlos en formato RC-2, almacenarlos en una memoria, y guardar las direcciones de comienzo de cada frase. En nuestro ejemplo de calculadora parlante los sonidos a reproducir son 26 (cero, uno, dos, tres, cuatro, cinco, seis, siete, ocho, nueve, mas, menos, signo mas, signo menos, por, dividido, igual a, coma, raíz cuadrada, borrar, con voz, sin voz, más infinito, menos infinito, exponencial, y no número).

La captura de los 26 sonidos se hace con la Grabadora de Sonidos de Windows utilizando el formato: mono, 16-bits, y 22050 muestras al segundo. Si se desea mayor calidad de voz puede aumentarse la frecuencia de muestreo a 44100 muestras/s. No es necesario realizar 26 grabaciones (una por sonido), se pueden realizar tres o cuatro grabaciones de grupos de sonidos para que resulten más uniformes; posteriormente se deberá cortar cada sonido, codificarlo de forma individual con el software "RC Sound Encoder" [8] y guardar el sonido codificado en lenguaje C, lenguaje ASM o formato BIN. Si el usuario dispone de un software de edición de audio tal como ACOUSTICA [9] podrá: grabar voz, ecualizar de sonidos, seleccionar regiones de audio, cortar y salvar secciones, eliminar ruidos, etc., con facilidad y en general obtendrá mejores resultados que trabajando exclusivamente con la Grabadora de Sonidos de Windows. Para la calculadora se ha salvado los ficheros de sonido en lenguaje ASM; en este formato, el sonido "uno" ofrece el siguiente aspecto:

```
UNO:   DB     3,  77, 243, 255, 243, 230, 149, 249,   2,  74, 181, 170,  42,  85, 181, 249
       DB     7,  74, 243,  46, 213, 170,  82, 243,   1,  42, 243,   2, 249,   4,  85, 173
       . . . . .
       DB   149, 249,   7,  84,  85, 165, 249,  14,  42, 243,  84, 165, 249, 192,  84
```

Los dos primeros bytes, 3 y 77, indican que el sonido está codificado en 845 bytes (3·256+77), esta información es utilizada por la función habla(), descrita en el apartado 12, para reproducir el sonido. Todo lo que queda consiste en agrupar los 26 ficheros ASM en un solo fichero ASM cuyas primeras líneas contengan las direcciones de comienzo de cada sonido en la forma siguiente:

```
       DW   CERO, UNO, DOS, TRES, CUATRO, CINCO, SEIS, SIETE, OCHO, NUEVE
       DW   PUNTO, EXPO, MAS, MENOS, POR, DIVI, BORRAR, IGUAL, RAIZ, MASINF
       DW   MENOSINF, NAN, SIGNOMAS, SIGNOMENOS, CONVOZ, SINVOZ
```

El contenido del fichero ASM se puede ensamblar y almacenar en una memoria serie externa al microcontrolador; de esta forma se puede adaptar la calculadora a un idioma diferente con solo cambiar la memoria serie. El resto del programa se ha escrito en lenguaje C. Para enlazar los ficheros C con el fichero ASM se debe incluir en el fichero C la línea:

```
enum frases {  // Aqui reside la union entre el programa en C y las frases en la EEPROM
     CERO, UNO, DOS, TRES, CUATRO, CINCO, SEIS, SIETE, OCHO, NUEVE,  // NO MODIFICAR !!!!!
     PUNTO, EXPO, MAS, MENOS, POR, DIVI, BORRAR, IGUAL, RAIZ, MASINF,
     MENOSINF, NONUM, SIGNOMAS, SIGNOMENOS, CONVOZ, SINVOZ};
```

Puede apreciarse que las constantes enumeradas siguen el mismo orden en el fichero ASM que en el código escrito en lenguaje C. Las constantes enumeradas sirven para poder generar sonidos mediante la función habla(); ésta función recibe como parámetro de entrada una de estas constantes; así para hacer

que la calculadora diga “mas infinito” se incluirá la línea “habla(MASINF);”. La función habla() utilizará la constante de entrada para conseguir la dirección de comienzo de la frase “mas infinito”, luego se posicionará a comienzo de la misma para obtener los dos primeros bytes que indican el número de bytes que debe reproducir, y seguidamente iniciará la interrupción de un temporizador, tal como aparece en el listado del apartado 12. El fichero en lenguaje ensamblador utiliza un tamaño de 37.551 bytes, para almacenar los 26 sonidos codificados y las direcciones de comienzo de los mismos.

15. Conclusiones

Este trabajo presenta el codificador de voz RC-2 cuya salida digital consiste en un bit por muestra y que genera voz de buena calidad con un reducido bit-rate. El codificador se comporta bien con señales del tipo no-voz y puede ser usado por cualquier persona independientemente del idioma en el que se exprese. La codificación exige poca complejidad de cálculo e introduce una demora insignificante. La decodificación del sonido es aún más sencilla, sólo requiere dos resistencias y un condensador. Se puede afirmar que el comportamiento del codificador RC-2 es superior al del conocido codificador CVSD utilizado a lo largo de más de 30 años y que recientemente ha sido seleccionado, junto al PCM A-law y al PCM μ-law, como uno de los posibles esquemas de codificación utilizados por la tecnología Bluetooth [6] debido a su comportamiento robusto frente a errores de transmisión. Además, el artículo presenta un software de PC que permite comprimir archivos WAV en formato RC-2 y se muestra con detalle una metodología que permite añadir voz a diseños basados en microcontrolador sin utilizar chips especializados. El software de libre disposición “RC Sound Encoder”, desarrollado por el autor, codifica los ficheros de sonido de tal forma que pueden reproducirse utilizando dos patillas de salida digitales, dos resistencias y un condensador. Como ejemplo para la realización de nuevos diseños se muestra el hardware y software de una Calculadora Parlante que opera con números reales y que utiliza la voz como elemento de salida hacia las personas. En [5] y [8] puede conseguirse el código fuente en lenguaje C para reproducir sonido codificado (con y sin compactación RLE), usando un μC de la familia 8051. El software “RC Sound Encoder”, descrito en el apartado 7, puede descargarse de [5].

Referencias

[1] D. Salomon, *Data Compression, the Complete Reference*, 3rd ed., Springer, 2004.
[2] R.Goldber and L. Riek, *A Practical Handbook of Speech Coders*, CRC Press, 2000.
[3] W.Chu, *Speech Coding Algorithms*, Wiley-Interscience, 2003.
[4] National Instruments, www.ni.com
[5] ftp.circuitcellar.com/pub/Circuit_Cellar/2005/180.
[6] D. Kammer et al., *Bluetooth Application Developer’s Guide*, Syngress Publishing, Inc., 2002.
[7] **AN1544** *Design of Continuously Variable Slope Delta Modulation Communications Systems*. Motorola Analog IC Device Data, 1996.
[8] M. Barrón, *Speech Waveform Encoder*, Circuit Cellar #180, Vernon CT, págs. 16-26, July 2005.
[9] http://www.acondigital.com