Universidad Nacional Autónoma de México

Facultad de Ingeniería

Síntesis de Voz en Tiempo Real empleando una arquitectura DSP

Gerardo Barajas Puente
Miguel Angel Molero Armenta

Director de Tesis:
M.I. Larry Hipólito Escobar Salguero

Abril 2004

Índice general

Índice de figuras

Índice de cuadros

Glosario y Abreviaturas

A/D: Analógico - Digital.

ADC: Convertidor Analógico - Digital.

CCS: Code Composer Studio. Ambiente integrado de desarrollo para la tarjeta DSK.

CODEC: Convertidor A/D y D/A (TLC320AD50C).

D/A: Digital - Analógico.

DAA: Interface de puerto telefónico.

DAC: Convertidor Digital - Analógico.

DARAM: Memoria RAM de acceso dual.

DFT: Transformada Discreta de Fourier.

DRR: Registro de recepción del puerto serie multicanal bufereado integrado en el DSP.

DSK: DSP Starter Kit. Tarjeta de desarrollo del DSP TMS320C5402.

DSP: Procesador de Señales Digitales.

DXR: Registro de transmisión del puerto serie multicanal bufereado integrado en el DSP.

FIR: Respuesta Finita al Impulso.

FFT: Transformada Rápida de Fourier.

FORMANTES: Frecuencia Formante o frecuencia resonante de la cavidad del tracto Vocal.

IDFT: Transformada Discreta de Fourier Inversa.

IEEE: Instituto de Ingenieros Eléctricos - Electrónicos.

IIR: Respuesta Infinita al Impulso.

JTAG: Controlador de prueba de bus para la emulación e interface con el puerto huésped conectado a la PC por medio del puerto paralelo.

LPC: Codificación de predicción Lineal.

MIPS: Millones de Instrucciones por segundo.

MODEM: Interfaz analógica - digital que se encarga de establecer los protocolos necesarios para modular las señales a transmitir de manera digital, así como la demodulación de los datos en la recepción a través de la línea telefónica.

PC: Computadora Personal.

PDS: Procesamiento Digital de Señales.

PITCH: Frecuencia fundamental o tono.

RAM: Memoria de Acceso aleatorio.

RTDX: Intercambio de datos en tiempo real. Aplicación integrada en el ambiente de desarrollo CCS.

VOCODER: Codificador de Voz.

WORDS: Palabra digital de 16 bits.

WSS: Proceso estacionario en el sentido amplio.

Resumen

El presente trabajo de tesis tiene como objetivo *implantar un sistema de síntesis de voz en tiempo real empleando un procesador de señales digitales (DSP); en particular, el TMS320C5402 de la familia de DSP's C5000 de punto fijo, de la marca Texas Instruments (TI)*. Los objetivos particulares engloban la validación del desempeño de una arquitectura DSP para un sistema de síntesis de voz en tiempo real, así como un análisis comparativo del sistema de síntesis de voz entre aritmética de punto fijo y flotante. Con base a la técnica de síntesis de voz se implanta un sistema para la caracterización de la señal de voz en función de un conjunto de parámetros que emulan el modelo del tracto vocal esperando una tasa baja de bits para su posible codificación, transmisión y almacenamiento. Por tanto, el diseño del sistema de síntesis de voz tendrá que cumplir con los requerimientos necesarios para su operación en *tiempo real*, además de contemplar la opción de ser un sistema portable y/o configurable en un sistema multitareas. Como aplicación de interés se presenta una evaluación de un sistema de comunicaciones para seguridad telefónica empleando un sistema de síntesis de voz.

Capítulo 1

Introducción

En la actualidad los sistemas de comunicaciones han sido y son parte crucial del desarrollo actual de la humanidad, ya que este tipo de procesos son vitales para cualquier actividad humana. Con este desarrollo tecnológico e informático que se ha venido dando en los tiempos recientes, cualquier persona puede tener acceso a estos recursos, tener contacto con los hechos del mundo, los sistemas financieros, las investigaciones de vanguardia, los eventos de la semana y mantener el contacto con sus seres queridos de manera rápida, eficiente y privada [1].

Las tendencias tecnológicas nos señalan que los teléfonos celulares, las interfaces conversacionales, Internet, entre otras tantas tecnologías presentan soluciones a las necesidades de comunicación actuales. En la mayoría de estas iniciativas de vanguardia existe un común denominador, que es la interfaz de comunicación más natural que existe: *la voz*.
Esto permite que el usuario pueda disponer de la movilidad suficiente para ejecutar otras tareas al mismo tiempo [2]. Debido a lo anterior, se puede observar que los procesos que consideran a la señal de voz son la parte medular de los sistemas de comunicación. Por lo que es necesario establecer aquellos procesos de voz que mejoren las tareas del almacenamiento, la velocidad de transmisión - recepción, la inteligibilidad (comprensión de la información), privacidad y la facilidad de uso con la calidad requerida en las aplicaciones en que tomen parte.

La reducción de la cantidad de datos que representan la información, concretamente en una señal de voz, es un proceso indispensable para la manipulación de ésta en cualquier sistema que se desee implantar en tiempo real. Tal reducción justifica el empleo de estrategias para eliminar la redundancia que presentan las señales de voz; siendo las técnicas de compresión y síntesis dos opciones eficientes para cumplir tal propósito. La elección de las técnicas a emplear dependerá de la aplicación a implantar, de las calidades perceptuales y la tasa de bits requerida para su transmisión y/o almacenamiento.
Por tanto, la técnica de sintesis de voz se eligió por que considera las propiedades físicas del sistema de producción de la señal de voz, caracterizándola con una calidad aceptable, y minimizando la tasa de bits a emplear sin perder la calidad de la voz al reconstruirla o descomprimirla. Por ejemplo, en lugar de transmitir 8000 muestras/s se podría reducir las muestras a transmitir hasta en un 10 % o menos [3].

La finalidad de implantar un sistema de síntesis de voz en tiempo real es la de concebir un bloque modular que pueda ser partícipe en un sistema más complejo como por ejemplo, un sistema de reconocimiento, transmisión, encriptamiento y almacenamiento de voz.

Por lo que el objetivo principal del presente trabajo es:

Implantar un sistema de síntesis de voz en tiempo real empleando un procesador de señales digitales (DSP) de aritmética de punto fijo; en particular, el TMS320C5402 de la familia de DSP's C5000 de la marca Texas Instruments (TI).

Los objetivos particulares que se fijaron son:

- Desarrollar un sistema de Síntesis en Tiempo Real.
- Validar el desempeño de una arquitectura DSP para un sistema de síntesis de voz en tiempo real.
- Análisis Comparativo del sistema de síntesis de voz entre aritmética de punto fijo y flotante.
- Evaluación de un sistema de comunicaciones para seguridad telefónica empleando un sistema de síntesis de voz.

En síntesis de voz, la señal es caracterizada en términos de un conjunto de parámetros de un modelo, los cuales pueden ser codificados para su almacenamiento y/o transmisión. Para nuestro sistema de síntesis empleamos el método de predicción lineal, siendo su solución un conjunto de coeficientes $\{a_i\}$ con los cuales definimos un filtro Todo Polo que emula las estructuras o cavidades resonantes del aparato de producción humano de la voz, principalmente el tracto vocal, y como excitación del filtro dos tipos de señales de entrada (tren de impulsos y ruido blanco) según la naturaleza del sonido (voceado o no voceado), que fungen como la emulación de las ondas de presión que emanan de las cuerdas vocales. Sin embargo, en muchos sistemas de síntesis de voz prácticos, la equivalencia entre la fisiología y el modelo es a menudo menos importante que el esfuerzo computacional y la calidad de la voz necesaria para la implantación [4]. Generalmente se analiza la señal de voz en segmentos de 160-256 muestras, de tal manera se justifica que este segmento cumple con las propiedades de estacionaridad[1] y ergódicidad[2] [4].

Para cada segmento de voz se pueden almacenar o transmitir los siguientes parámetros;

- Coeficientes del filtro Todo Polo $\{a_i\}$.
- La ganancia del segmento de voz G.
- La naturaleza del segmento de voz (voceado o no).
- La frecuencia fundamental (pitch).

Aunque un sistema de síntesis es integrado por algoritmos básicos del procesamiento digital de señales, se tienen que considerar los desempeños del sistema como son: la precisión numérica, la carga computacional y los tiempos de cálculo necesarios para su implantación en tiempo real, lo cual es uno de los objetivos primordiales del presente trabajo. Además, al implantar el sistema de síntesis de voz en tiempo real, es necesario establecer una organización y sincronía entre los

[1] Se dice que una señal es estacionaria si sus estimaciones estadísticas no varían en el tiempo [5].
[2] Una señal es ergódica si sus estimaciones estadísticas se aproximan a sus estimaciones temporales [5].

procesos de adquisición de la señal, la estimación de los parámetros y la entrega de la señal reconstruida, observando que es fundamental el uso de un DSP, dado que es una arquitectura acorde al diseño de sistemas incrustados y multitareas. Cabe destacar que se diseña el sistema de síntesis como un bloque modular que podrá ser integrado en sistemas de voz más complejos.

El sistema de síntesis de voz propuesto se implanta en una tarjeta de evaluación y desarrollo (DSK) que integra un DSP TMS320C5402 que hará la adquisición de la señal de voz real, el proceso de análisis - síntesis y la entrega de voz sintética en tiempo real. Por lo tanto, el diseño del sistema de síntesis de voz tendrá que cumplir con los requerimientos necesarios para su operación en *tiempo real* así como tener la opción de ser portable y/o configurable en un sistema multitareas.

A continuación se enumeran una serie de aplicaciones donde se pueden aplicar los resultados del presente trabajo:

- Reconocimiento de voz en tiempo real.
- Sistemas de almacenamiento masivo de voz.
- Sistemas de comunicaciones.
- Sistemas de seguridad para comunicaciones.

Como aplicación de interés se propone la implantación de un sistema de comunicaciones para seguridad telefónica usando el sistema de síntesis de voz diseñado. En esta propuesta se describe los módulos necesarios y los requerimientos para su futura implantación.

El presente trabajo está organizado de la siguiente manera: en el *capitulo dos*, se hace una breve introducción a la señal de voz, su sistema de producción y el modelo digital de representación para ella. En el *capítulo tres*, se muestran las técnicas clásicas de codificación de voz y se introduce la teoría del método de predicción lineal. En los *capítulos cuatro y cinco*, se desarrolla el diseño de síntesis de voz e implantación en el DSP de punto fijo (TMS320C5402), respectivamente. Los resultados obtenidos se presentan en el *capitulo seis*, así como una propuesta de aplicación práctica del diseño del sistema de síntesis (sistema de comunicaciones para seguridad telefónica). Finalmente, en el *capítulo siete* se integran las conclusiones del trabajo. Además se incluyen apéndices referentes a las principales características de la arquitectura DSP empleada y una introducción al ambiente de desarrollo Code Composer Studio (CCS).

Capítulo 2

La Señal de Voz

Al implantar cualquier sistema de procesamiento digital para las señales de voz es necesario estudiar y establecer las principales características y parámetros que determinan a ésta, por lo que es indispensable dedicar este capítulo para el estudio de sus características, naturaleza y los modelos de producción de voz. Para ello, consideramos a la voz como una onda de presión acústica originada a partir de movimientos voluntarios de estructuras anatómicas, la cual es producida por el sistema de producción de voz [5].

2.1. Producción de la Voz

El sistema de producción de la voz está constituido por los pulmones, la tráquea, la laringe y la cavidad faringea, bucal y nasal. Los pulmones y la tráquea forman el *tracto pulmonar*, la cavidad bucal y faringea se agrupan en un subsistema llamado *tracto vocal*, mientras que la cavidad nasal se denomina *tracto nasal*. Para que un sistema de producción de voz sea preciso se deberá considerar a las cuerdas vocales, el velo, la lengua, los dientes y los labios. Dichos elementos se mueven en diferentes posiciones para la producción de distintos sonidos, por tanto, se denominan *elementos articuladores* o simplemente *articuladores*. La mandíbula o maxila también se considera como un elemento articulador, siendo responsable de afectar al tamaño y la forma del tracto vocal, como también las posibles posiciones de los otros articuladores. En la figura (2.1) se muestra la sección sagital del sistema de producción de voz.

2.1.1. Generación de la Voz

En el tracto pulmonar, los pulmones generan aire comprimido que fluye a través de la tráquea, de tal manera que en este subsistema se define la intensidad de los sonidos a producir y rara vez hace una notoria contribución auditiva, excepto en los posibles sonidos pausados [6]. La laringe está constituida por un conjunto de cartílagos (cricoideo, tiroideo) y músculos, además de contener a las cuerdas vocales. Su interior recibe el nombre de epiglotis y presenta dos pares de pliegues, el par superior recibe el nombre de pliegues vestibulares (cuerdas vocales falsas) y el inferior, de pliegues vocales (cuerdas vocales verdaderas que vibran al paso del aire produciendo la voz). La función de la laringe en la producción de la voz consiste en proveer una excitación que genera sonidos con cierta periodicidad, mediante una vibración periódica de las cuerdas vocales. La función del tracto vocal es la articulación y modulación de la voz mediante la lengua, los labios, los dientes y la maxila. La forma de onda del sonido producido en la laringe es rica en armónicas que son

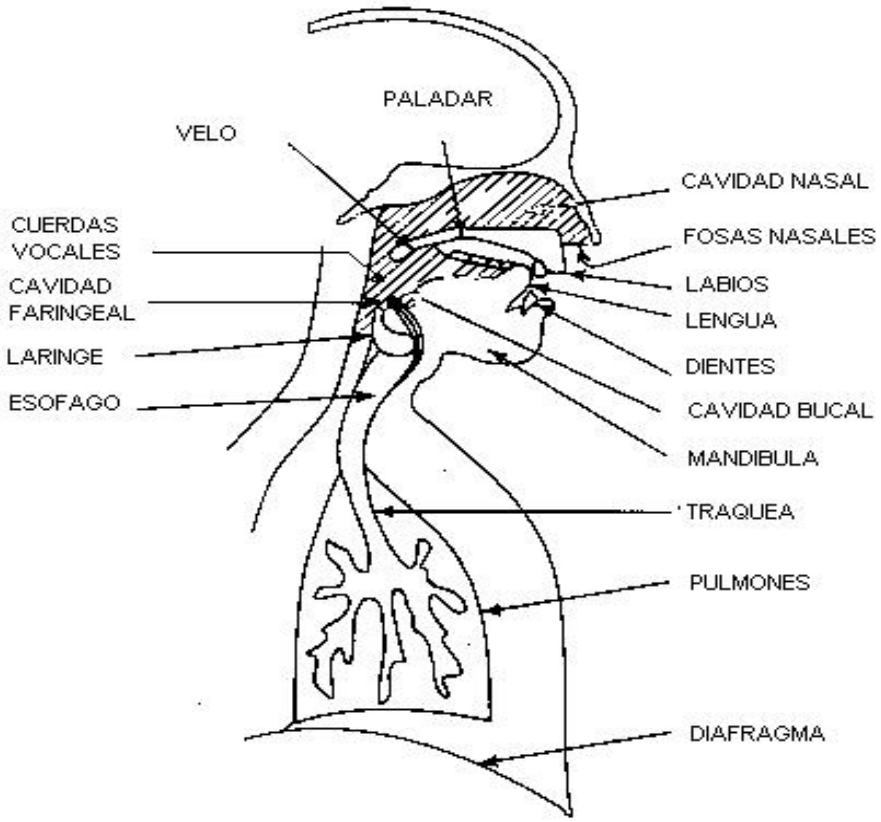

Figura 2.1: Diagrama del Sistema de Producción de la Voz (Deller 1993).

modificadas por el tracto vocal dependiendo de la forma de éste, de tal manera que el sonido está caracterizado por una serie de frecuencias resonantes llamadas *formantes*. El tracto nasal constituye un camino auxiliar para la transmisión del sonido produciendo de está manera sonidos nasales. Para su estudio es útil asociar al sistema de producción de la voz en términos de un filtrado acústico. Tal modelo deberá tomar en cuenta las contribuciones de las estructuras resonantes de las tres cavidades -anteriormente presentadas- como un filtro acústico. Un diagrama de bloques del modelo acústico simplificado se ilustra en la figura (2.2).

2.1.2. Clasificación de los Sonidos de la Voz

Los sonidos de la señal de voz pueden identificarse en dos clases según su modo de excitación: sonidos voceados o no voceados. Sin embargo, al combinar sonidos voceados, no voceados y tramos de silencio se generan otros tipos de sonidos, tales como; sonidos mezclados, sonidos plosivos, susurros, o bien, sólo tramos de silencio [5].

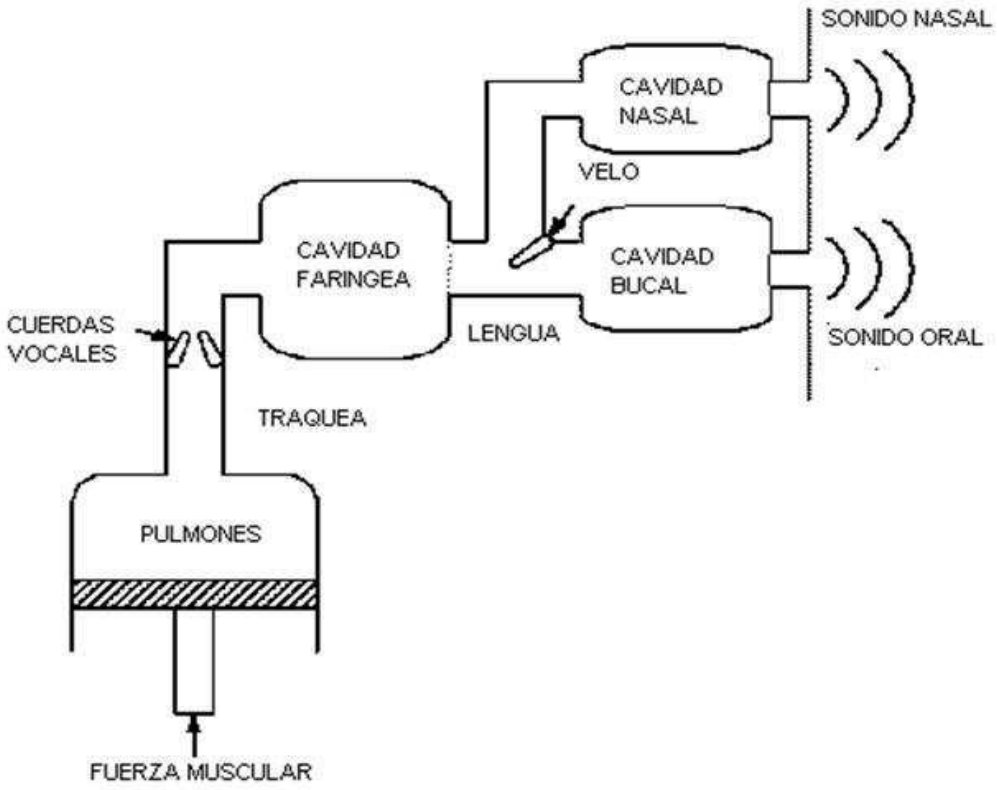

Figura 2.2: Diagrama de Bloques del Modelo Acústico simplificado (Deller 1993).

Los *Sonidos Voceados* son producidos como consecuencia de la oscilación de las cuerdas vocales forzando la apertura de la epiglotis. La apertura y cierre de las cuerdas vocales secciona el pulso de aire en pulsos cuasi - periódicos llamados pulsos glotales, aproximadamente con forma de onda triangular, con una frecuencia fundamental llamada *frecuencia de pitch* y frecuencias armónicas que disminuyen su amplitud a razón de 40 dB/decada [7].
El tracto vocal actúa como una cavidad resonante con una naturaleza de un filtro paso - bajas, proporcionando un espectro con una fuerte frecuencia fundamental y atenuando las frecuencias armónicas.
El intervalo en el que vibran las cuerdas vocales depende de la presión del aire comprimido proveniente de los pulmones y la tensión de éstas, obteniendo así, la variación de la frecuencia fundamental (pitch) y produciendo distintos sonidos voceados, donde el intervalo de pitch para hombres adultos es de 50 Hz hasta 250 Hz, y para una mujer adulta de 120 Hz hasta 500 Hz [5].
En la figura (2.3) se muestra un segmento de sonido voceado de voz real masculina (256 muestras del fonema /e/ a una frecuencia de muestreo de $f_s = 8$ kHz).

En los *sonidos no voceados* las cuerdas vocales no vibran y son generados mediante la contracción de algunos puntos a lo largo del tracto vocal, forzando al aire a atravesar dicha contracción para producir cierta turbulencia, por tanto, la excitación de estos sonidos es de naturaleza ruidosa.
En la figura (2.4) se muestra un segmento de sonido no voceado de voz real masculina (256 muestras del fonema /s/ a $f_s = 8$ kHz).

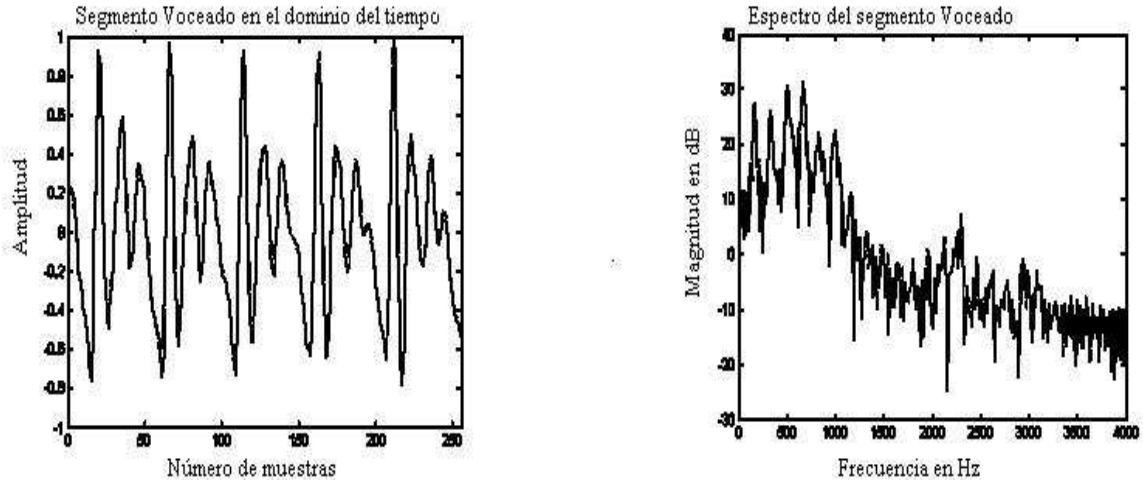

Figura 2.3: Representación de un segmento de Sonido Voceado en los dominios del tiempo y la frecuencia.

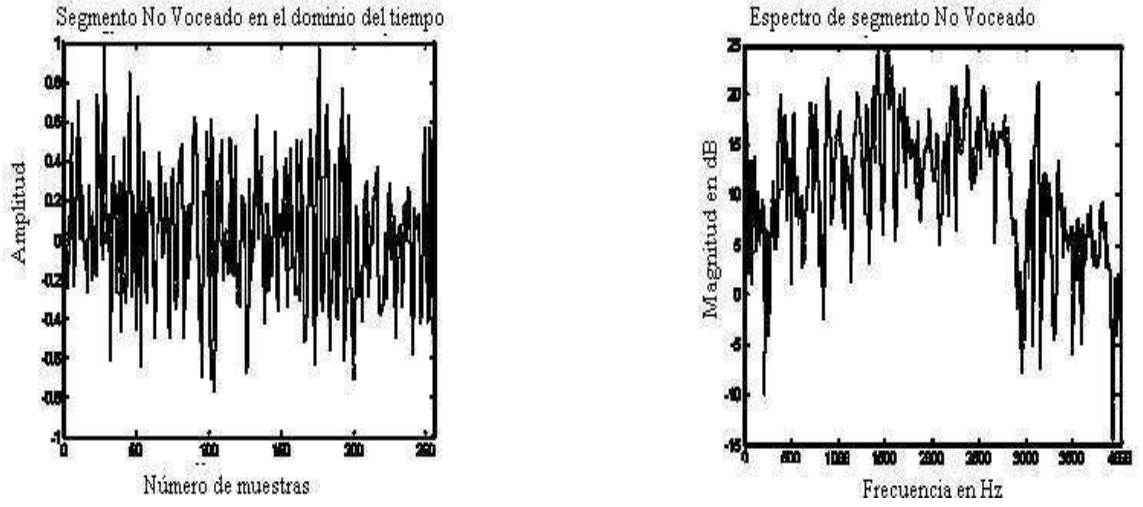

Figura 2.4: Representación de un segmento de Sonido No Voceado en los dominios del tiempo y la frecuencia.

2.2. Teoría Acústica de la Producción de la Voz

Anteriormente, hemos explicado el mecanismo de producción de la voz, su generación y sus posibles tipos de excitaciones desde un punto de vista fisiológico, por tanto, es necesario obtener las representaciones matemáticas del proceso de producción de voz y de esta manera llegar a un modelo acústico de la producción de voz.

2.2.1. Modelo Acústico de la Producción de la Voz

Las ondas de sonido son generadas a partir de vibraciones que se propagan en el aire u otro medio por vibraciones de las partículas de éste. De tal manera que la generación y propagación de los sonidos del sistema de producción de la voz obedecen a las leyes fundamentales de la física, es decir, por medio de los principios de conservación de la masa y energía, las leyes de la termodinámica y de la mecánica de fluidos podemos obtener las ecuaciones diferenciales parciales que lo describen [8]. Para formular detalladamente el sistema de producción de la voz debemos considerar [5], [8]:

- La variación temporal de la forma del tracto vocal.
- Las pérdidas a causa de los efectos térmicos y la fricción viscosa a lo largo de las paredes del tracto vocal.
- Los efectos del tejido epitelial a lo largo de las paredes del tracto vocal.
- La radiación del sonido en los labios.
- El acoplamiento de la cavidad nasal.
- Los efectos del tracto pulmonar acoplados con la estructura resonante del tracto vocal.
- La posible excitación del sonido en el tracto vocal.

2.2.2. Modelado del Tracto Vocal

Para obtener un modelo acústico de la producción de la voz (simplificado) consideramos al tracto vocal como un tubo de sección transversal no uniforme y variante en el tiempo. La figura (2.5) muestra un esquema del modelo del tracto vocal.

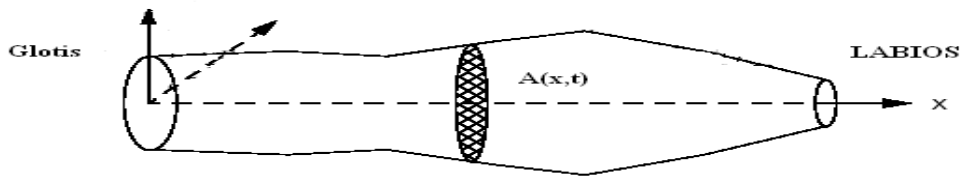

Figura 2.5: Esquema del modelo del Tracto Vocal.

Por cuestiones de simplificación consideramos que no existen pérdidas a causa de efectos térmicos y fricción viscosa a lo largo del tubo. Las ecuaciones en derivadas parciales (2.1) y (2.2) describen el comportamiento del tracto vocal [8].

$$-\frac{\partial p}{\partial x} = \rho \frac{\partial (u/A)}{\partial t} \tag{2.1}$$

$$-\frac{\partial u}{\partial x} = \frac{1}{\rho c^2}\frac{\partial (pA)}{\partial t} + \frac{\partial A}{\partial t} \tag{2.2}$$

donde:

$p = p(u,t)$:	Variación de la presión del sonido.
$u = u(x,t)$:	Variación del flujo volumétrico de aire.
ρ :	Densidad del aire en el tubo.
c :	Velocidad del sonido en el tubo.
$A = A(x,t)$:	Función Area del tubo.

Aunque las ecuaciones (2.1) y (2.2) representan un modelo simplificado del tracto vocal no es posible obtener su solución cerrada excepto con configuraciones más sencillas, o bien, obtener sus soluciones por métodos numéricos [8]. Una solución completa de las ecuaciones diferenciales parciales que modelan al tracto vocal requieren determinar los valores de la presión y el flujo volumétrico en las posiciones x e instantes t que definen las regiones de frontera de la glotis y los labios. Las condiciones de frontera de la glotis dependen del tipo de excitación, mientras que en los labios, los efectos de radiación definen sus condiciones de frontera [8].

2.3. Modelo Digital para la Producción de la Voz

En la sección anterior describimos las representaciones matemáticas del sistema de producción de la voz, concentrándonos en el modelo acústico del tracto vocal y describiendo las simplificaciones para la obtención de su solución. Sin embargo, debemos encontrar un modelo que nos garantice y simplifique el comportamiento del sistema de producción de la voz evitándonos ecuaciones o sistemas complejos. Mediante un modelo llamado modelo terminal - análogo (figura 2.6), podemos obtener una representación equivalente de las terminales de salida del modelo físico sin considerar la estructura interna del sistema de producción de voz [8]. Dicho modelo pretende representar un proceso de producción de voz basado en las características de la señal de salida produciendo una calidad razonable para propósitos de aplicaciones que contemplen la codificación, transmisión y almacenamiento.

En términos generales del modelo terminal - análogo, un modelo del tracto vocal $H(z)$ y un modelo de la radiación de los labios $R(z)$ son excitados por una señal de entrada que modela los efectos de la glotis en tiempo discreto $u_{glotis}(n)$. Para sonidos no voceados, la fuente de excitación es una fuente de ruido con espectro plano modelada por un generador de ruido blanco. Para sonidos voceados, su respectiva excitación necesita una estimación del período de pitch para determinar un generador de tren de impulsos que controla un filtro de forma de pulsos glotales $G(z)$. Esta

excitación produce un pulso glotal de forma de onda similar a la envolvente de los sonidos voceados en cuestión [5].

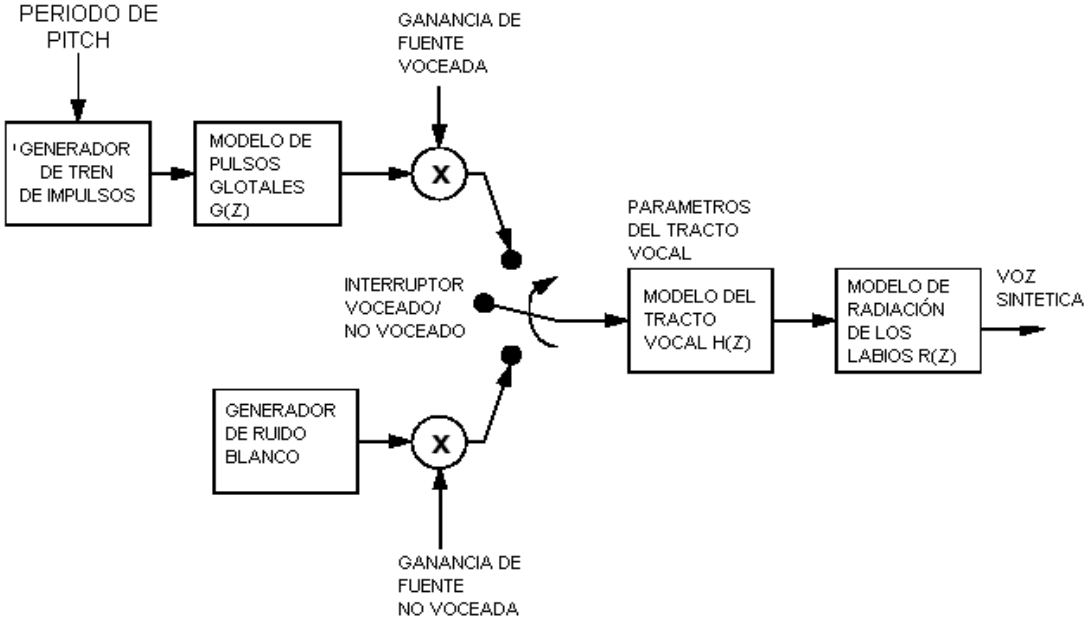

Figura 2.6: Modelo General de Producción de Voz en Tiempo Discreto.

La función de transferencia $H(z)$ del tracto vocal se muestra en la ecuación (2.3):

$$H(z) = \frac{H_0}{1 + \sum_{k=1}^{N} a_k z^{-k}} = \frac{H_0}{\prod_{k=1}^{N} (1 - p_k z^{-1})} \tag{2.3}$$

Donde H_o representa una ganancia global, p_k la ubicación de los polos complejos para el modelo de N-tubos (modelo acústico de producción de voz) y $\{a_k\}$ el conjunto de coeficientes del filtro [5].

Cada ubicación de los pares de polos conjugados en el plano $\mathcal{Z}$, de manera aproximada, *corresponde a un formante* en el espectro de $H(z)$. Para que $H(z)$ sea estable, entonces todos los polos deberán estar ubicados dentro de la circunferencia unitaria en el dominio $\mathcal{Z}$. Si los polos, en el plano $\mathcal{Z}$, están separados, una buena estimación para la obtención de las frecuencias formantes $\widehat{F}_i$ con sus respectivos anchos de banda $\widehat{B}_i$ están dadas por las ecuaciones (2.4) y (2.5) [5]:

$$\hat{F}_i = \left(\frac{f_s}{2\pi}\right) \arctan\left[\frac{\mathrm{Im}\,(p_i)}{\mathrm{Re}\,(p_i)}\right] \tag{2.4}$$

$$\hat{B}_i = -\left(\frac{f_s}{2\pi}\right) In\,|p_i| \tag{2.5}$$

Donde F_i representa el i-ésimo formante y p_i su correspondiente polo en la mitad superior del plano $\mathcal{Z}$. La frecuencia de muestreo es denotada por f_s en Hz.

Para sonidos voceados, el modelo digital completo incluye un subsistema $G(z)$ que modela a la laringe (filtro de forma de pulsos glotales). Dependiendo del propósito del modelo, el filtro glotal puede ser representado por una función de transferencia Todo Polo como la ecuación (2.3), o bien, se sugiere usar una función con dos polos que es apropiada para la respuesta de dicho filtro (ecuación 2.6) [5].

$$u_{glotis}(n) = [\alpha^n - \beta^n]\, U(n),\ \beta < \alpha < 1,\ \alpha \approx 1 \tag{2.6}$$

Donde $U(n)$ es la función escalón. De resultados empíricos, la ecuación (2.6) es una buena elección, observando que no se requiere de ceros para modelar $G(z)$ [5]. Los parámetros α y β son los polos que modelan los efectos en frecuencia de la glotis, además de contribuir en magnitud al espectro del sonido Voceado.

Los efectos de radiación de los labios $R(z)$ (o en términos de una impedancia y frecuencia analógica $R(\Omega) = Z_{labios}(\Omega)$) puede pensarse como una carga de baja impedancia en las terminales del tracto vocal para convertir el flujo volumétrico en los labios a una onda de presión a radiar. Tal impedancia está definida por las ecuaciones (2.7) y (2.8) [5],[9]:

$$|Z_{labios}\,(\Omega)| = \frac{\Omega K_1 K_2}{\sqrt{K_1^2 + \Omega^2 K_2^2}} \tag{2.7}$$

$$arg\,\{Z_{labios}\,(\Omega)\} = \frac{\pi}{2} - arctan\left\{\frac{\Omega K_2}{K_1}\right\} \tag{2.8}$$

Donde $K_1 = 128/9\pi^2$ y $K_2 = 8r/3\pi c$, con r como el radio de la abertura en los labios que se asume circular y c la velocidad del sonido. De cualquier manera, para el modelo digital completo sólo consideramos los efectos de las magnitudes espectrales, por tanto, se observa que Z_{labios} tiene un comportamiento de un filtro paso - altas: $|Z_{labios}(0)| = 0$ y $dZ_{labios}/d\Omega$, que tiene valores positivos en los intervalos de frecuencia prácticos de Nyquist (0 - 4000 Hz) [5]. Un filtro digital que consta de las anteriores propiedades es un diferenciador como se muestra en la ecuación (2.9):

$$R\,(z) = Z_{labios}(z) = 1 - z^{-1} \tag{2.9}$$

El filtro de la ecuación (2.9) tiene un cero en $z_o = 1$ en el plano $\mathcal{Z}$. Existen situaciones en la cuales se presenta el inverso de este filtro ($R^{-1}(z)$) en el modelado de la voz, por tanto, se acostumbra disminuir ligeramente el radio de z_o de tal manera que el filtro inverso sea estable [5]. En este caso, el modelo es representado con la ecuación (2.10):

$$R(z) = 1 - z_o z^{-1}, \quad z_o < 1 \tag{2.10}$$

Conjuntando todos los subsistemas, asumimos que la salida (onda de presión) del sistema de producción de voz es el resultado de filtrar las excitaciones posibles (fuente voceada o no voceada) en filtros lineales por separado.

Entonces podemos definir a la señal de salida $S(z)$ por la ecuación (2.11):

$$S(z) = \begin{cases} E_{nv}(z)H(z)R(z) & \text{para sonidos no voceados} \\ E_v(z)G(z)H(z)R(z) & \text{para sonidos voceados} \end{cases} \tag{2.11}$$

Donde $E_{nv}(z)$ representa la fuente de excitación no voceada (generador de ruido blanco) y $E_v(z)$ la fuente de excitación voceada (generador de tren de impulsos).

2.4. Resumen

En este capítulo se cubrieron los tópicos relacionados con el sistema de producción de voz, describiendo los componentes que integran a éste, explicando sus procesos desde un punto de vista fisiológico - anatómico y acústico, ayudándonos mediante la teoría acústica de la producción de la voz. Antes de establecer un modelo digital de producción de voz, discutimos sus posibles representaciones matemáticas de dicho sistema sin profundizar en el tema, dado que un análisis más complejo está afuera del alcance del presente trabajo de tesis, por lo que sólo se concluyó en un modelo digital completo de la producción de voz en tiempo discreto llamado modelo terminal - análogo explicando sólo los efectos de manera cualitativa en cada subsistema que lo integran. Cabe mencionar que el modelo anteriormente presentado, se describe en términos generales, dado que en el capítulo siguiente se presentará un algoritmo rápido para la obtención de los parámetros que modelan al tracto vocal segmentando a la señal de voz en tiempos cortos. En la extracción de los parámetros del sistema de producción de voz, éstos son conocidos como parámetros de predicción lineal, por lo que el modelo terminal - análogo se simplificará asumiendo un modelo de filtro - fuente, desacoplando los efectos de la glotis al tracto vocal y así obteniendo los parámetros a través de una algorítmia de predicción lineal.

Capítulo 3

Codificadores de Voz

Las técnicas de codificación de la señal de voz pretenden reducir el volumen de información necesario para almacenar o transmitir una señal de voz, de tal forma que la pérdida de calidad en la señal reconstruida con respecto a la señal sin comprimir sea mínima; además de mantener la inteligibilidad del mensaje y existir un compromiso de calidad contra la tasa de compresión, complejidad computacional, etc. Los métodos para la codificación de la señal de voz se dividen en dos categorías principales: Codificadores de Forma de Onda (dominio temporal y frecuencial) y Codificadores de Voz conocidos como *Vocoders*.
Los codificadores de *forma de onda* realizan de manera eficiente la compresión explotando las características temporales y/o espectrales de la señal de voz. Sin embargo, para obtener tal eficiencia, la tasa de bits necesaria para su transmisión y almacenamiento es muy alta. En cambio, en los *Vocoders*, la representación de la señal de voz es por medio de un conjunto de parámetros estimados mediante un modelo de producción de voz y una codificación en forma digital de éstos. Para la transmisión y almacenamiento de estos parámetros, los Vocoders resultan ser los más óptimos, ya que la tasa de bits a usar es mínima [5].
Los Vocoders se basan en la representación de la señal de voz mediante un modelo Todo Polo del sistema del tracto vocal. En la producción de voz sintética para segmentos de voz con naturaleza voceada, la excitación es un tren de impulsos con un período igual al período de pitch. En el caso de segmentos no voceados, la excitación es una secuencia de ruido blanco. En ambos casos la ganancia del segmento es estimada (figura 3.1).

En el presente capítulo se describirán las técnicas clásicas más usuales de Vocoders con una tasa de transmisión de 2400 a 9600 bps, que garantizan una calidad de reconstrucción aceptable [5]. Los diferentes Vocoders que se describen en este capítulo estiman los parámetros del modelo mediante segmentos de la señal de voz. A este proceso se le llama de *análisis*. Mientras que al proceso de reconstrucción de la señal de voz a partir de los parámetros estimados se le conoce como proceso de *síntesis*. Además, en este capítulo se describe el método de predicción lineal y su solución, enfocándonos a la predicción y estimación de parámetros de una señal de voz.

3.1. Vocoder Channel

El Vocoder Channel, en el proceso de análisis (figura 3.2), emplea un banco de filtros paso - banda con anchos de banda entre los 100 Hz y los 300 Hz de tal manera que puede estimar las variaciones temporales lentas de las magnitudes espectrales en cada banda. Comúnmente se usan

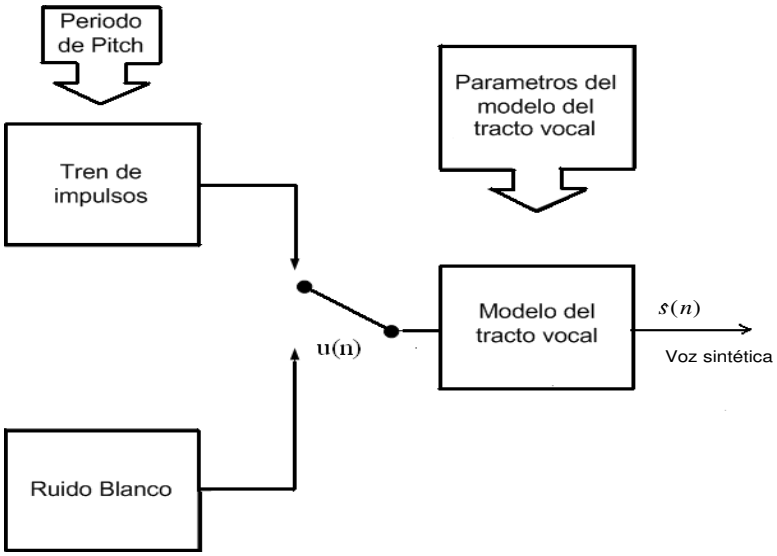

Figura 3.1: Modelo de Producción de voz.

de 16 a 20 filtros tipo de respuesta finita al impulso (FIR) de fase lineal para cubrir el ancho de banda de la voz natural (0 - 4 kHz). Para las bandas en baja frecuencia se emplean filtros de banda estrecha mientras que en las bandas de alta frecuencia se utilizan filtros de banda ancha. La salida de cada filtro es rectificada y filtrada con un filtro paso - bajas. En éste tipo de Vocoders se implanta un detector para determinar la naturaleza de la señal de excitación (voceada o no voceada) y un estimador del pitch [5].
En el proceso de síntesis (figura 3.3), la señal de voz es reconstruida mediante la señal codificada digitalmente, la cual se multiplica por las fuentes de excitación dependiendo de su naturaleza y las señales resultantes se pasan a través de su correspondiente filtro paso - banda, de tal manera que las salidas son sumadas para formar la señal de voz sintetizada.
Cabe mencionar que el Vocoder Channel es el primero y más antiguo Vocoder que se ha estudiado e implantado. Su implantación con técnicas modernas de procesamiento de señales digitales proporcionan una calidad aceptable para aplicaciones de comunicaciones comerciales con una tasa de transmisión de 2400 bps [5].

3.2. Vocoder de Fase

El Vocoder de Fase estima la magnitud del espectro de los segmentos de voz mediante el uso de un banco de filtros. Sin embargo, en lugar de estimar el período del pitch, este Vocoder estima la derivada de la fase resultante en la salida de cada filtro [5].
En el proceso de análisis del Vocoder de Fase (figura 3.4), la señal de voz en cada banda de frecuencia es modulada al multiplicar por $cos(w_k n)$ y $sen(w_k n)$, donde w_k es la frecuencia de traslación. Posteriormente, la señal modulada es pasada a través de un filtro paso - bajas con ancho de banda de aproximadamente 50 Hz. Las frecuencias del modulador son espaciadas cada 100 Hz, de tal manera que la implantación del Vocoder utiliza de 25 a 30 filtros. La salida del filtro paso - bajas es usada para calcular la magnitud y la derivada de la señal en cada banda.

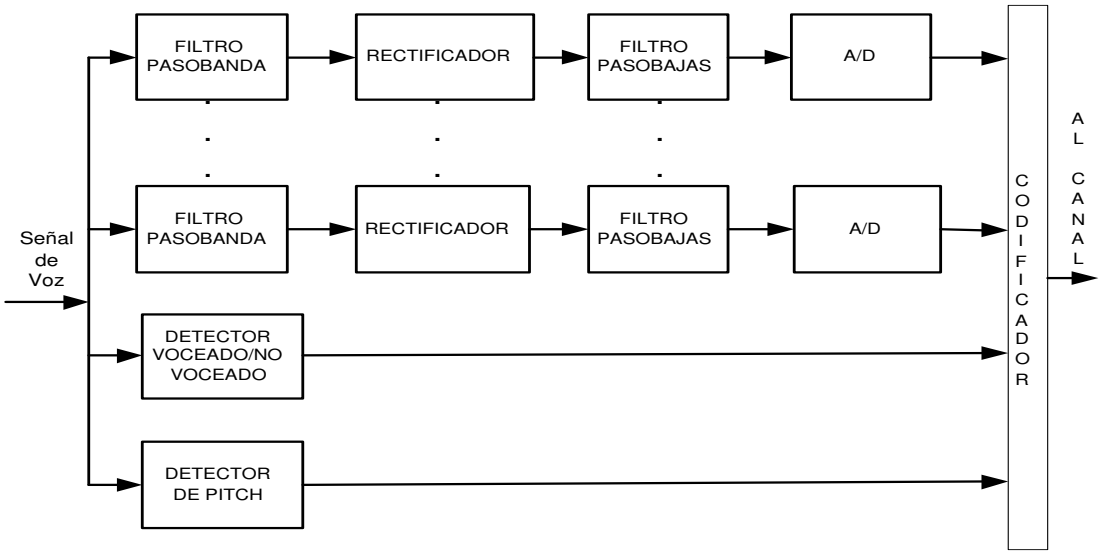

Figura 3.2: Diagrama de Bloques del Analizador del Vocoder Channel.

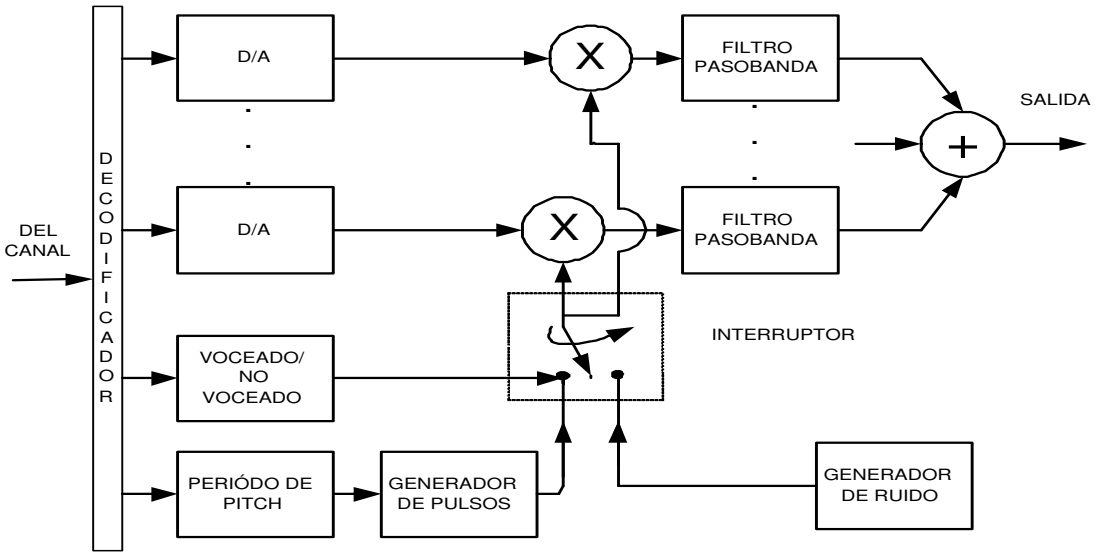

Figura 3.3: Diagrama de Bloques del Sintetizador del Vocoder Channel.

Estas bandas son muestreadas a razón de 50 a 60 muestras por segundo. Dichas muestras de la magnitud y la derivada son codificadas y transmitidas al receptor.
En cuanto al proceso de síntesis (figura 3.5), la información de la derivada de la fase es integrada para después formar con la magnitud de la señal y la fase resultante dos componentes de la señal en fase y en cuadratura. Las componentes resultantes son interpoladas y trasladadas en frecuencia al multiplicar cada una de éstas por $cos(w_k n)$ y $sen(w_k n)$. Estas componentes son sumadas para formar la señal de voz en cada banda de frecuencia [5].

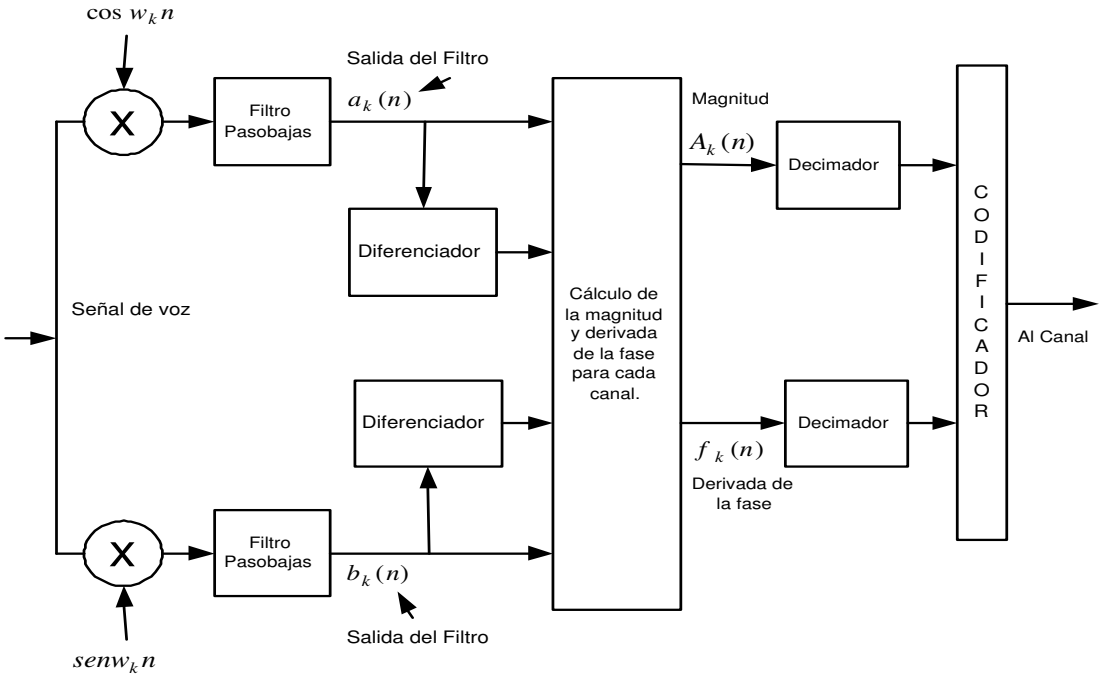

Figura 3.4: Diagrama de Bloques de un k_{th} canal del Analizador del Vocoder de Fase.

3.3. Vocoder de Formantes

Un Vocoder de Formantes puede ser visto como una clase de vocoder Channel que estima las primeras tres o cuatro frecuencias formantes[1] en un segmento de la señal de voz y sus correspondientes anchos de banda. Por tanto, con la información anterior y el período de pitch se codifica y transmite hacia el receptor. Cada segmento dado de la señal de voz, en cada frecuencia formante, puede ser caracterizado por un filtro de segundo orden de la forma [5]:

$$\hat{H}_k(z) = \frac{\hat{H}_k}{(1-\rho_k e^{j\omega_k} z^{-1})(1-\rho_k e^{-j\omega_k} z^{-1})} \quad \text{k=1,2,3,4,} \tag{3.1}$$

[1]Ver capítulo dos.

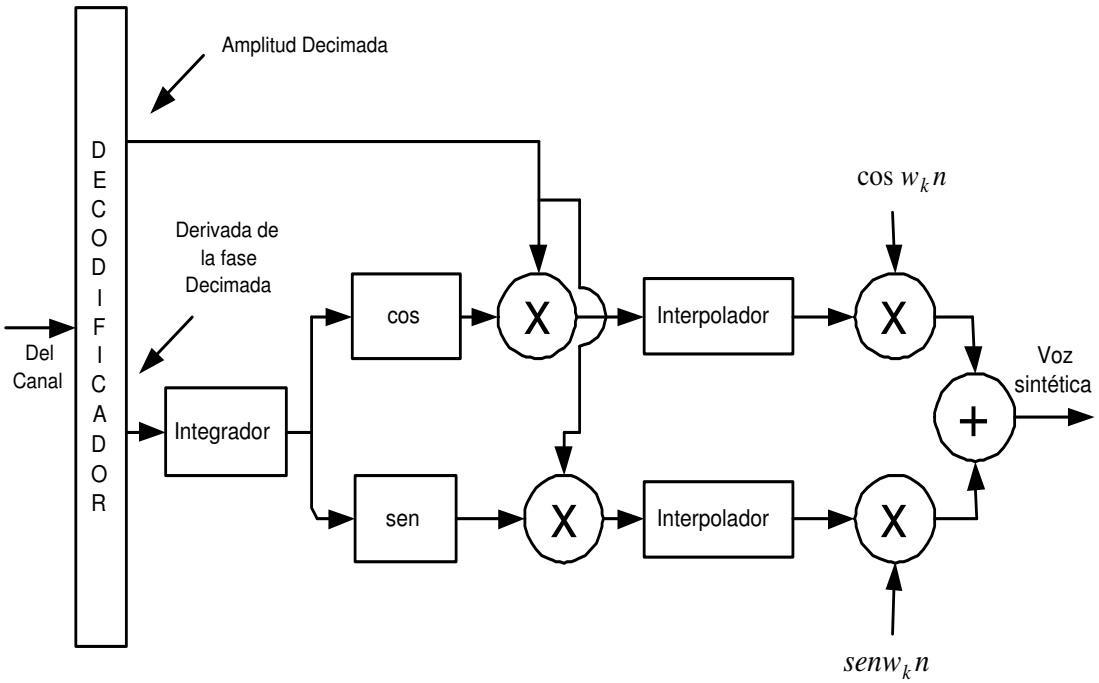

Figura 3.5: Diagrama de Bloques de un k_{th} canal del Sintetizador del Vocoder de Fase.

Donde $\hat{H}_k$ es un factor de ganancia, ρ_k es el valor de la distancia del par de polos conjugados complejos desde el origen de la circunferencia unitaria y $\omega_k = 2\pi F_k T$. La frecuencia del formante k_{th} se representa por F_k y T es el período de muestreo en segundos. El ancho de banda puede determinarse por la distancia del polo a la circunferencia unitaria. Las frecuencias formantes pueden ser estimadas por medio de un análisis de predicción lineal o por un análisis cepstral. El sintetizador para el Vocoder de Formantes puede implantarse mediante una configuración en cascada de filtros de segundo orden, donde cada filtro de segundo orden corresponde a un formante. Para el caso de segmentos no voceados podemos simplificar el sintetizador del Vocoder con tan sólo utilizar dos secciones del filtrado, es decir, considerando sólo dos frecuencias formantes. En la figura (3.6) se muestra el sintetizador del Vocoder de Formantes.

3.4. Vocoder Cepstral

La técnica empleada por este Vocoder consiste en separar el espectro de la excitación y el espectro del tracto vocal aplicando la transformada discreta de Fourier Inversa (IDFT) al logaritmo de las magnitudes espectrales para producir lo que se conoce como *el cepstrum*[2] de la señal. Los coeficientes cepstrum asociados al tracto vocal corresponden a las componentes de bajas frecuencias de la envolvente espectral, mientras que los coeficientes de excitación corresponden a las componentes de altas frecuencias, de tal manera que estos coeficientes pueden ser separados mediante una operación de filtrado lineal [7]. El período de pitch es estimado por medio del

[2]Para extraer la envolvente espectral del tracto vocal se requiere una técnica que remueva los efectos del período de pitch, dicha técnica es conocida como *Truncación Cepstrum* [7].

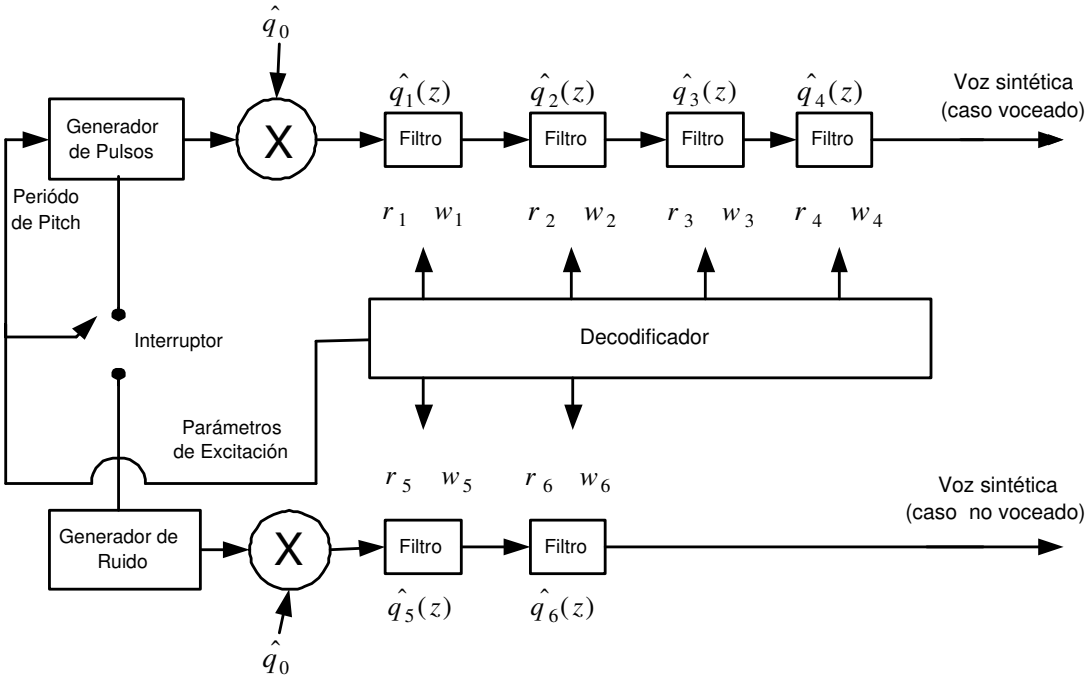

Figura 3.6: Diagrama de Bloques del Sintetizador del Vocoder de Formantes.

primer pulso en el cepstrum y la presencia o ausencia de pulsos fuertes indican la naturaleza del segmento analizado (voceado / no voceado). La figura (3.7) muestra el proceso de análisis y de síntesis del Vocoder Cepstral. En el proceso de síntesis, a los coeficientes cepstrum del tracto vocal se les aplica la transformada de Fourier, seguido de una ponderación exponencial para después aplicarles la transformada de Fourier Inversa y así producir la respuesta impulso del tracto vocal. Convolucionando la respuesta impulso del tracto vocal con la señal de excitación (ruido blanco o un tren de pulsos periódicos) se reconstruye la señal original.

3.5. Vocoder por codificación de predicción lineal

Es el tipo de Vocoder más utilizado debido a su simplicidad de implantación. Este Vocoder se basa en el modelo digital de la producción de voz[3] (Modelo terminal - análogo) mostrado en la figura (2.6), de tal manera que desacoplando los efectos producidos por el modelo del pulso glotal y el modelo de radiación debido a los labios, podemos simplificar tal modelo en un filtro Todo Polo conocido como modelo filtro - fuente como se muestra en la figura (3.1) [5]. Al reducir el modelo digital de la producción de la voz a un sistema Todo Polo resulta un sistema modelado por ecuaciones lineales cuyos parámetros son estimados por medio del método de predicción lineal, simplificando enormemente la modelización de la voz. Por tanto, la función de transferencia que caracteriza al modelo Todo Polo se muestra en la ecuación (3.2):

[3]Descrito en el capítulo 2.

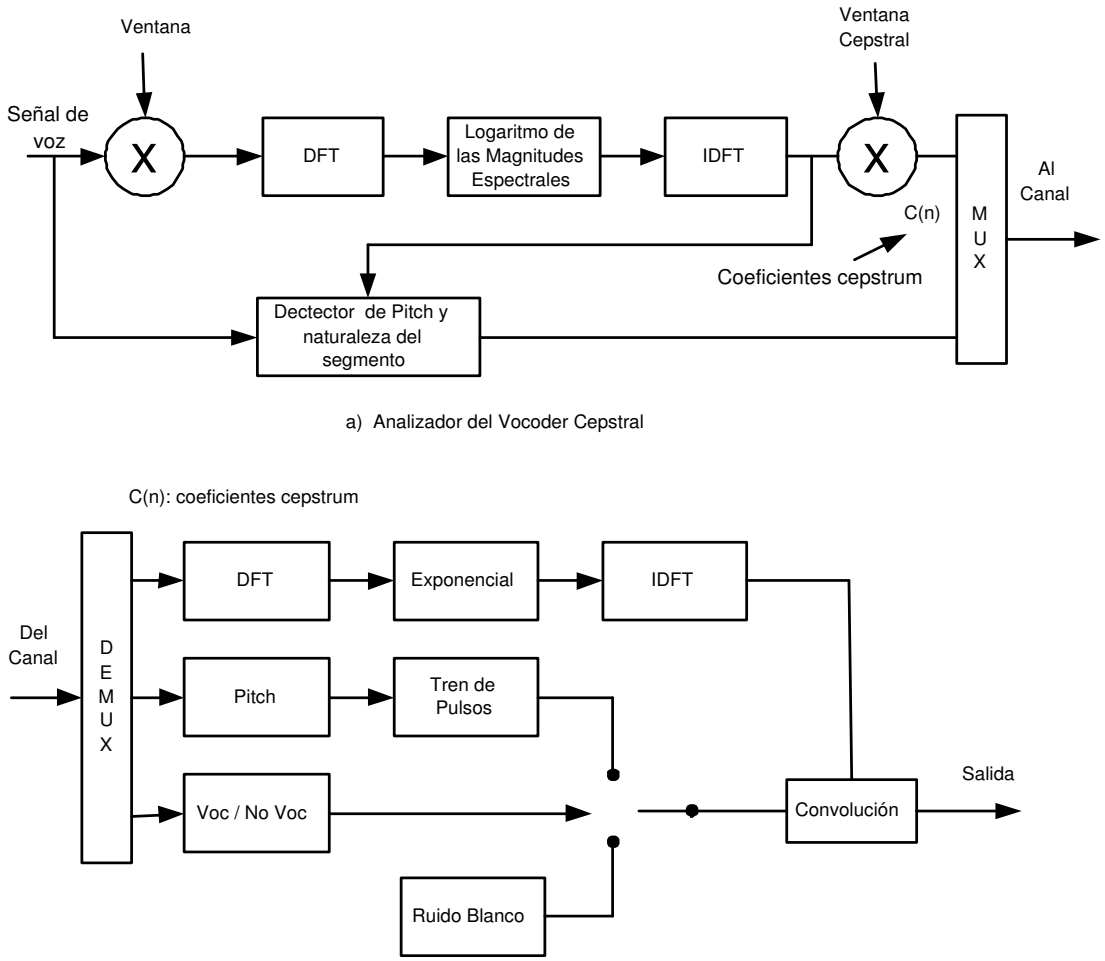

Figura 3.7: a) Diagrama de Bloques del Analizador del Vocoder Cepstral b) Sintetizador del Vocoder Cepstral.

$$H(z) = \frac{G}{1 + \sum_{k=1}^{p} a_k z^{-k}} = \frac{G}{\prod_{k=1}^{p} (1 - p_k z^{-1})} \tag{3.2}$$

Donde G representa una ganancia global, p_k los polos complejos que caracterizan al sistema y p el orden de la predicción. La ecuación en diferencias del modelo Todo Polo se presenta en la ecuación (3.3):

$$\widehat{s}(n) = -\sum_{k=1}^{p} a_k \widehat{s}(n-k) + u(n) \tag{3.3}$$

Donde $\widehat{s}(n)$ es la señal estimada de salida, $u(n)$ la entrada respectiva según la naturaleza del segmento y p el orden de la predicción lineal, como se muestra en la figura (3.1).

De la ecuación (3.3) se supone que cada muestra es una combinación lineal de las muestras anteriores. Los coeficientes del filtro $\{a_k\}$ (coeficientes de predicción lineal) se calculan al minimizar el error entre la muestra actual y su predicción.
Este Vocoder trabaja sobre segmentos de 20 a 30 ms de voz, por lo que se supone que sus estimaciones estadísticas no varían en dicho intervalo temporal (propiedad de estacionaridad). Los segmentos de voz se analizan para determinar los coeficientes de predicción, la naturaleza del segmento, las estimación del período del pitch y la ganancia de éste. Para la reconstrucción de la señal son utilizados los parámetros estimados por la predicción lineal que definen al filtro Todo Polo tal que la entrada de excitación de éste, dependerá de la naturaleza del segmento (voceado-tren de impulsos, no voceado-ruido blanco). El esquema básico del proceso de síntesis del vocoder por predicción lineal es el modelo filtro - fuente mostrado en la figura (3.1).

3.6. Método de Predicción Lineal

Una de las técnicas más poderosas en el análisis de señales de voz es el *método de predicción lineal*. Con este método es posible estimar los parámetros de la señal de voz que modelan al sistema de producción de la voz [8]. El proceso de análisis por predicción lineal [10], [11] consiste en aproximar las muestras a partir de una combinación lineal de muestras anteriores de la señal de voz. Al minimizar el promedio del error al cuadrado (en un intervalo de duración finita) entre la señal original y la señal estimada, se obtiene un conjunto único de coeficientes que ponderarán a la combinación lineal. Es común en la literatura que el método de predicción lineal se le denomine *codificación de predicción lineal (LPC)*.

3.6.1. Predicción Lineal por el Método de Autocorrelación

Mediante el método de la predicción lineal para la estimación de los parámetros del modelo Todo Polo [10], [11] consideramos segmentos de la señal de 10 a 30 ms [5], para garantizar que el segmento en estudio cumpla con las condiciones de estacionaridad [4] y ergódicidad[5].

Considerando el error de la predicción (figura 3.8 y 3.9):

$$e(n) = s(n) - \widehat{s}(n) \tag{3.4}$$

$$\widehat{s}(n) = -\sum_{k=1}^{p} a_k s(n-k) \tag{3.5}$$

$$e(n) = s(n) + \sum_{k=1}^{p} a_k s(n-k) \tag{3.6}$$

[4]Un proceso aleatorio es estacionario si sus estimaciones estadísticas no varían con el tiempo [5].
[5]Un proceso aleatorio es ergódico si sus estimaciones estadísticas puede ser aproximadas por sus estimaciones temporales [5].

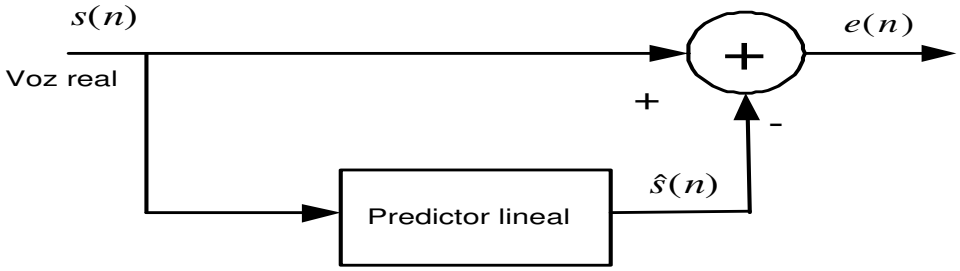

Figura 3.8: Error de predicción.

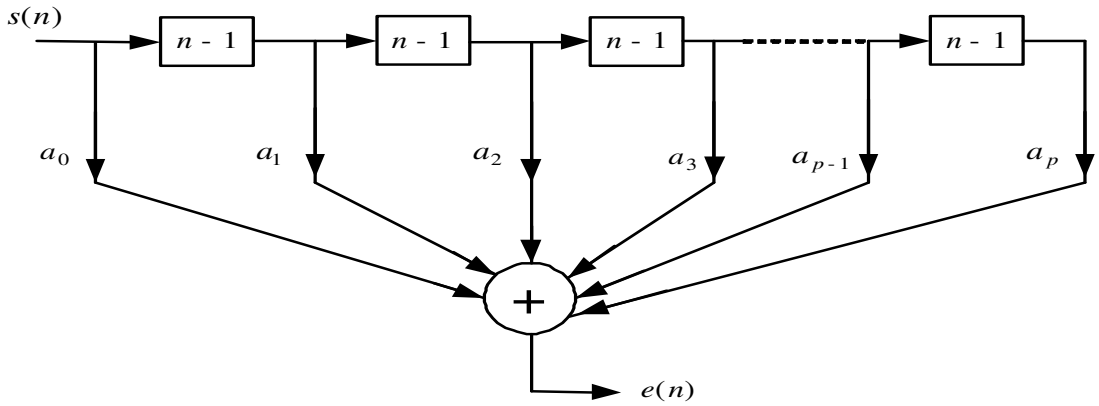

Figura 3.9: Filtro de predicción del error.

Donde $s(n)$ es la señal original, $\widehat{s}(n)$ es la señal estimada, $e(n)$ es la señal de error, $\{a_i\}$ el conjunto de coeficientes de la predicción y p el orden de la predicción. Mostrando la ecuación (3.6) en notación vectorial (ecuación 3.7):

$$e(n) = s(n) + A_P^T s_{p_i} \tag{3.7}$$

$$A_P^T = \begin{bmatrix} a_1 & a_2 & a_3 & .... & a_p \end{bmatrix}$$

$$s_{p_i} = \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ \vdots \\ s_p \end{bmatrix}$$

Donde el superíndice T significa el transpuesto del vector. Aplicando el criterio del error cuadrático a la ecuación (3.7), resulta la ecuación (3.8):

$$\begin{aligned} e^2(n) &= \left(s(n) + A_P^T s_{p_i}\right)\left(s(n) + A_P^T s_{p_i}\right) \\ &= s^2(n) + s(n)A_P^T s_{p_i} + A_P^T s_{p_i} s(n) + A_P^T s_{p_i} A_p^T s_{p_i} \\ &= s^2(n) + 2s(n)A_p^T s_{p_i} + A_p^T s_{p_i} s_{p_i}^T A_p \end{aligned} \tag{3.8}$$

Considerando a la señal de voz $s(n)$ como estacionaria en el sentido amplio (WSS), obtenemos la esperanza matemática $E\{\}$ de la ecuación (3.8):

$$\begin{aligned} E\left\{e^2(n)\right\} &= E\left\{s^2(n) + 2s(n)A_p^T s_{p_i} + A_p^T s_{p_i} s_{p_i}^T A_p\right\} \\ &= r_p(0) + 2A_p^T E\left\{s(n)s_{p_i}\right\} + A_p^T E\left\{s_{p_i} s_{p_i}^T\right\} A_p \end{aligned} \tag{3.9}$$

Donde:

$$r_p(0) = E\{s^2(n)\} = \text{Energía del segmento de voz} \tag{3.10}$$

Expresando al vector de autocorrelación de la señal $s(n)$ como:

$$r_p(i) = E\left\{s(n)s_{p_i}\right\} = \left\{ \begin{array}{ccccc} r_1 & r_2 & r_3 & .... & r_p \end{array} \right\} \tag{3.11}$$

Y a la ecuación (3.12) como la matriz de autocorrelación, siendo ésta una matriz del tipo Toeplitz simétrica[6].

$$R_p = E\left\{s_{p_i} s_{p_i}^T\right\} = \left[\begin{array}{ccccc} r_0 & r_1 & r_2 & ... & r_p \\ r_1 & r_0 & r_1 & ... & r_{p-1} \\ r_2 & r_1 & r_0 & .. & r_{p-2} \\ \vdots & & & & \vdots \\ r_p & r_{p-1} & r_{p-2} & ... & r_0 \end{array} \right] \tag{3.12}$$

Entonces, la esperanza matemática del error cuadrático se expresa como muestra la ecuación (3.13):

$$E\left\{e^2(n)\right\} = r_p(0) + 2A_p^T r_p + A_P^T R_p A_p \tag{3.13}$$

Aplicando el criterio de optimización sobre $E\{e^2(n)\}$ resulta la ecuación (3.14):

$$\frac{\partial E\left\{e^2(n)\right\}}{\partial A_p} = 0 + 2r_p + 2R_p A_p = 0$$

$$R_p A_p = -r_p \tag{3.14}$$

$$A_p = -R_p^{-1} r_p \tag{3.15}$$

A la ecuación (3.15) se le denomina *ecuación de Wiener - Hopf* o *ecuación Normal*, dado que el error es perpendicular al espacio vectorial s_p. La ecuación (3.15) es la solución obtenida al minimizar el promedio del error cuadrático entre la señal $s(n)$ y la señal estimada $\widehat{s}(n)$ y de esta manera definimos el método de la autocorrelación para la resolución de la codificación por

[6]Si una matriz Toeplitz es simétrica o Hermitiana (caso complejo), entonces todos los elementos de la matriz son completamente determinados por la primera fila o columna de ésta [12]. Por ejemplo:

$$A = \left[\begin{array}{cccc} 1 & 3 & 5 & 7 \\ 3 & 1 & 3 & 5 \\ 5 & 3 & 1 & 3 \\ 7 & 5 & 3 & 1 \end{array} \right]$$

predicción lineal. Todavía queda encontrar el conjunto de valores de A_p, que son los coeficientes que ponderan a la combinación lineal que define la predicción lineal, para ello, en la siguiente sección presentamos un algoritmo que resuelve lo anterior, llamado *algoritmo de Levinson - Durbin*.

3.6.2. Solución de la Ecuación Normal

Para la solución de la ecuación normal (ecuación 3.15) es necesario invertir la matriz R_p, considerando que el cálculo del conjunto de valores de A_p se tendrá que hacer con el mínimo de operaciones para justificar su empleo en un sistema de síntesis de voz en tiempo Real. El algoritmo de Levinson - Durbin es una manera eficiente para la solución de la ecuación normal contemplando que R_p es una matriz del tipo Toeplitz.

Algoritmo de Levinson - Durbin

El *algoritmo de Levinson - Durbin* es un procedimiento computacionalmente eficiente para resolver las ecuaciones normales (ecuación 3.15) para los coeficientes de predicción. Este algoritmo explota la simetría especial en la matriz de autocorrelación (ecuación 3.12) de tal modo que $R_p(i,j) = R_p(i+1, j+1)$, así la matriz de autocorrelación es una *matriz Toeplitz*. Además, $R_p(i,j) = R_p^*(j,i)$, la matriz es también *Hermitiana* (caso complejo) o simétrica (caso real) [13]. La clave de la solución del algoritmo de Levinson - Durbin consiste en proceder recursivamente, empezando con un predictor de orden $p = 1$ y después incrementar el orden recursivamente, usando las soluciones de orden menor para obtener la solución al siguiente orden superior. Entonces, el algoritmo de Levinson - Durbin propone una solución al orden $(p+1)$ con base a la solución al orden p [12].

Para el desarrollo de la recursión de Levinson - Durbin [12] requerimos resolver la ecuación normal que se puede presentar como:

$$r_p(k) + \sum_{l=1}^{p} a_p(l) r_p(k-l) = 0 \qquad k = 1, 2, \ldots, p \tag{3.16}$$

Donde el error cuadrático del modelo de predicción es:

$$\varepsilon_p = r_p(0) + \sum_{l=1}^{p} a_p(l) r_p(l) \tag{3.17}$$

Combinando las ecuaciones (3.16) y (3.17) se forma la matriz:

$$\begin{bmatrix} r_p(0) & r_p^*(1) & r_p^*(2) & \ldots & r_p^*(p) \\ r_p(1) & r_p(0) & r_p^*(1) & \ldots & r_p^*(p-1) \\ r_p(2) & r_p(1) & r_p(0) & \ldots & r_p^*(p-2) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ r_p(p) & r_p(p-1) & r_p(p-2) & \ldots & r_p(0) \end{bmatrix} \begin{bmatrix} 1 \\ a_p(1) \\ a_p(2) \\ \vdots \\ a_p(p) \end{bmatrix} = \varepsilon_p \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{3.18}$$

La ecuación (3.18) puede escribirse de manera equivalente como:

$$R_p A_p = \varepsilon_p u_1 \tag{3.19}$$

Donde R_p es una matriz Toeplitz Hermitiana de dimensiones $(p+1)X(p+1)$ y $u_1 = [1\,0\,0\,\ldots\,0]^T$ es un vector unitario. En el caso especial para valores reales la matriz R_p es una matriz Toeplitz simétrica [12].

Para resolver la ecuación (3.19) es necesario un algoritmo recursivo, donde los coeficientes a_{j+1} del modelo Todo Polo de orden $(j+1)$ son encontrados mediante los coeficientes a_j del modelo de j-polos. Entonces, dado a_j queremos calcular la solución de las ecuaciones normales del orden $(j+1)$:

$$R_{j+1}A_{j+1} = \varepsilon_{j+1}u_1 \tag{3.20}$$

El procedimiento para la solución supone que se agrega un elemento con valor de cero al vector a_j y se multiplica el vector resultante por R_{j+1}:

$$\begin{bmatrix} r_p(0) & r_p^*(1) & r_p^*(2) & \ldots & r_p^*(j) & r_p^*(j+1) \\ r_p(1) & r_p(0) & r_p^*(1) & \ldots & r_p^*(j-1) & r_p^*(j) \\ r_p(2) & r_p(1) & r_p(0) & \ldots & r_p^*(j-2) & r_p^*(j-1) \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ r_p(j) & r_p(j-1) & r_p(j-2) & \ldots & r_p(0) & r_p^*(1) \\ r_p(j+1) & r_p(j) & r_p(j-1) & \ldots & r_p(1) & r_p^*(0) \end{bmatrix} \begin{bmatrix} 1 \\ a_j(1) \\ a_j(2) \\ \vdots \\ a_j(j) \\ 0 \end{bmatrix} = \begin{bmatrix} \varepsilon_j \\ 0 \\ 0 \\ \vdots \\ 0 \\ \gamma_j \end{bmatrix} \tag{3.21}$$

Donde el parámetro γ_j está definido por:

$$\gamma_j = r_p(j+1) + \sum_{i=1}^{j} a_j(i) r_p(j+1-i) \tag{3.22}$$

El paso clave para la determinación de la recursión de Levinson - Durbin es notar la propiedad de que R_{j+1} es una matriz Toeplitz Hermitiana, de tal manera que la ecuación (3.21) para un proceso WSS puede reescribirse como se muestra en la ecuación (3.23):

$$\begin{bmatrix} r_p(0) & r_p(1) & r_p(2) & \ldots & r_p(j) & r_p(j+1) \\ r_p^*(1) & r_p(0) & r_p(1) & \ldots & r_p(j-1) & r_p(j) \\ r_p^*(2) & r_p^*(1) & r_p(0) & \ldots & r_p(j-2) & r_p(j-1) \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ r_p^*(j) & r_p^*(j-1) & r_p^*(j-2) & \ldots & r_p^*(0) & r_p(1) \\ r_p^*(j+1) & r_p^*(j) & r_p^*(j-1) & \ldots & r_p^*(1) & r_p(0) \end{bmatrix} \begin{bmatrix} 0 \\ a_j(j) \\ a_j(j-1) \\ \vdots \\ a_j(j) \\ 1 \end{bmatrix} = \begin{bmatrix} \gamma_j \\ 0 \\ 0 \\ \vdots \\ 0 \\ \varepsilon_j \end{bmatrix} \tag{3.23}$$

Tomando los valores complejos conjugados de la ecuación (3.23) y multiplicándola por una constante (compleja) k_{j+1}, resulta una ecuación que al sumarle la ecuación (3.21), es como se muestra en la ecuación (3.24) :

$$R_{j+1}\left\{ \begin{bmatrix} 1 \\ a_j(1) \\ a_j(2) \\ \vdots \\ a_j(j) \\ 0 \end{bmatrix} + k_{j+1} \begin{bmatrix} 0 \\ a_j^*(j) \\ a_j^*(j-1) \\ \vdots \\ a_j^*(1) \\ 1 \end{bmatrix} \right\} = \begin{bmatrix} \varepsilon_j \\ 0 \\ 0 \\ \vdots \\ 0 \\ \gamma_j \end{bmatrix} + k_{j+1} \begin{bmatrix} \gamma_j^* \\ 0 \\ 0 \\ \vdots \\ 0 \\ \varepsilon_j^* \end{bmatrix} \tag{3.24}$$

Los coeficientes k_j, mejor conocidos como los coeficientes de reflexión (PARCOR) [4], deberán cumplir que $|k_j| < 1$ para que el sistema sea estable, de tal manera que todos los polos siempre estarán dentro del circulo unitario en el plano $\mathcal{Z}$. Dado que queremos encontrar al vector a_{j+1}, el cual es multiplicado por R_{j+1}, desarrollando la ecuación (3.24), de la última fila se observa que:

$$k_{j+1} = -\frac{\gamma_j}{\varepsilon_j^*} \tag{3.25}$$

Debido a que de la primera fila del sumando derecho de la ecuación (3.24) es igual a $\varepsilon_j + k_{j+1}\gamma_j^*$ y de la última fila del sumando derecho de la ecuación (3.24) es igual a $\gamma_j + k_{j+1}\varepsilon_j^*$, de tal manera que sustituyendo la ecuación (3.25) en esta ultima relación, la ecuación (3.24) puede expresarse como:

$$R_{j+1}a_{j+1} = \varepsilon_{j+1}u_1 \tag{3.26}$$

Donde

$$a_{j+1} = \begin{bmatrix} 1 \\ a_j(1) \\ a_j(2) \\ \vdots \\ a_j(j) \\ 0 \end{bmatrix} + k_{j+1} \begin{bmatrix} 0 \\ a_j^*(j) \\ a_j^*(j-1) \\ \vdots \\ a_j^*(1) \\ 1 \end{bmatrix}$$

Tal que es la solución de la ecuación normal de orden $(j+1)$, donde:

$$\varepsilon_{j+1} = \varepsilon_j + k_{j+1}\gamma_j^* = \varepsilon_j[1 - |k_{j+1}|^2] \tag{3.27}$$

Donde ε_{j+1} es el error cuadrático de orden (j+1). Si definimos a $a_j(0) = 1$ y $a_j(j+1) = 0$, entonces la ecuación (3.26) es referida como la ecuación de Levinson - Durbin de orden actualizado [12] y puede expresarse como:

$$a_{j+1}(i) = a_j(i) + k_{j+1}a_j^*(j-i+1) \qquad i = 0, 1, \ldots, j+1 \tag{3.28}$$

con las condiciones iniciales de $a_0(0) = 1$ y $\varepsilon_0 = r_p(0)$. Por tanto para $j = 0, 1, \ldots, p-1$, el modelo de orden (j+1) es encontrado a partir del modelo de orden j en tres pasos.

Antes de realizar estos tres pasos se calcula $p+1$ autocorrelaciones de la señal de voz $r_p(l) = \{r_p(0), r_p(1), \cdots, r_p(p+1)\}$ donde los $r(p)$ están normalizados con respecto a la energía de la señal de voz en un segmento:

$$r(0) = \frac{r_p(0)}{r_p(0)} = 1,\ r(1) = \frac{r_p(1)}{r_p(0)},\ r(2) = \frac{r_p(2)}{r_p(0)},\ \cdots,\ r(i) = \frac{r_p(i)}{r_p(0)} < 1$$

El organigrama del algoritmo de la recursión de Levinson - Durbin se muestra en el cuadro (3.1):

1) Iniciar con:
a) $a_0(0) = 1 = k_0$
b) $\varepsilon_0 = r_0$
2) Para j=0,1,2,$\cdots$, $p-1$
a) $\gamma_j = r_p(j+1) + \sum_{i=1}^{j} a_j(i) r(j-i+1)$
b) $k_{j+1} = -\frac{\gamma_j}{\varepsilon_j}$
c) Para i=1,2,3,$\cdots$, j
$a_{j+1}(i) = a_j(i) + k_{j+1} a_j(j-i+1)$
d) $a_{j+1}(j+1) = k_{j+1}$
e) $\varepsilon_{j+1} = \varepsilon_j(1 - k_{j+1}^2)$
3) $\mathrm{E}^2(0) = \varepsilon_j$

Cuadro 3.1: Organigrama del algoritmo de Levinson - Durbin

Donde E es error cuadrático y podemos considerarlo como una ponderación de la energía de la señal con los coeficientes de la predicción.

La relación entre los coeficientes de reflexión y el conjunto de parámetros a_i se puede expresar por las ecuaciones (3.29), (3.30), (3.31) y (3.32) [4]:

k_i a a_i:

$$a_i(i) = k_i \tag{3.29}$$

$$a_j(j) = a_{i-1}(j) + k_i a_{i-1}(i-j) \qquad \begin{matrix} i = 1, \ldots, p \\ j = 1, \ldots, i-1 \end{matrix} \tag{3.30}$$

a_i a k_i:

$$k_i = a_i(i) \tag{3.31}$$

$$a_{i-1}(j) = \frac{a_i(j) - a_i(i) a_{i-1}(i-j)}{1 - k_i^2} \qquad \begin{matrix} i = p, \ldots, 1 \\ j = 1, \ldots, i-1 \end{matrix} \tag{3.32}$$

En Síntesis de voz, el orden más aceptable y eficiente computacionalmente es el de $p = 10$ [14]. Una ventaja al emplear el algoritmo de Levinson - Durbin para su implantación en un sistema en tiempo real es que requiere de p^2 multiplicaciones y sumas para encontrar los coeficientes de la predicción, comparadas con p^3 operaciones por eliminación Gauss - Jordan, si no se explota la propiedad Toeplitz de la matriz de autocorrelación [13].

3.7. Resumen

En el presente capítulo se describieron los métodos más usuales de codificación para señales de voz (Vocoders), haciendo énfasis sólo en las características principales de éstos. El objetivo de presentar las anteriores técnicas de codificación es elegir el método más adecuado y óptimo para nuestro diseño. Uno de los objetivos principales del presente trabajo de tesis es el diseño y desarrollo de un sistema de síntesis de voz en tiempo real; por lo que fue necesario hacer una descripción de los algoritmos que integran a los Vocoders y con ello determinar cuál de ellos es el más rentable y óptimo para su implantación en tiempo real, eligiendo, por su simplicidad al Vocoder de Predicción Lineal.

Además, se describió el modelo simplificado del sistema de producción de la voz, siendo este el modelo Filtro - Fuente (Todo Polo). Para la estimación de los parámetros que caracterizan al modelo Todo Polo se emplea un método de predicción lineal, el cual, a grandes rasgos, es la predicción de la señal empleando el criterio del error cuadrático promedio; resultando un sistema matricial (ecuación normal) cuya solución corresponde a los coeficientes del filtro Todo Polo. El algoritmo descrito para la solución de la ecuación normal correspondiente al planteamiento de la codificación de predicción lineal por el método de Autocorrelación fue el algoritmo de Levinson - Durbin.

Capítulo 4

Diseño del Sistema de Síntesis de Voz

En el capítulo tres se presentó una breve descripción de las diferentes técnicas de codificación clásicas para señales de voz, mencionando sólo sus principales características y eligiendo como codificador a emplear, el Vocoder basado en predicción lineal. Aunque el método de codificación seleccionado genera una calidad de voz sintética, su implantación en un sistema en tiempo real requerirá menor tiempo de proceso y menor tasa de bits para la transmisión y/o almacenamiento de los parámetros estimados con respecto a los demás Vocoders. En este capítulo, se hará una descripción y análisis de los algoritmos usados para la realización del Vocoder basado en predicción lineal, mostrando los procesos que integran a dicho codificador tanto en el bloque de análisis como en el de síntesis.

En la figura (4.1) se presenta un esquema general del sistema de síntesis que engloba un *bloque de análisis*, donde se estimarán los parámetros de la señal en cuestión (coeficientes LPC, naturaleza del segmento, estimación del pitch, ganancia del segmento) y un *bloque de síntesis*, que mediante los parámetros obtenidos en el proceso de análisis permite reconstruir a la señal. En las siguientes secciones se describirán los bloques que integran a los procesos de análisis y de síntesis.

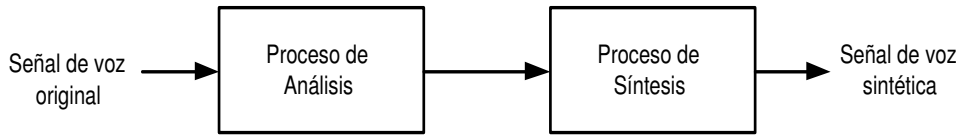

Figura 4.1: Esquema General de un Sistema de Síntesis de Voz.

4.1. Proceso de Análisis

El proceso de análisis consta de varios subprocesos como son: la decisión de la existencia de un segmento de silencio, el proceso de ventaneo, un filtrado de preénfasis, la decisión de la naturaleza de la ventana, la estimación del pitch, la estimación de los parámetros LPC y el cálculo de la ganancia del segmento. En la presente sección describiremos cada etapa del proceso de análisis con base al diagrama de flujo que se muestra en la figura (4.2).

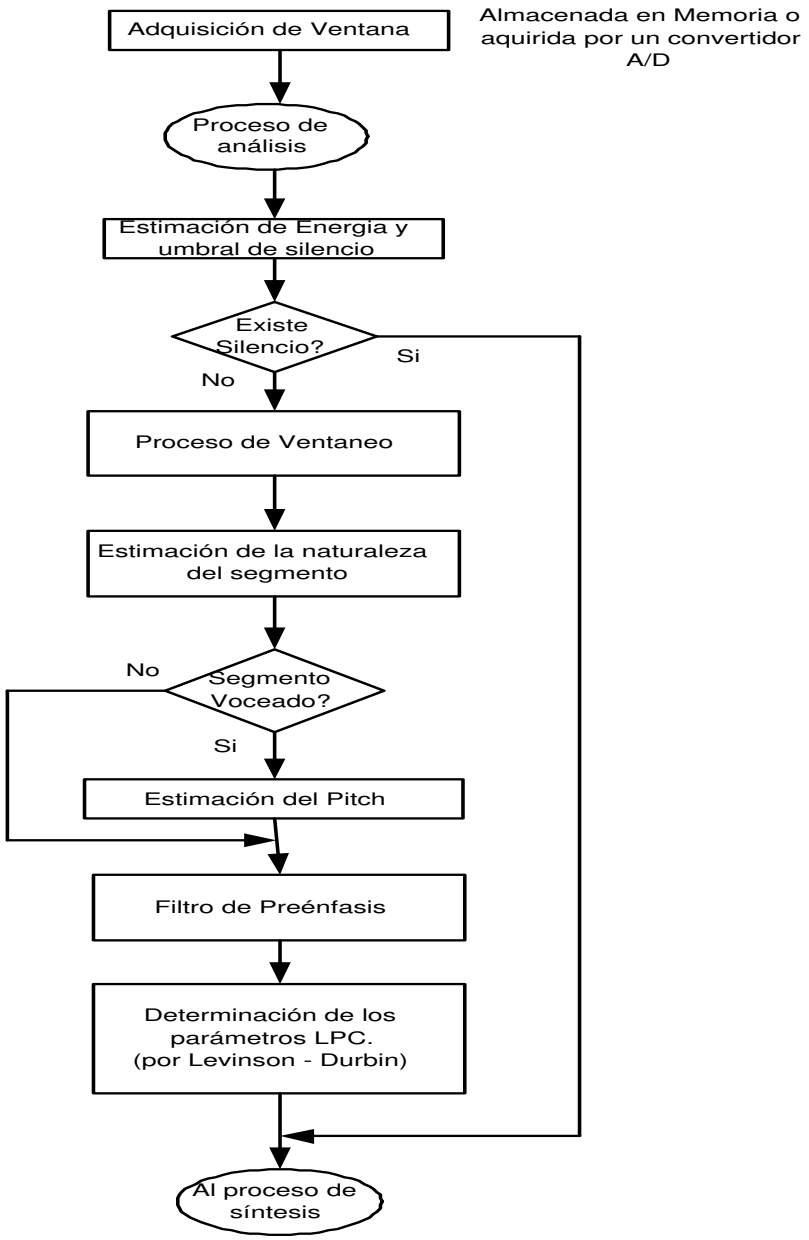

Figura 4.2: Diagrama de flujo del proceso de Análisis.

4.1.1. Estimación de Energía y Umbral de Silencio

Antes de estimar los parámetros necesarios en cada subproceso que integra al bloque de análisis es necesario determinar si el segmento de voz en estudio no es considerado como un segmento de silencio, es decir, mediante el cálculo de la energía de éste, y una comparación con un umbral (definido por pruebas experimentales en el ambiente), se determina si existe silencio o no. Una forma de calcular la energía del segmento de voz en estudio es por medio de la ecuación (4.1):

$$E = \sum_{i=0}^{N-1} |s(n)|^2 \tag{4.1}$$

Donde E es la energía del segmento, $s(n)$ son las muestras de un segmento de voz y N es el número de muestras para cada segmento. La elección del valor del umbral para la determinación de la existencia de un segmento de silencio se hace dependiendo de las condiciones de ruido en que se realice la adquisición de la voz. Cabe mencionar que si el segmento es considerado como de silencio no será necesario estimar los parámetros de las siguientes etapas en el proceso de análisis.

4.1.2. Proceso de Ventaneo

Como se mencionó en el capítulo tres, consideramos segmentos de la señal de 10 a 30 ms para garantizar que el segmento en estudio cumpla con las condiciones de estacionaridad y ergódicidad. Sin embargo, es necesario delimitar los segmentos de voz sin que existan cambios abruptos en éstos, es decir, tenemos que suavizar los limites temporales entre segmentos sin distorsionar la información espectral de éstos. Una solución es aplicar al segmento una ventana, es decir, multiplicar al segmento por una función que suavice los cambios abruptos temporales, conservando aproximadamente la información espectral de ésta. Existen diversas ventanas, como son: la ventana triangular (Bartlett), Hanning, Hamming, Blackman, Kaiser, Lanczos y Tukey. Se eligió a la ventana de Hamming por ser la más utilizada para síntesis de voz [5]. La ecuación (4.2) muestra la definición de la ventana de Hamming:

$$w(n) = 0.54 \text{ - } 0.46 \cos \frac{2\pi n}{N-1} \tag{4.2}$$

Donde N es la longitud de la ventana. El ancho de transición aproximado del lóbulo principal de la ventana es de $8\pi/N$ y la amplitud de la diferencia entre el primer y segundo lóbulo es de $-43\ dB$ [5]. En la figura (4.3) se muestra la ventana de Hamming tanto en el dominio temporal como en el frecuencial.

Entonces, por medio de un proceso de ventaneo es posible suavizar las transiciones entre los límites temporales de los segmentos a analizar sin afectar en gran medida el contenido frecuencial de éste. En la figura (4.4) se muestra un diagrama de la fragmentación de la señal en segmentos de voz ventaneados. En la figura (4.5.a) se muestra un segmento de voz real con $N = 256$ muestras del fonema /o/, mientras que en la figura (4.5.b) se muestra el segmento con ventana de Hamming.

De la figura (4.5.b) observamos que al aplicarle el proceso de ventaneo al segmento, suavizamos los cambios abruptos entre los límites temporales del segmento. En la figura (4.6) se muestra la

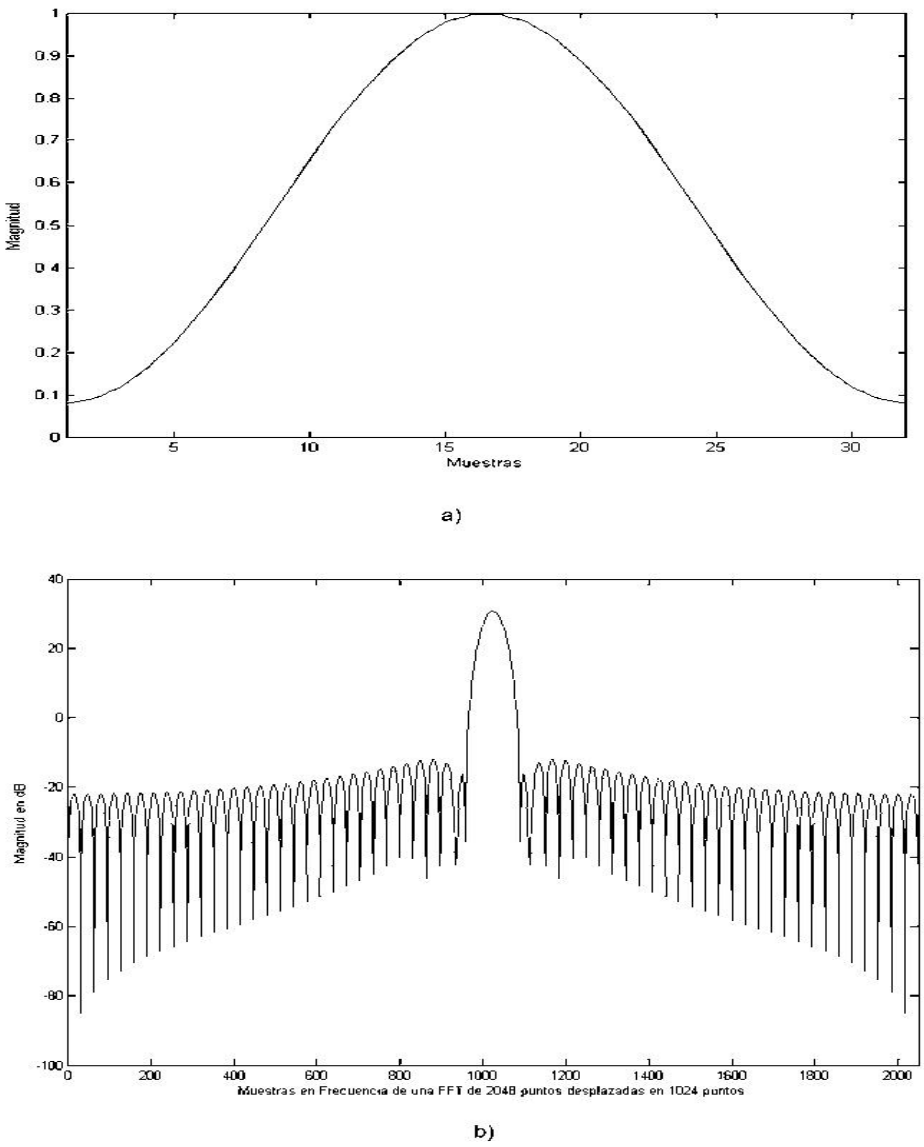

Figura 4.3: Ventana de Hamming. a) Dominio temporal b) Dominio frecuencial.

comparación entre el espectro del segmento de voz del fonema /o/ sin ventana y con ventana. También podemos realizar *traslapes* en la asignación de ventanas sucesivas, lo cual se hace para reconsiderar la información de los límites temporales que fueron atenuados. El porcentaje de traslape con respecto a la longitud de la ventana se recomienda entre la tercera y cuarta parte del segmento total [4]. En la figura (4.7) se muestra una señal de voz con una asignación de ventanas traslapadas.

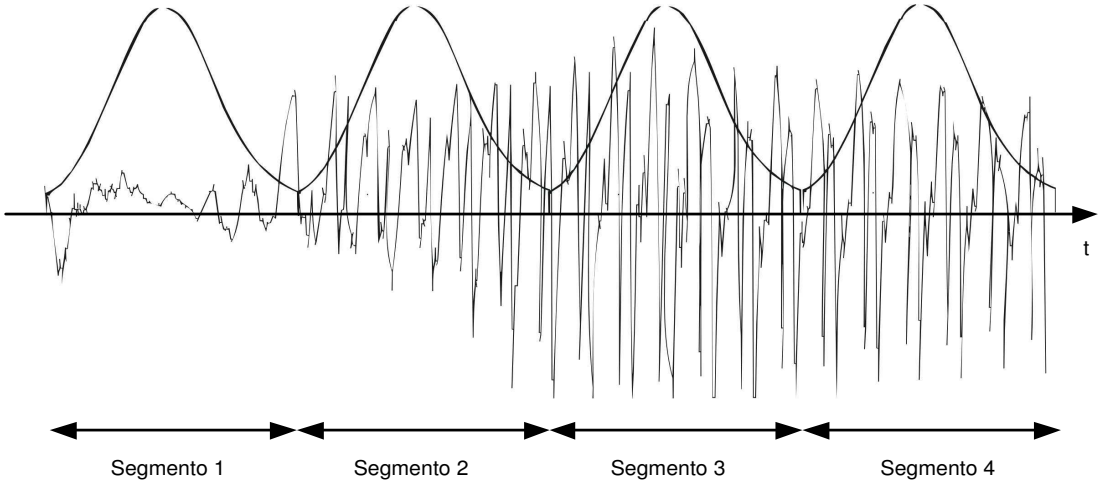

Figura 4.4: Diagrama de fragmentación de la señal en ventanas.

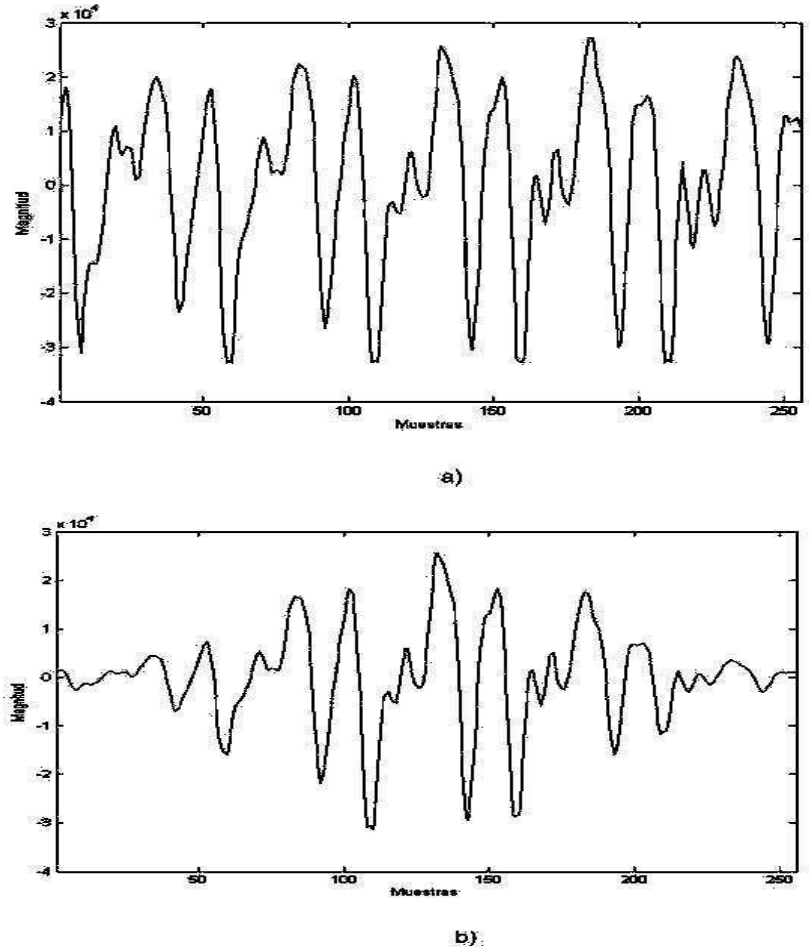

Figura 4.5: Segmento de Voz real N=256, fonema /o/. a) Sin ventana, b) Con ventana.

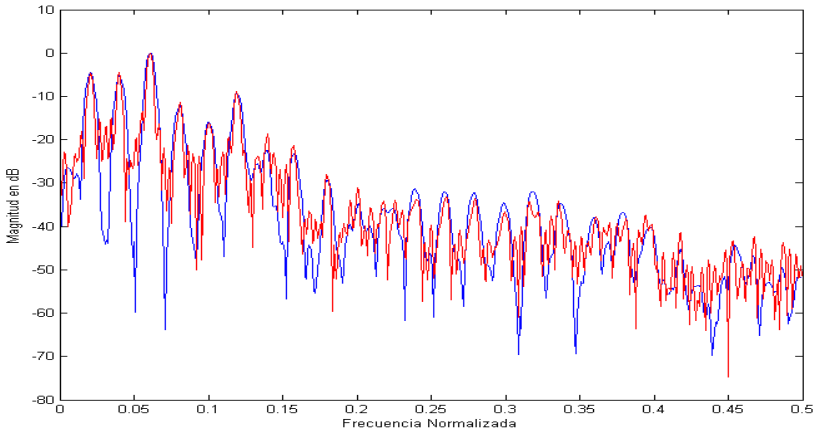

Figura 4.6: Espectro del Segmento de Voz real N=256, fonema /o/. a) Sin ventana (rojo), b) Con ventana (azul).

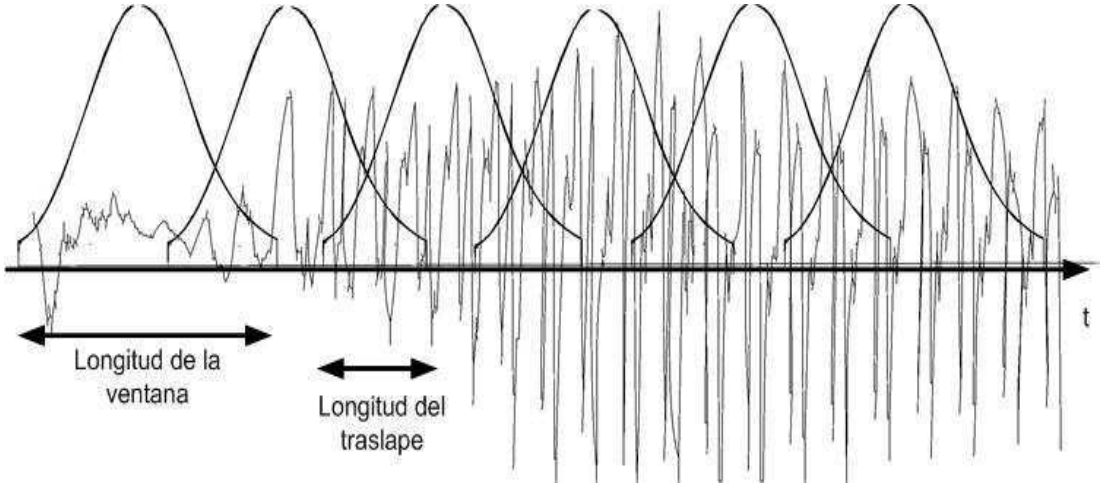

Figura 4.7: Asignación de ventanas sucesivas con traslape.

4.1.3. Filtro de Preénfasis

Aunque el modelo Filtro - fuente es un modelo simplificado que no considera los efectos producidos por el pulso glotal y la radiación debida a los labios, consideramos una etapa que compense tal efecto. Al introducir un proceso de filtrado al segmento analizado, mediante un proceso de preénfasis (filtro paso - altas), se resaltan las componentes de altas frecuencias ya que el efecto de radiación de los labios las atenúa [4].
Por otra parte, el modelo de predicción lineal hace una aproximación eficiente a las señales con componentes frecuenciales bajas, pero no así a las altas. Por lo que es necesario realzar las componentes frecuenciales altas si queremos considerar toda la información espectral del segmento en cuestión. En la ecuación (4.3) definimos el filtro de preénfasis, para un sistema FIR, con un cero en α.

$$H_{pre}(z) = 1 - \alpha z^{-1} \tag{4.3}$$

Donde α es un valor cercano a 0.9. El valor de α comúnmente usado es de 0.9375. [4]. La ecuación en diferencias del filtro de preénfasis es como sigue:

$$y(n) = x(n) - \alpha x(n-1) \tag{4.4}$$

Donde $y(n)$ es la salida y $x(n)$ es la entrada al filtro. En la figura (4.8) se presentan las respuestas en frecuencia del filtro de preénfasis al variar el valor de α (0.9, 0.9375, 0.95).

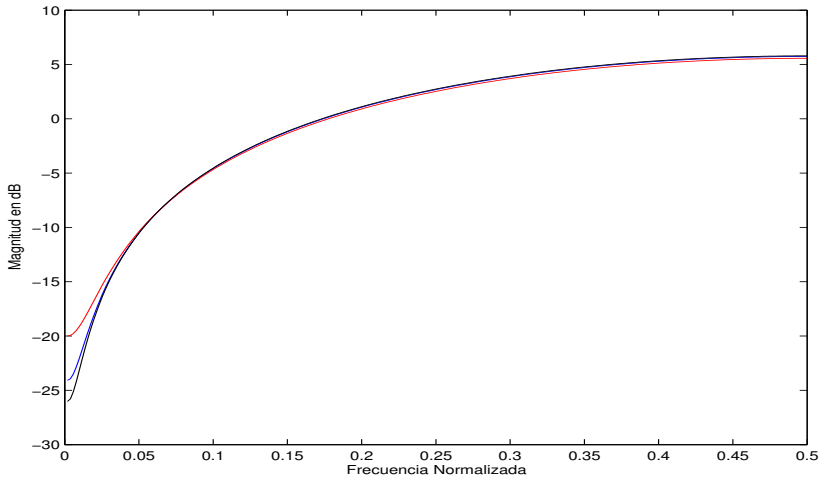

Figura 4.8: Respuesta en Frecuencia del Filtro de Preénfasis: $\alpha = 0.9$ (Rojo), 0.9375 (Azul) y 0.95 (Negro).

De la figura (4.8) se observa que, al variar los valores de α (cercanos a 0.9), la respuesta en frecuencia del filtro de preénfasis atenúa las componentes en bajas frecuencias y contribuye en aumentar la ganancia de las componentes en altas frecuencias. Además se visualiza que la elección del valor de α sólo variará la atenuación de las componentes de frecuencias bajas, siempre y cuando dicho valor se encuentre cercano a 0.9. Si elegimos a α=0.9375, la frecuencia de corte es de 600 Hz, es decir, atenuará las frecuencias por debajo de los 600 Hz. En la figura (4.9.a) mostramos un

segmento de voz real con $N = 256$ muestras, del fonema /o/. De la figura (4.9.b) observamos el resultado de aplicarle una ventana de Hamming al segmento de voz ya mencionado. Mientras que en la figura (4.9.c) se presenta la salida del filtro de preénfasis de la señal ventaneada.

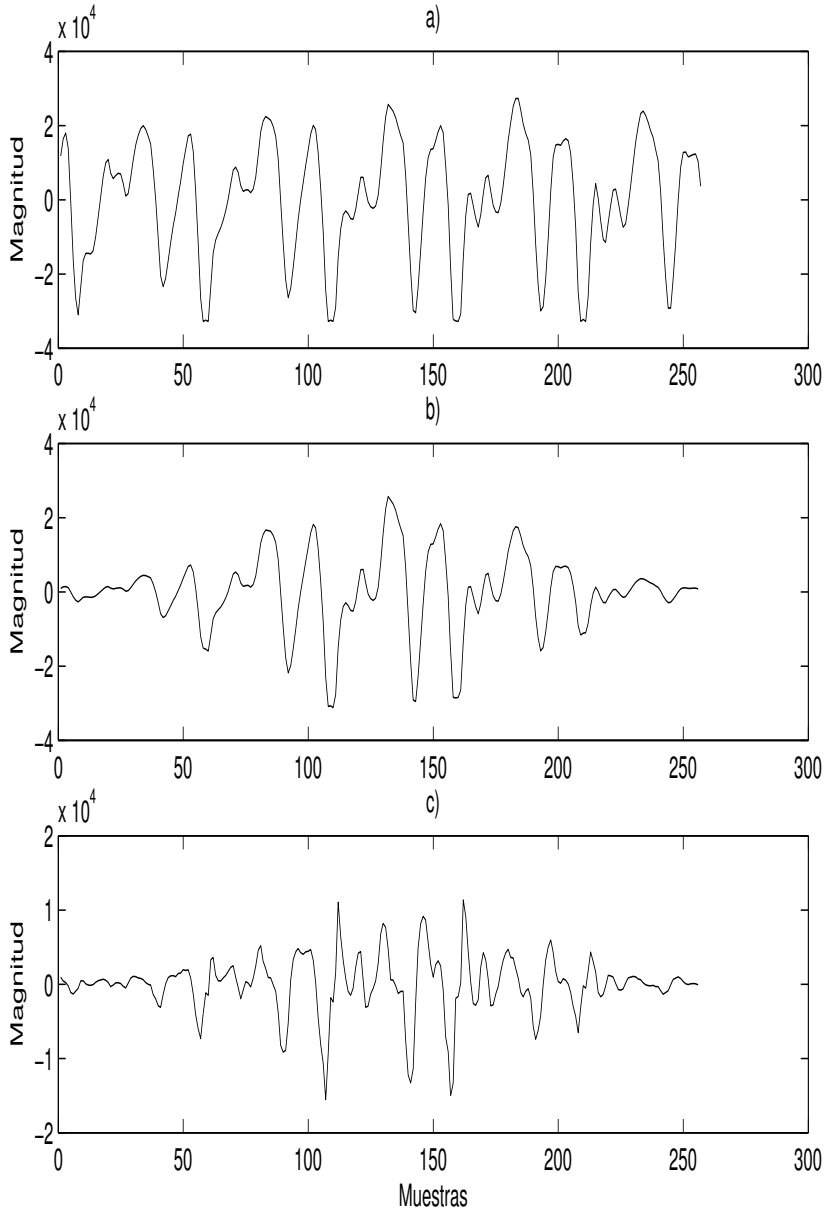

Figura 4.9: Segmento de Voz real, $N = 256$, fonema /o/. a) Segmento Original, b) Con ventana, c) Con ventana y filtrado de preénfasis.

4.1.4. Estimación de la Naturaleza del Segmento

Para la estimación del período de pitch es necesario determinar, previamente, la naturaleza del segmento bajo análisis. Para segmentos de sonidos voceados se tendrá que estimar el período de pitch, mientras que para segmentos con sonidos no voceados se omitirá tal cálculo. Existen diversas estrategias para la estimación del pitch, como son: Método del Recorte Central, Método de Gold - Rabiner, Método Homomórfico, Método AMDF (Average Magnitude Diference Function), Algoritmo SIFT (Simplified Inverse Filter Tracking), entre otras [4], [5], [8]. Por su mínima carga computacional y fácil implantación elegimos al Método del Recorte Central, debido a que explota las propiedades de la función de autocorrelación.

Método del Recorte Central

El método del Recorte Central ha sido una metodología muy usada para la estimación del período de pitch, como también de decisión de la naturaleza del segmento [4].
El proceso de recorte central consiste en recortar las amplitudes de una ventana en análisis entre dos niveles, es decir, sólo consideramos las amplitudes de la ventana a partir de los niveles de umbral propuestos y posteriormente calculamos su función de autocorrelación para determinar la periodicidad de la señal. En forma analítica podemos definir a la función del recorte central como se muestra en la ecuación (4.5) [4]:

$$\begin{array}{ll} y(n) = s(n) - C_L & \text{Si } \ s(n) \geq C_L \\ y(n) = s(n) + C_L & \text{Si } \ s(n) \leq -C_L \\ y(n) = 0 & \text{Otro caso} \end{array} \tag{4.5}$$

En la figura (4.10) se muestra la función del recorte central, así como en la figura (4.11) se observa un ejemplo del recorte de una ventana por medio de la función no lineal de la ecuación (4.5).

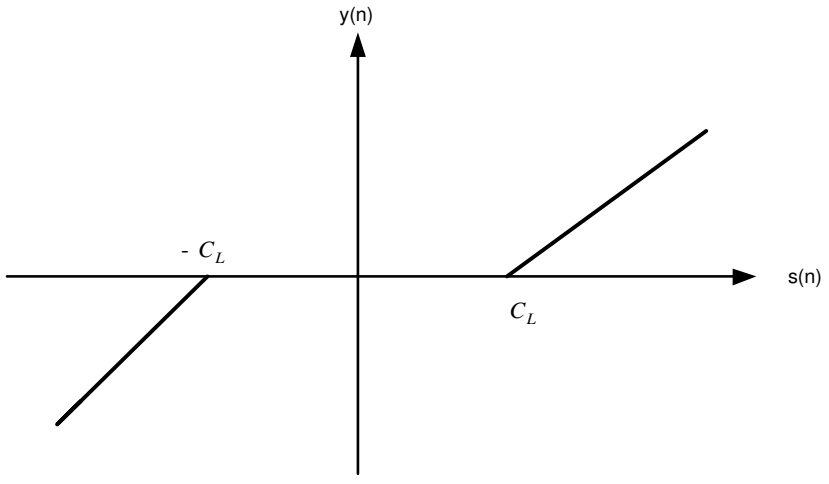

Figura 4.10: Función $y(n)$ del recorte central.

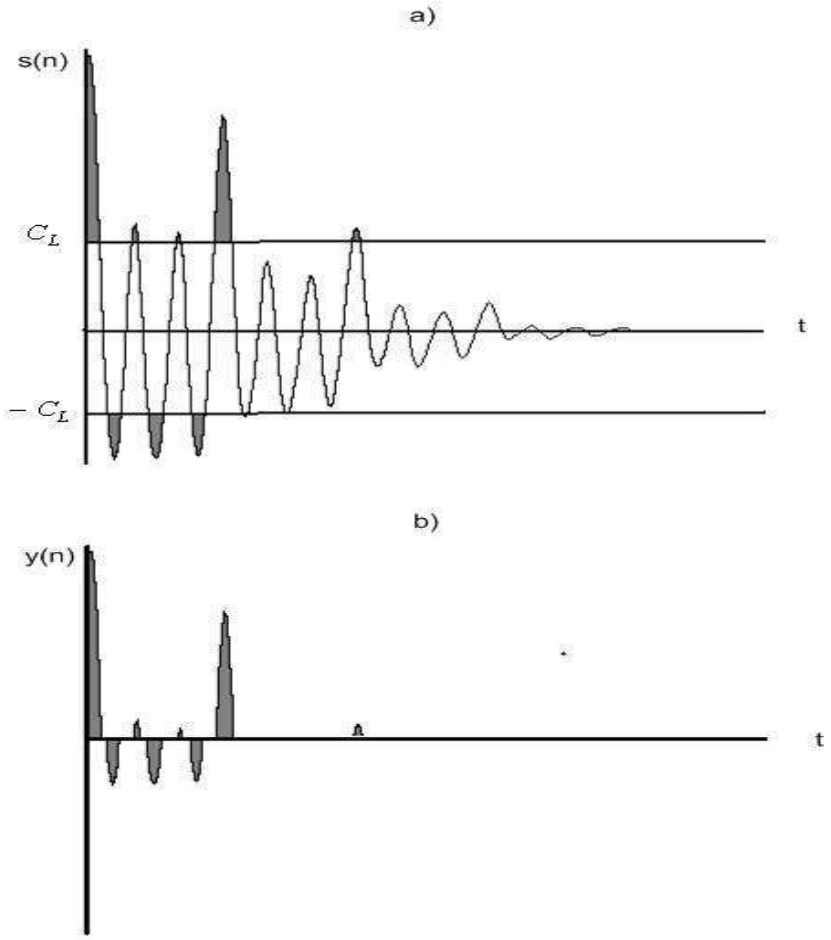

Figura 4.11: Ejemplo de recorte de un segmento. a) segmento original, b) segmento recortado.

Otra manera de definir a la función del recorte central, considerando una forma más eficiente de recortar a la señal por razones de formato numérico[1] es como sigue:

$$\begin{array}{lll} y(n) = & 1 & \text{Si } \ s(n) \geq C_L \\ y(n) = & -1 & \text{Si } \ s(n) \leq -C_L \\ y(n) = & 0 & \text{Otro caso} \end{array} \tag{4.6}$$

El efecto de aplicar un proceso de recorte es conservar sólo la información de la periodicidad de la señal. El nivel de umbral de C_L es calculado para cada ventana dependiendo del valor máximo de éste (A_{max}). Un valor típicamente empleado de C_L es [4]:

$$C_L = 0.3A_{max} \tag{4.7}$$

Sin embargo, al utilizar la ecuación (4.7) para la determinación de C_L puede conducir a errores notables ya que existen muchos cambios en las amplitudes del segmento, es decir, el recorte sólo se

[1]Evitando así desbordamientos al momento de realizar las multiplicaciones y acumulaciones en la función de autocorrelación.

concentra en la amplitud máxima, por lo que pueden existir pérdidas de información significativas en el proceso de recorte. Para evitar ésto, dividimos a la ventana en tres subsegmentos y empleamos la siguiente estrategia:

- Encontrar las amplitudes máximas del primer y tercer subsegmento (A_1 y A_3).
- Calcular el umbral C_L como se muestra en la ecuación (4.8):

$$C_L = K\ min(A_1, A_3) \tag{4.8}$$

  Donde el operador $min()$ calcula el valor mínimo entre A_1 y A_3 y K es un parámetro de calibración que se encuentra en el intervalo de 0.6 a 0.8 [4].

Entonces, ya determinado el umbral C_L y obtenida la señal recortada, aplicamos la función de autocorrelación a la señal recortada. La función de autocorrelación se define como:

$$R_c(l) = \sum_{n=0}^{N-1-l} s_c(n)s_c(n+l) \tag{4.9}$$

Donde $s_c(n)$ es el segmento de voz recortado, N es el número de muestras del segmento y l el índice temporal de retraso. Al aplicar la ecuación (4.9) a un segmento de voz real con N=256 muestras del fonema /o/ se obtiene la figura (4.12.c). En la figura (4.13) mostramos el resultado de aplicar el método de recorte central y la función de autocorrelación para distintos valores de K a la señal que se muestra en la figura (4.12.b). Por tanto, de la figura (4.13) se observa que la elección del valor de K sólo determinará el grado de recorte en la señal, pero conservando en cualquier caso la periodicidad en ésta.

En el caso de segmentos de sonidos voceados existe una periodicidad de la forma de onda en la señal, no así para ventanas no voceadas. Por tanto, la función de autocorrelación tiene la propiedad de ser periódica si la ventana en estudio lo es. Una ventaja de usar la función de autocorrelación es que presenta con mayor realce la propiedad de periodicidad de las señales. En consecuencia, la propiedad de periodicidad puede ser utilizada como criterio para la estimación de la naturaleza del segmento. Sin embargo, además de resaltar la periodicidad del segmento, también muestra información de la energía de la señal, $R_c(0)$. Esta información adicional puede propiciar decisiones erróneas [4]. Entonces, para la determinación de la naturaleza de la ventana se calcula el valor máximo de la función de autocorrelación comprendida entre las muestras 20 a la 200, y este valor máximo es comparado con un valor umbral igual a $0.3R_c(0)$. Si el valor máximo es mayor que el valor umbral se decide como un segmento voceado, si es menor, el segmento será no voceado.

Estimación del Pitch

Sí la ventana se decidió como voceada, entonces se estima el período de pitch, pero sí se decidió como ventana no voceada, se omite este cálculo. La estimación del pitch es por medio del cálculo del índice del valor máximo de la función de autocorrelación comprendido entre las muestras 20 a la 200, por lo que a este índice se le tendrá que sumar una constante igual a 20 para considerar el verdadero valor del período de pitch.

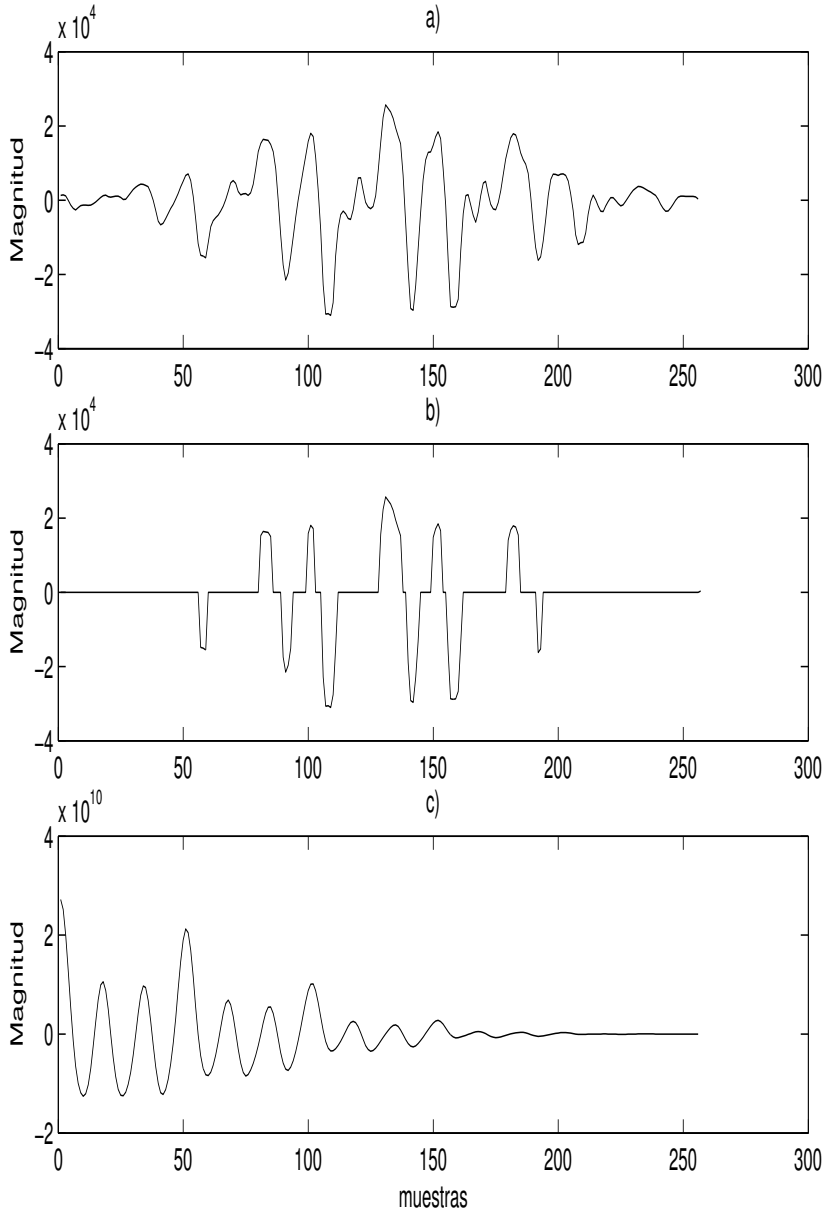

Figura 4.12: Ventana de Voz real, $N = 256$, fonema /o/. a) señal original enfatizada b) señal recortada, c) Aplicando la Función de Autocorrelación.

Metodología del Método del Recorte Central

Teniendo en cuenta los parámetros y características que integran al método del recorte central, mencionamos la metodología para la estimación de la naturaleza del segmento y la estimación del período de pitch:

- Se calcula el umbral C_L y se recorta la señal para aplicarle la función de autocorrelación
- El valor máximo de la función de autocorrelación es comparado con un valor umbral de

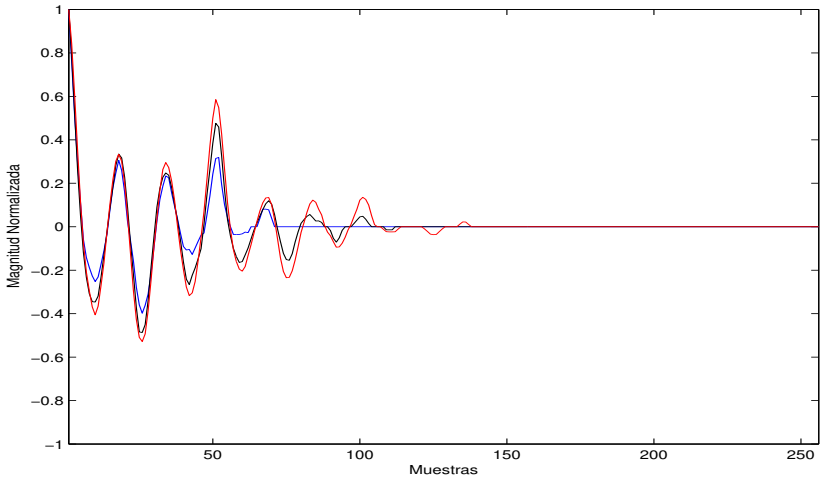

Figura 4.13: Función de Autocorrelación aplicada al Método del Recorte Central para los valores de $K = 0.8$ (Azul), 0.7 (Negro), 0.6 (Rojo).

$0.3R_c(0)$. Sí el valor máximo es mayor que el valor umbral, entonces se decide como un segmento voceado, si es menor, el segmento será no voceado.

- Sí el segmento se decidió como voceado, el período de pitch es igual al índice temporal (retraso) del valor máximo de la función de autocorrelación $R_c(l)$ entre las muestras 20 y 200, y sumándole una constante igual a 20.

4.1.5. Estimación de los parámetros LPC

Por medio del método de predicción lineal y resolviendo la ecuación normal que resulta de tal procedimiento[2] se obtiene los parámetros de la codificación por predicción lineal.
El orden de la predicción lineal está relacionado con el número de formantes que se quieran modelar, es decir, como los coeficientes de la predicción lineal definen al filtro Todo Polo del modelo filtro - fuente, los polos complejos determinarán los formantes del segmento en análisis, por ejemplo, si el orden de la predicción es de $p = 10$ se tomarán en cuenta los primeros 5 formantes que en teoría son suficientes para un modelo de voz [14].
La estimación de los parámetros LPC se logra empleando el algoritmo de Levinson - Durbin, con vector de entrada igual al vector de autocorrelación de la salida del filtrado de preénfasis. En la figura (4.14) se muestra la respuesta en frecuencia a diferentes valores de p para la estimación del segmento de voz real con $N = 256$ muestras del fonema /o/. En la figura (4.14) se observa que la elección del orden de la predicción lineal sólo consiste en la aproximación de las frecuencias formantes del segmento en análisis, pero al aumentar el orden de la predicción en términos prácticos únicamente incrementará la carga computacional, siendo $p = 10$ el orden más usado para la estimación de los parámetros LPC [14].
En la figura (4.15) se muestra la respuesta en frecuencia de los parámetros de la predicción lineal para un orden $p = 10$ y el espectro del segmento de voz real a $N = 256$ muestras del fonema /o/. En la figura (4.15) se visualiza que la respuesta en frecuencia de los parámetros estimados

[2]Ver capítulo tres.

aproximan a la envolvente del espectro del segmento real. Por lo que el espectro resultante de reconstruir la señal mediante los parámetros será más suave con respecto al original.

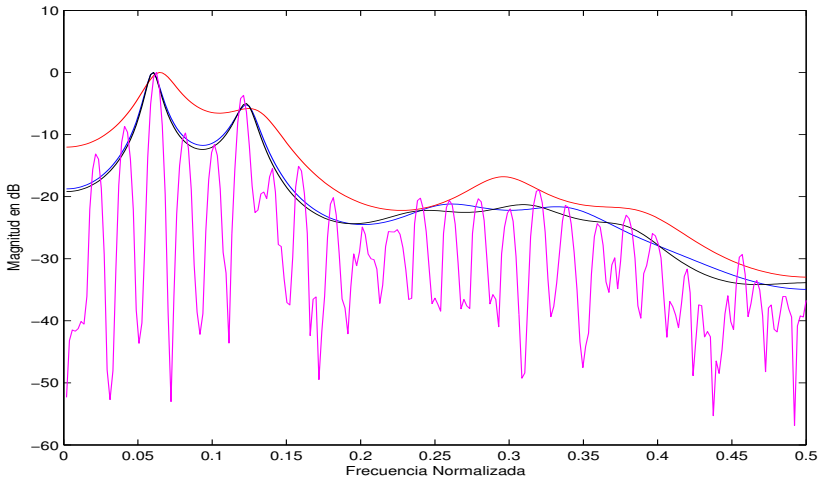

Figura 4.14: Respuesta en frecuencia a diferentes valores del orden de la predicción: $p = 10$ (Azul), 8 (rojo), 12 (negro). El espectro del segmento se muestra en color magenta.

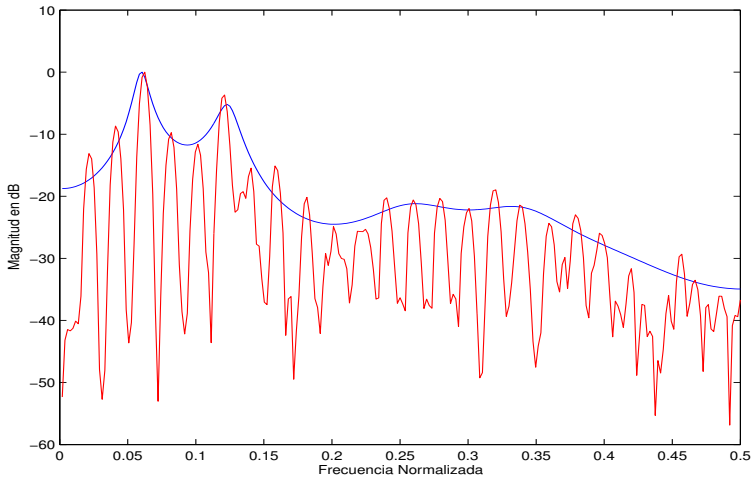

Figura 4.15: Respuesta en Frecuencia de los parámetros LPC con orden p=10 (Azul) y el espectro del segmento de voz (Rojo).

Cálculo de la Ganancia

El cálculo de la ganancia se realiza por medio de la ecuación (4.10):

$$G = \sqrt{R(0) - \sum_{i=1}^{p} a_i R(i)} \tag{4.10}$$

Donde $R(i)$ es el vector de autocorrelación, $\{a_i\}$ el conjunto de coeficientes LPC y p el orden de la predicción. Otra forma de estimar la ganancia del segmento es empleado el valor del error cuadrático que es calculado por el algoritmo de Levinson - Durbin[3].

En resumen, los parámetros estimados en los subprocesos que integran al proceso de análisis para realizar el proceso de síntesis son: la decisión de la existencia de un segmento de silencio, la decisión de la naturaleza del segmento (voceado o no voceado), el período del pitch, los coeficientes LPC y la ganancia.

4.2. Proceso de Síntesis

En el proceso de síntesis, con base a los parámetros estimados en el proceso de análisis, se reconstruye a la señal por medio de un filtro Todo Polo, siendo las entradas de excitación un tren de impulsos o ruido blanco, según la naturaleza de la ventana. Posteriormente se realiza un filtrado de deénfasis a la señal reconstruida. Sí en un segmento se decide como silencio, entonces se genera una ventana de ceros. El diagrama de flujo del procedimiento del bloque de síntesis se muestra en la figura (4.16).

4.2.1. Filtro Todo Polo

En el proceso de síntesis, se integra un filtro Todo Polo, que modela las frecuencias formantes del sistema de producción de la voz, dependiendo de los coeficientes LPC anteriormente estimados en el proceso de análisis. La señal de excitación a este filtro variará según sea la naturaleza del segmento analizado. Para el caso de un segmento voceado, la señal de entrada será un tren de impulsos con período igual al período de pitch, mientras que para segmentos no voceados, la señal de entrada será ruido blanco (figura 3.1). Para definir al filtro Todo Polo, sólo son necesarios los coeficientes $\{a_i\}$ y el valor de la ganancia para el cálculo de la respuesta de tal filtro. Tales coeficientes $\{a_i\}$ y la ganancia se estiman por medio del algoritmo de Levinson - Durbin.
Entonces, la función de transferencia del filtro de síntesis está definida por la ecuación (4.11):

$$H(z) = \frac{G}{1 + \sum_{i=1}^{p} a_i z^{-1}} \tag{4.11}$$

o bien, por su ecuación en diferencias:

$$\hat{y}(n) = Gx(n) - a_1\hat{y}(n-1) - a_2\hat{y}(n-2) - \ldots - a_p\hat{y}(n-p) \tag{4.12}$$

Donde G es la ganancia del segmento, $x(n)$ la entrada de excitación y $\hat{y}(n)$ es la voz sintética o estimada. En la figura (4.17) se muestra la reconstrucción del segmento de voz con N=256 muestras del fonema /o/ y el segmento de voz original.

De la figura (4.17) se observa que la señal reconstruida (ventana de voz de 256 muestras) no guarda gran parecido en el dominio temporal, sin embargo, al reconstruir una señal con mayor

[3]Ver capítulo tres.

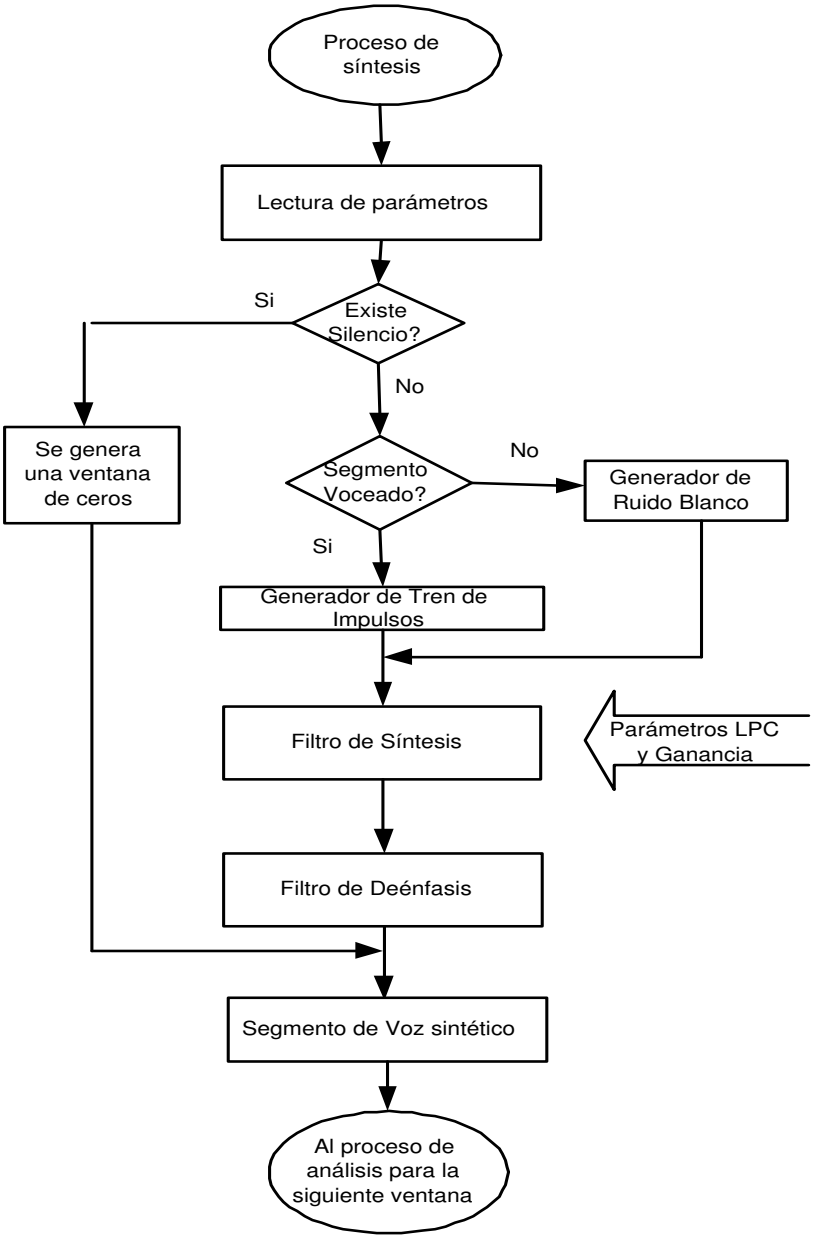

Figura 4.16: Diagrama de flujo del proceso de Síntesis.

número de ventanas sí se presenta similitud entre una señal original y una señal sintética (figura 4.18).

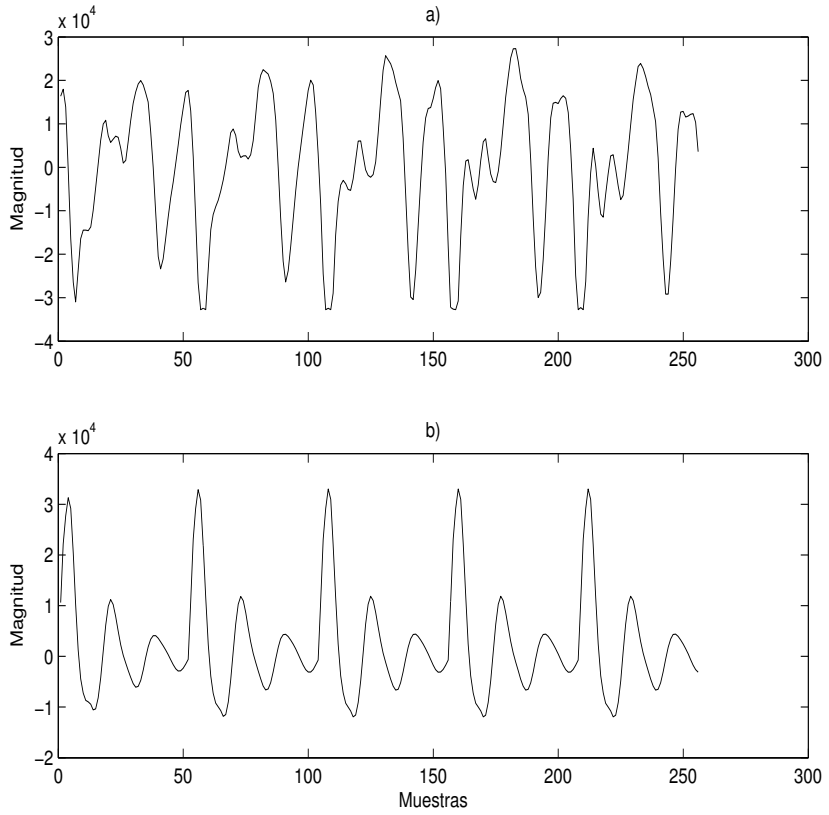

Figura 4.17: Reconstrucción del segmento de voz utilizando el filtro Todo Polo. a) Segmento sintético. b) Segmento Original.

Para el caso del dominio frecuencial se observa la gran similitud entre los espectros (figura 4.19). Por tanto, la información en la reconstrucción de la señal empleando el filtro de síntesis nos proporciona similitud en el dominio frecuencial, garantizando con esto la aproximación de las frecuencias formantes al momento de reconstruir la señal.

4.2.2. Filtro de Deénfasis

Para compensar los efectos de radiación debido a los labios, introducimos un filtro de deénfasis que resulta como la emulación del modelo de radiación de los mismos [5]. El filtro de deénfasis puede ser definido como un sistema de respuesta infinita al impulso (IIR), que se muestra en la ecuación (4.13):

$$H(z) = \frac{1}{1 - \alpha z^{-1}} \tag{4.13}$$

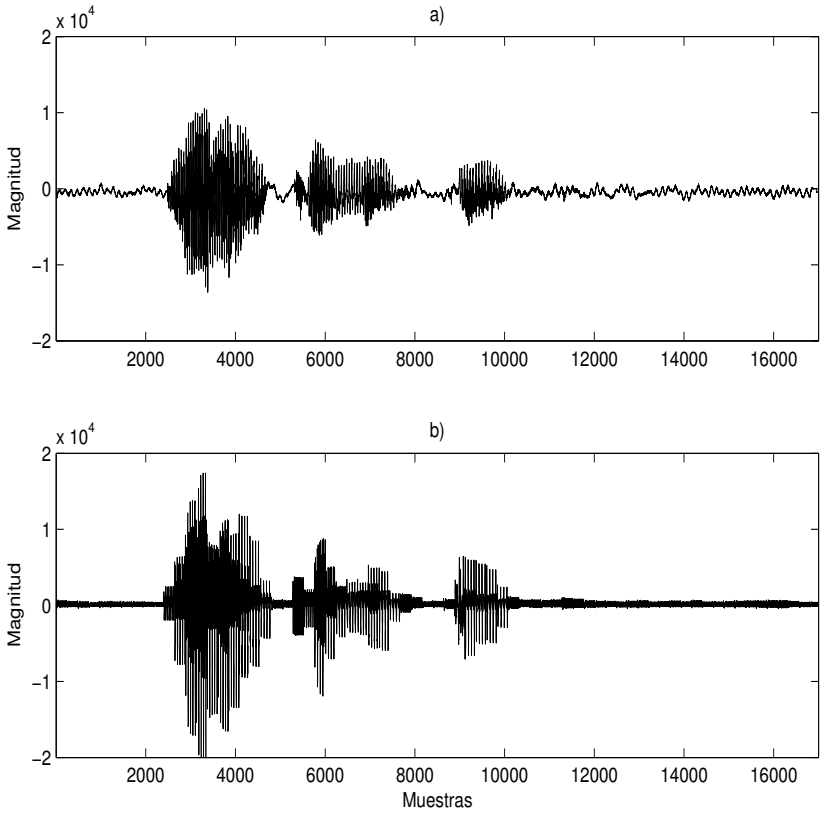

Figura 4.18: Ejemplo al reconstruir una señal con mayor número de ventanas a) señal original, b) señal reconstruida.

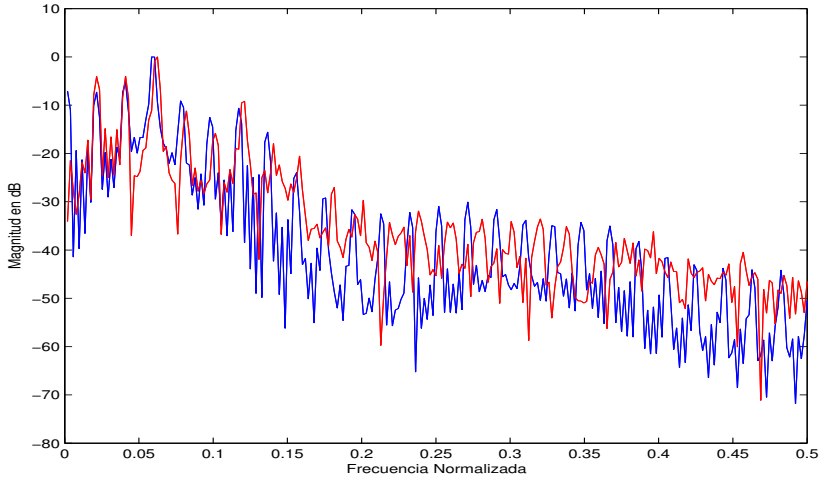

Figura 4.19: Espectro del segmento de voz utilizando el filtro Todo Polo. Segmento sintético (Azul) y Segmento Original (Rojo).

Donde α es el valor del polo en la función de transferencia. El filtro de deénfasis tiene un comportamiento paso - bajas, resaltando de esta forma las primeras frecuencias formantes. Se puede observar que el filtro de deénfasis es un filtro inverso del filtro de preénfasis (proceso de análisis) por lo que es usual elegir el mismo valor empleado por éste [5]. La ecuación en diferencias del filtro de deénfasis es como sigue:

$$y(n) = x(n) + \alpha y(n-1) \tag{4.14}$$

La figura (4.20) muestra la respuesta en frecuencia del filtro de deénfasis para diferentes valores de α. Sí aplicamos el filtro de deénfasis con valor de $\alpha = 0.9375$ al segmento reconstruido presentado en la figura (4.17) se obtiene lo mostrado en la figura (4.21). La frecuencia de corte para el valor de α=0.9375 es de 1600 Hz. De igual manera, se observa en la figura (4.21.b) que la forma de onda temporal de la señal filtrada no guarda gran similitud con la señal original, pero como en el caso de la figura (4.18) al emplear mayor número de ventanas, la señal original sí es parecida a la señal sintética. Para el caso del dominio frecuencial se observa la gran similitud entre los espectros (figura 4.22), donde el espectro de la señal sintética decae en las componentes de altas frecuencias, esto por emular los efectos de radiación de los labios.

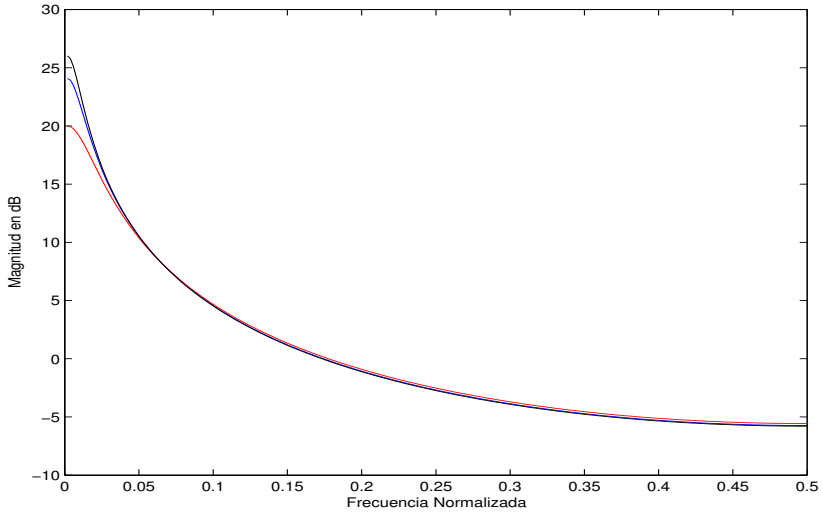

Figura 4.20: Respuesta en Frecuencia del Filtro de Deénfasis: $\alpha = 0.9$ (Rojo), 0.9375 (Azul) y 0.95 (Negro).

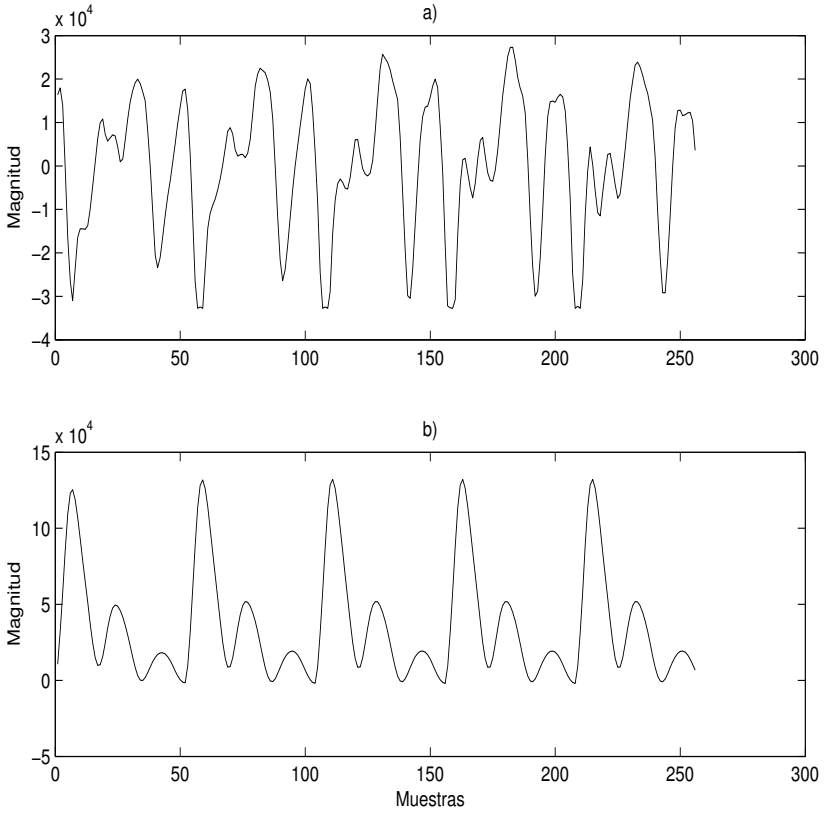

Figura 4.21: Salida del filtro de deénfasis. a) segmento original (figura 4.17). b) segmento filtrado.

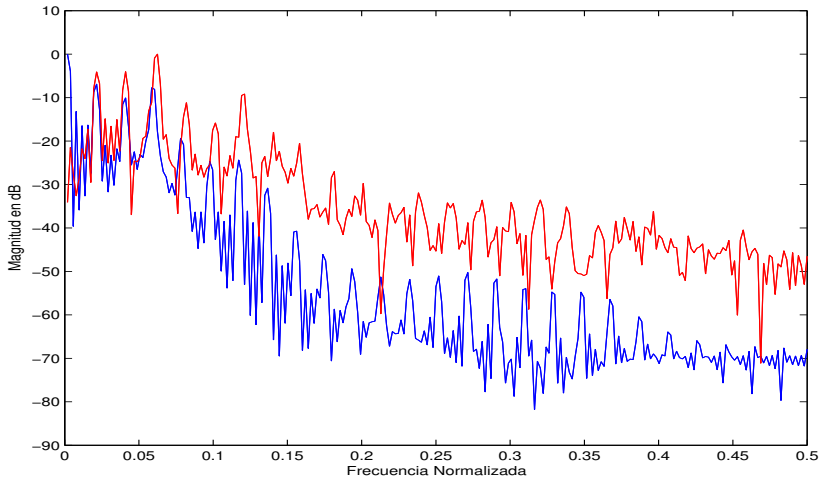

Figura 4.22: Espectro de la salida del filtro de deénfasis. a) segmento original (Rojo). b) segmento filtrado (Azul).

4.3. Resumen

En el presente capítulo se mostró una metodología, así como estrategias para diseñar un sistema de sintesis de voz, describiendo los algoritmos que integran a los procesos de análisis y síntesis. Por lo que podemos observar que la secuencia de pasos a realizarse en el sistema de síntesis es la siguiente:

- Proceso de Análisis:
  - Estimación de la energía de la señal para decidir si el segmento es de silencio o no.
  - Aplicar un proceso de ventaneo de la señal de voz original.
  - Decidir la naturaleza del segmento bajo estudio, mediante el método del recorte central.
  - Si el segmento bajo estudio es voceado, se estima el período de pitch con el método del recorte central.
  - Aplicar al segmento (ya sea voceado o no voceado) un proceso de filtrado de preénfasis.
  - Estimar los parámetros LPC y calcular la ganancia del segmento.
- Proceso de Síntesis:
  - Lectura de parámetros.
  - Si el segmento en el proceso de análisis se decidió como de silencio generar una ventana de ceros.
  - Aplicar un Filtro Todo Polo (Modelo filtro - fuente) definido por los parámetros LPC y la ganancia, a la entrada de excitación según sea la naturaleza del segmento (voceado - tren de impulsos, no voceado - ruido blanco).
  - Proceso de filtrado de deénfasis.

En el siguiente capítulo se describe la implantación del *Sistema de Síntesis en Tiempo Real empleando un DSP* (TMS320C5402) y basándonos en la metodología aquí presentada.

Capítulo 5

Implantación del Sistema de Síntesis de Voz en Tiempo Real

En este capítulo se desarrolla la implantación del diseño del sistema de síntesis de voz en tiempo real empleando el *DSP TMS320C5402 de Texas Instruments*. Tomando en cuenta la teoría descrita en el capítulo anterior, se implanta tal sistema en una arquitectura DSP, la cual maneja aritmética de punto fijo. La metodología empleada para nuestro diseño es con base a los diagramas de flujo mostrados en las figuras (4.2) y (4.16). Sin embargo, los algoritmos necesarios para los procesos de análisis y síntesis están sujetos a las características de la arquitectura empleada; como son: el tamaño de la palabra digital, el formato numérico, los tiempos de cálculo, la transferencia de datos, la disponibilidad de memoria, etc.

Los algoritmos que constituyen los procesos de análisis y síntesis de voz requieren de operaciones aritméticas (suma, multiplicación, división, etc.), operaciones lógicas (operadores booleanos, corrimientos, etc.) y operaciones básicas del procesamiento digital de señales (convolución, autocorrelación, etc); por lo que necesitamos de una arquitectura que cumpla con los requerimientos de implantación de éstos algoritmos en tiempo real.

El DSP TMS320C5402, al incluir una unidad aritmética - lógica de 32 bits, una unidad de multiplicación - acumulación de 17x17 bits, así como una unidad de corrimientos dedicada, entre otras características[1], resulta ser una arquitectura adecuada para implantar el sistema de síntesis de voz en tiempo real. Además, el ciclo de instrucción del TMS320C5402 dura 10 ns, por lo que este DSP es capaz de realizar 100 millones de instrucciones por segundo (MIPS) [15].

5.1. Esquema General del Sistema de Síntesis de Voz

Al implantar el sistema de síntesis de voz en el TMS320C5402, los procesos de análisis y síntesis se integran en un sistema multitareas, es decir, ya no consideramos a la señal de voz almacenada en un archivo o en memoria, por lo que se incluye una etapa de adquisición de la señal de voz por medio de un proceso de conversión analógico - digital. De igual manera, la voz sintética obtenida deberá ser presentada en el dominio analógico, esto, al emplear un convertidor digital - analógico. Así, en primera instancia, se observa que la adquisición y la entrega de voz sintética se realizan en conjunción con los procesos de análisis - síntesis. Lo anterior, se logra con el uso de

[1]Ver apéndice A.

las interrupciones definidas por la arquitectura del DSP. En la figura (5.1) se muestra el esquema general del sistema de síntesis de voz en tiempo real que emplea interrupciones asociadas a un convertidor A/D, D/A y puerto serie.

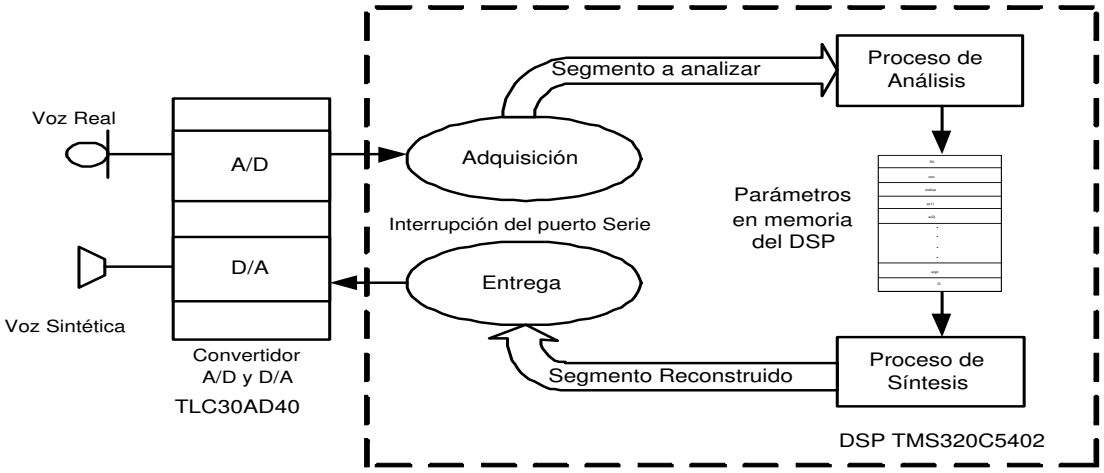

Figura 5.1: Esquema General del sistema de síntesis de voz en el DSP.

El esquema general del sistema de síntesis de voz en tiempo real presentado en la figura (5.1) está constituido por:

- **El proceso de adquisición de voz real y entrega de voz sintética (óvalos)**: producidos por una llamada de atención a interrupción de recepción del puerto serie multicanal bufereado (periférico interno del DSP) [16], que está conectado a un convertidor A/D y D/A (doble convertidor TLC30AD50, Codec) que cuenta con una interfaz para micrófono y otra para altavoz.
- **El proceso de análisis**: aquí se estiman los parámetros de voz del segmento adquirido. Este proceso es ejecutado por el DSP y consta de los siguientes bloques:
  - *Proceso de Ventaneo*: Se realiza el ventaneo, utilizando la ventana de Hamming, de la señal con un traslape del 30 % de la totalidad del segmento. La longitud de la ventana es de N=240 muestras, es decir, adquirimos un segmento en 30 ms a una frecuencia de muestreo de $f_s = 8$ kHz.
  - *Decisión de la existencia de un segmento de Silencio*: Se calcula la energía de la ventana y se compara con un umbral de silencio de forma experimental según las condiciones de ruido del entorno de adquisición.
  - *Decisión de la naturaleza de la ventana*: Mediante el método del recorte central se toma la decisión de la naturaleza de la ventana (segmento voceado o no voceado).
  - *Estimación del Pitch*: Si la ventana se decidió como voceada, se dispone a estimar el período de pitch.
  - *Filtrado de Preénfasis*: El segmento con ventana Hamming es filtrado para realzar las altas frecuencias.

- *Estimación de los coeficientes LPC*: Por medio del algoritmo de Levinson - Durbin se estima los parámetros LPC.
- *Cálculo de la Ganancia del segmento*: La ganancia es calculada por la recursión de Levinson - Durbin, siendo igual al error cuadrático evaluado en el orden p.

Los parámetros estimados se definen como un paquete de datos para su posible transmisión y/o almacenamiento mostrado en la figura (5.2) y el cuadro (5.1). El tamaño del paquete de datos que contiene a los parámetros estimados está en función del orden de la predicción a utilizar, es decir, para nuestro caso el paquete consta de 14 parámetros (p=10). El formato numérico empleado para los parámetros mostrados en la figura (5.2) y el cuadro (5.1) están en q_{15}, es decir, utilizamos 15 bits para la parte fraccionaria y un bit para el signo[2].

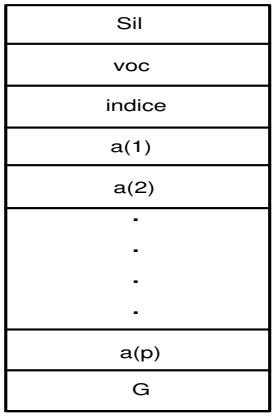

Figura 5.2: Paquete de datos de los parámetros estimados en una ventana.

Parámetros	Significado
Sil	Existe un segmento de silencio. Si sil=1, No sil=0.
Voc	Segmento Voceado. Si voc=1, No voc=0.
Índice	índice del período de pitch
a(1), a(2), ..., a(p)	Parámetros LPC.
G	Ganancia del segmento.

Cuadro 5.1: Definición de los parámetros estimados almacenados en el paquete.

- **El proceso de síntesis**: aquí se reconstruye la señal sintética por medio de los parámetros estimados y definidos por el paquete de datos que se muestran en la figura (5.2). El proceso de síntesis (ejecutado por el DSP) engloba los siguientes bloques:

  - *Filtro Todo Polo*: Según la decisión de la naturaleza del segmento (voceado o no voceado) se elige la excitación (tren de impulsos o ruido blanco) de un filtro Todo Polo definido por los parámetros LPC y la ganancia. El filtro Todo Polo reconstruye una señal sintética al modelar al sistema de producción de voz (tracto vocal).

[2]La longitud de la palabra digital (words) que se usa es de 16 bits.

- *Filtrado de Deénfasis*: la señal reconstruida por el filtro Todo Polo es filtrada para atenuar las altas frecuencias modelando la radiación producida por los labios.

La organización de los procesos a realizar por nuestro sistema de síntesis de voz es como una estructura "pipeline", es decir, las tareas de adquisición, proceso de análisis - sintesis y de entrega de la voz sintética se ejecutan contemplando que mientras se adquiere un segmento, se procesa el anterior y, posteriormente se entrega la voz reconstruida y así sucesivamente.
En la figura (5.3) se muestra la organización de las tareas a ejecutarse.

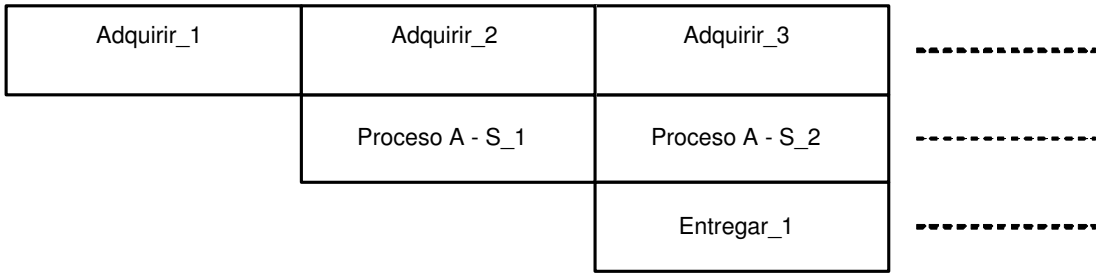

Figura 5.3: Organización de las tareas a ejecutarse.

El esquema de tiempos de la ejecución de las tareas (figura 5.4) presenta una señal de voz real de la palabra rojo (voz masculina) muestreada a una frecuencia de f_s= 8 kHz que a continuación se describe:

- Se observa que la primera tarea a ejecutarse es adquirir un segmento de tamaño de 240 muestras. El segmento 1 se almacena en un arreglo en memoria DARAM (RAM de doble acceso) del DSP.
- Al llenarse este arreglo, mediante una bandera de sincronía, se disponen a ejecutarse los procesos de análisis - síntesis para el segmento 1, estos procesos no deberán de sobrepasar los 30 ms (tiempo de la adquisición para cada ventana) en sus respectivos cálculos. Sin embargo, por medio de la interrupción de recepción del puerto serie del DSP, la adquisición del siguiente segmento (segmento 2) se realiza en conjunción con los procesos de análisis - síntesis para el segmento 1.
- De igual manera, al adquirir el segmento 3, se levanta la bandera de sincronía, ejecutándose los procesos de análisis - síntesis para el segmento 2, mientras se entrega el segmento reconstruido del segmento 1 (por medio de la interrupción de recepción del puerto serie).

Las tareas de adquisición, procesos de análisis - sintesis y entrega de voz sintética se ejecutan respectivamente de manera indefinida por ser un sistema en tiempo real.

En la figura (5.5) se muestra el diagrama de flujo de las tareas de adquisición, entrega y los procesos de análisis - síntesis, para su implantación en el DSP C5402. Los procedimientos presentados en el diagrama de flujo de la figura (5.5) se describen a continuación:

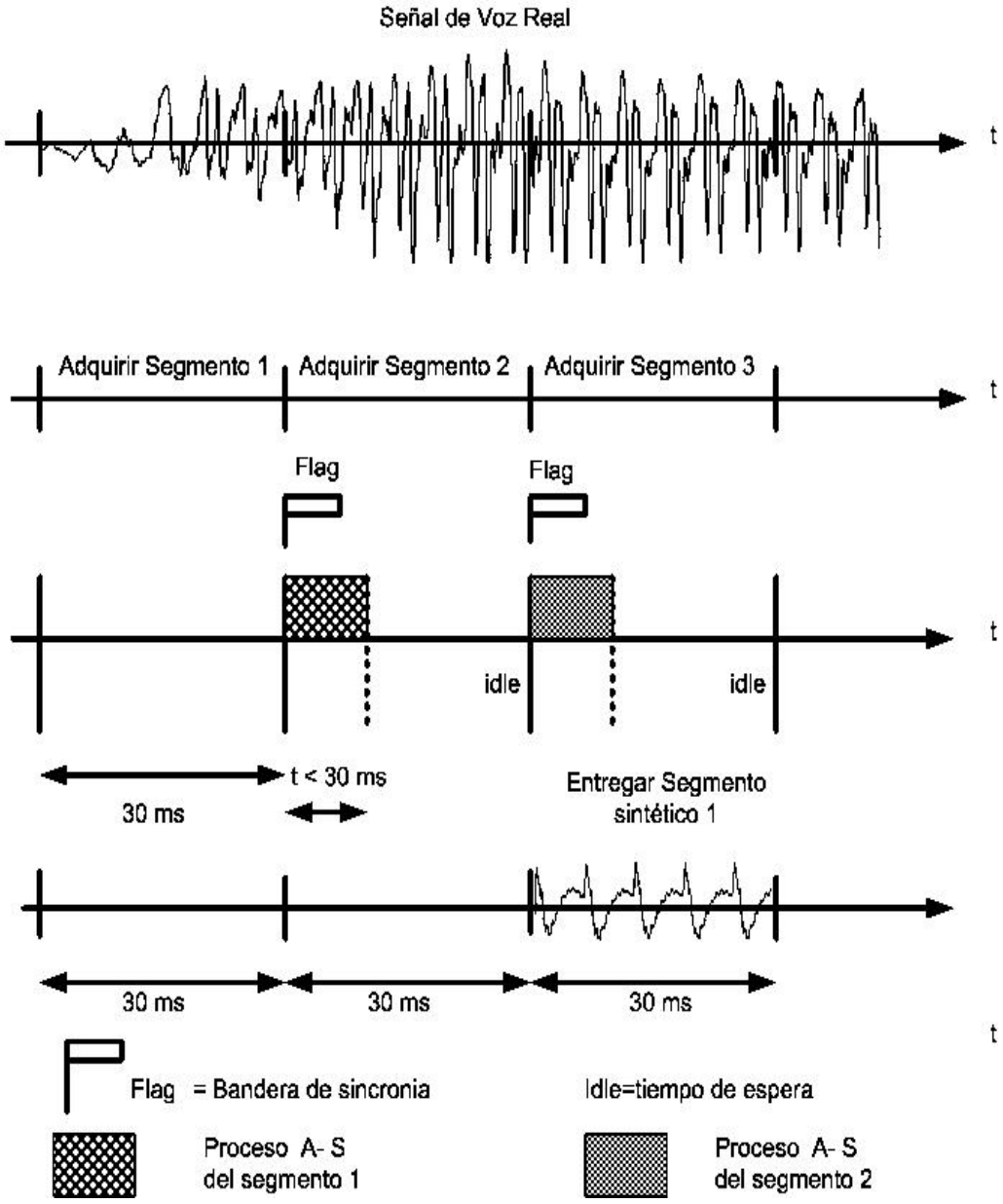

Figura 5.4: Esquema de tiempos de la ejecución de las tareas.

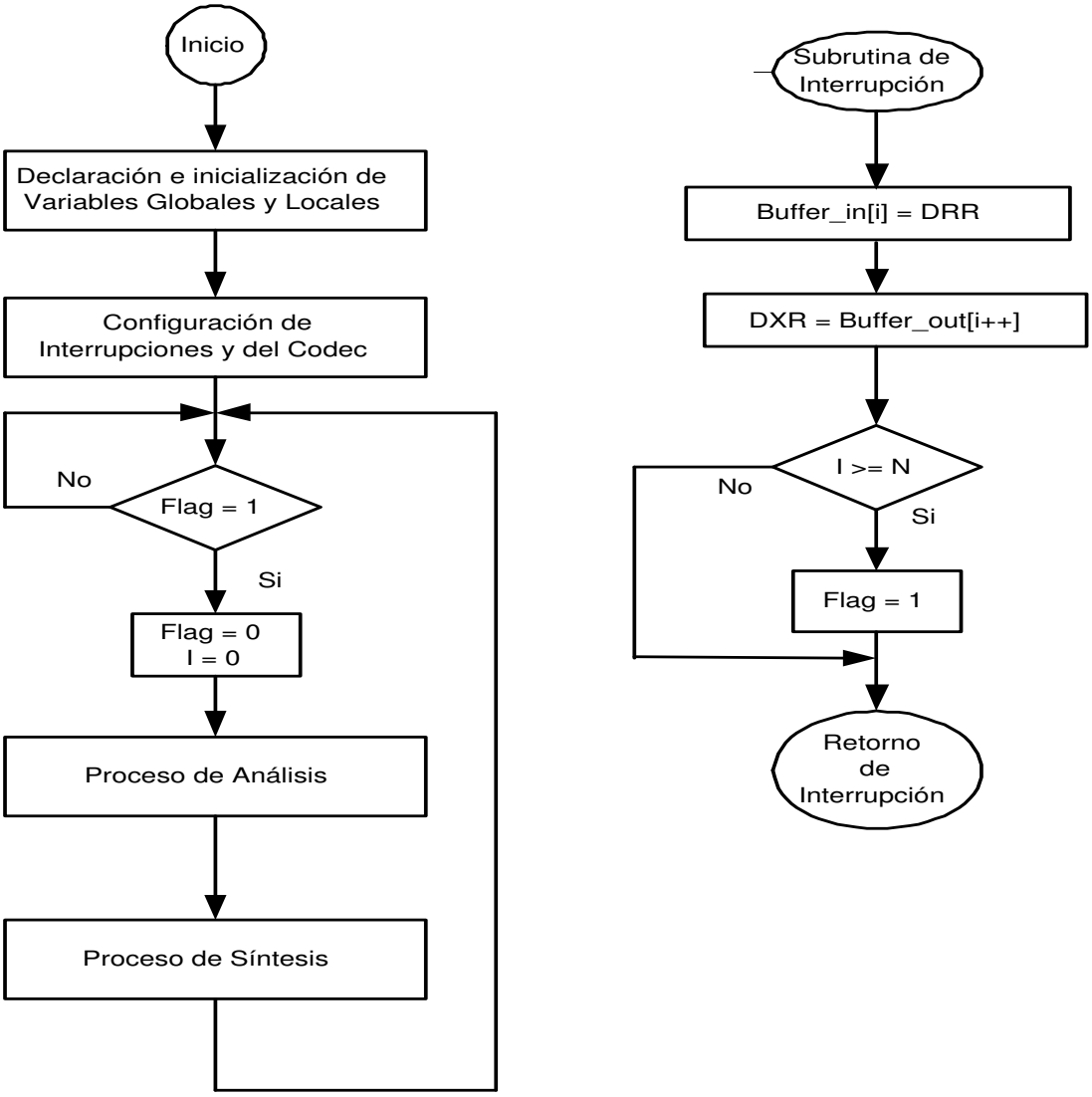

Figura 5.5: Diagrama de Flujo de las tareas de adquisición, entrega y los procesos de análisis - síntesis.

- ***Definición e inicialización de variables***: Se definen e inicializan las variables y arreglos necesarios para la ejecución de las tareas de adquisición de voz real, entrega de la señal reconstruida y los procesos de análisis y síntesis.

- ***Configuración e inicialización del Codec***: Es necesario la inicialización del Convertidor A/D y D/A y su configuración que consiste en abrir el puerto del codec conectado al puerto serie y definir la frecuencia de muestreo, así como las ganancias de entrada y salida de éste.

- ***Configuración de Interrupciones***: Se habilitan las interrupciones a emplear; para nuestro diseño, se utiliza la interrupción de recepción por hardware del puerto serie.

- ***Bandera de sincronía***: Esta variable (flag) funge como bandera para determinar si el arreglo está lleno (adquisición) dependiendo de la longitud del segmento que sea empleada. Para nuestro diseño la longitud de los segmentos a analizar es de $N = 240$ muestras.

- ***Proceso de Análisis***: Se ejecutan los bloques correspondientes al proceso de análisis:

  - Proceso de Ventaneo.
  - Decisión de la existencia de un segmento de Silencio.
  - Decisión de la naturaleza de la ventana (voceado - no voceado.)
  - Estimación del Pitch.
  - Filtrado de Preénfasis.
  - Estimación de los coeficientes LPC.
  - Cálculo de la ganancia.

- ***Proceso de Síntesis***: Se ejecutan los bloques correspondientes al proceso de síntesis:

  - Filtro Todo Polo.
  - Filtrado de Deénfasis.

- ***Llamada de atención a Interrupción***: Mediante una subrutina de interrupción se adquiere la señal real, donde los valores adquiridos se almacenan en un registro de recepción del puerto serie (DRR), así como se entrega la señal sintética almacenando estos valores en el registro de transmisión del puerto serie (DXR). Estos registros pasan sus respectivos valores a las etapas de conversión A/D y D/A. Por medio de la bandera de sincronía (flag) se establece si el arreglo utilizado para almacenar al segmento adquirido está lleno.

5.2. Consideraciones del Hardware para el Sistema de Síntesis de Voz

Los elementos principales que constituyen el hardware necesario para la implantación del diseño del sistema de síntesis de voz son:

- Un DSP TMS320C5402 que constituye la unidad de ejecución principal para los procesos de análisis - síntesis requeridos, así como las interrupciones por hardware necesarias para la adquisición y entrega de voz. Los bloques principales que integran al DSP (memoria, unidad aritmética - lógica, unidad de multiplicación - acumulación, bloque de corrimientos, periféricos internos etc.), se explican en detalle en el apéndice A.

- Un doble convertidor TLC30AD50 de 16 bits, conectado a un puerto serie bufereado multicanal que integra el DSP, donde la señal de voz real se muestrea por el convertidor A/D a una $f_s = 8$ kHz, y a la misma frecuencia es entregada la voz sintética por el convertidor D/A. Cabe mencionar que el doble convertidor TLC30AD50 puede configurarse para distintas tasas de muestreo.

El diagrama de bloques de la configuración del Hardware necesario para nuestro diseño se presenta en la figura (5.6).

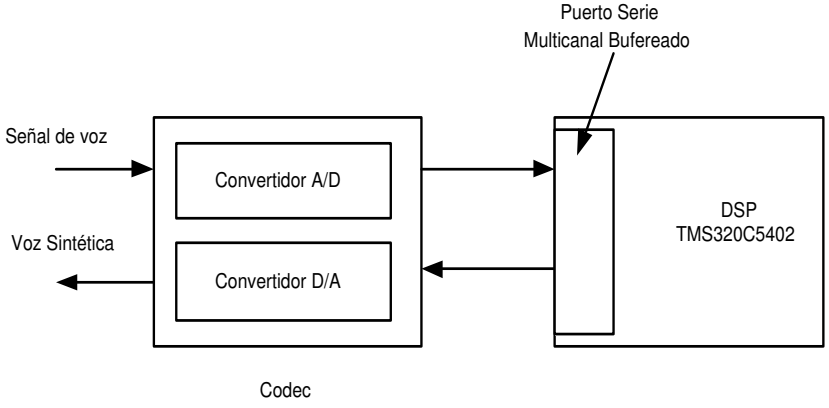

Figura 5.6: Diagrama de bloques de la configuración del Hardware empleado.

Para el diseño del sistema de síntesis de voz se dispone de una tarjeta de desarrollo y evaluación (DSK) que incluye una interface entre una terminal huésped (PC) y el C5402, de tal manera que mediante un ambiente de desarrollo (Code Composer Studio) se pueden realizar desarrollos, pruebas y emulaciones en aplicaciones en tiempo real. La tarjeta de desarrollo DSK para el DSP TMS320C5402 consta de los siguientes componentes en Hardware necesarios para nuestro diseño:

- *DSP TMS320VC5402 a 100 MHz.*

- *Un doble convertidor A/D TLC30AD50 para interface con micrófono y altavoz.*

- *Un controlador de prueba de bus JTAG TBC SN74ACT8990 para emulación e interface con el puerto huésped conectado a la PC por medio de un puerto paralelo.*

Además de incluir los siguientes componentes:

- Interface telefónica (DAA).

- Interface de datos asíncronos RS-232.
- SRAM Externa (64k x 16 bits).
- Memoria Flash Externa (256k x 16 bits).
- Interface de expansión para tarjetas (daughterboard).

5.3. Consideraciones del Software para el Sistema de Síntesis de Voz

El *Starter Kit* del DSP TMS320C5402 de Texas Instruments cuenta con el ambiente de desarrollo integral *Code Composer Studio*[3] (CCS) para su programación. El CCS contiene un compilador de lenguaje C, que traslada dicho código al lenguaje ensamblador nativo del C5402.
Para configurar el DSP y el Codec en uso se debe crear un proyecto en el cual se incluirán las bibliotecas y funciones para la generación del código máquina con el cual operará el DSP. Mediante el ambiente Code Composer podemos albergar nuestro código en memoria DARAM del DSP.

Los pasos generales para la realización de un sistema en tiempo real empleando el DSP TMS320C5402 y el ambiente CCS se muestran a continuación:

- Creación de un proyecto, en el que se incluyen las librerías rts.lib, dsk5402.lib y drv5402.lib. Estas librerías están asociadas a la definición de punto de entrada, configuración del tamaño de pila y las subrutinas para operaciones en aritmética de punto flotante, como también de las variables y punteros destinadas a la conversión del lenguaje C a lenguaje ensamblador, respectivamente.
- Incluir las librerías asociadas a la inicialización del codec.
- Declaración de variables globales e inicialización de variables.
- Definición de las subrutinas de atención de llamada a interrupción.
- Subrutina principal (kernel).
- Se incluye un archivo de comandos de ligado el cual define el mapa de memoria a usar para memoria dato y programa.
- Se incluyen las definiciones de la subrutinas de los vectores de interrupción.

5.4. Evaluación de los requerimientos para el Sistema de Síntesis de Voz

Con las consideraciones anteriormente presentadas mostramos la cantidad de memoria requerida para el diseño realizado del sistema de síntesis de voz en tiempo real (cuadro 5.2), esto, con

[3]Ver apéndice B.

base al mapa de memoria del C5402[4]. Además, en el cuadro (5.2) se incluye una columna que muestra el porcentaje de memoria requerida con respecto a la memoria total que integra el DSP (16k x 16 bits localidades en DARAM).

Bloque	Memoria dato (words)	memoria programa (words)	Porcentaje (%)
rts.lib	34	2007	12.46
dsk5402.lib	60	5660	34.91
drv5402.lib	30	75	0.64
pila	1024	-	6.25
Constantes	488	-	2.98
Código	1323	1553	17.55
Total	2959	9295	74.79

Cuadro 5.2: Memoria dato y programa requerida en el diseño.

El bloque de constantes se refiere a los arreglos asignados a la ventana de Hamming, al ruido blanco, y valores constantes necesarios en los proceso de análisis y síntesis. En cuanto al bloque de código (memoria programa), éste alberga las funciones y subrutinas de los procesos de análisis, síntesis y subrutina de la llamada a atención de interrupción, mientras que en la memoria dato requerida en el bloque de código se definen las variables y arreglos temporales necesarios. En el cuadro (5.3) se presentan los requerimientos de memoria programa, desglosados, para las funciones y subrutinas que integran al código del diseño.

Subproceso	Memoria programa (words)
Proceso de ventaneo y traslape	140
Proceso de umbral de silencio	82
Decisión de la naturaleza y estimación del pitch	248
Filtro de preénfasis	56
Recursión de Levinson - Durbin	252
Filtro Todo Polo	184
Generador de tren de impulsos o lectura del ruido	95
Filtro de deénfasis	56
Subrutina de interrupción	57
Función de autocorrelación	79
Total	1249

Cuadro 5.3: Memoria programa requerida para cada proceso.

En el siguiente capítulo se hace una evaluación y análisis de tiempos de cálculo para cada uno de los bloques que integran a los procesos de análisis y síntesis. Como el tiempo de adquisición para 240 muestras es de 30 ms a una frecuencia de muestreo de f_s= 8 kHz, para operar en tiempo real es necesario que el tiempo total con que se ejecutan los procesos de análisis - síntesis no sobrepasen tal limite de tiempo (30 ms).

[4]Ver apéndice A.

5.5. Formatos Numéricos empleados en el Diseño del Sistema de Síntesis

Cuando un sistema de procesamiento digital de señales es implantado en una arquitectura dedicada de punto fijo (DSP TMS320C5402), debe existir un compromiso entre el rango dinámico de las variables y la precisión de las mismas. Esto también es válido en procesadores de punto flotante, pero es específicamente cierto para procesadores de punto fijo [17].
Al efectuar operaciones aritméticas en un sistema de procesamiento digital de señales representado en formatos numéricos de precisión finita se tienen tres tipos de errores:

- Efecto de conversión de una señal analógica a digital.
- La representación de los coeficientes.
- El truncamiento de los resultados cuando se efectúan sumas o productos.

En cualquier sistema digital, los números son representados como una combinación finita de números binarios, o bits que toman valores "0" y "1", lo que ocasiona errores por los efectos de precisión finita debido a los efectos de cuantización. Los bits son organizados en conjuntos de ocho, llamados bytes, o 16 bits, llamados words. La representación más frecuente de los números en un sistema digital es a través de dos tipos de notación o formatos: de punto fijo y de punto flotante.

Formato Numérico de Punto Fijo

La representación numérica en formato digital de punto fijo es similar a la representación de números decimales, es decir como un conjunto de números con un punto decimal. En este formato la cantidad total de bits L de la palabra digital se particiona en: los bits para la parte entera Q_E, los bits para la parte fraccionaria Q_F y un bit de signo, es decir que L = 1 + Q_E + Q_F como se muestra en la figura (5.7), donde entre la parte entera y la parte fraccionaria existe una punto hipotético o punto entero, que sólo lo interpreta el programador [17].

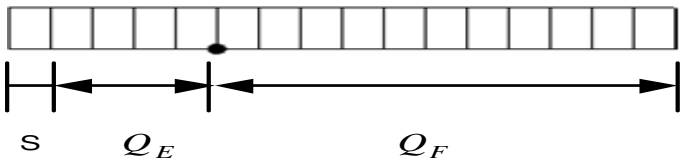

Figura 5.7: Representación del formato en punto fijo.

Comúnmente en la utilización del formato en punto fijo para su representación se emplea la nomenclatura q_i (Q_F), que se refiere a la cantidad de bits para la parte fraccionaria, donde i significa el número de bits para la parte fraccionaria.

Los valores adquiridos por el codec se almacenan en un formato q_{15}, es decir, consideramos a los valores en el intervalo de -1 a 1, es decir, asignando 15 bits para la parte fraccionaria y un bit para el signo. Por ejemplo, si al muestrear, cuantizar y codificar obtenemos el valor de 15336 en sistema decimal o 0011101111101000 en representación binaria, al considerar el formato q_{15} dicho

valor lo supondremos como 0.468017578125. Nuestro diseño de síntesis de voz está integrado por código en aritmética de punto fijo y flotante.
En las subrutinas en aritmética de punto fijo, al realizar las operaciones de multiplicaciones y sumas, es obligatorio realizar las conversiones necesarias entre formatos:

- Para operaciones de suma y resta, los operandos deberán encontrarse en el mismo formato. En el caso que los operandos estén definidos en diferentes formatos, mediante operaciones de corrimiento se puede recuperar el formato establecido.
- Para operaciones de multiplicación, al considerar los operandos (16 bits) en formato q_{15} la palabra digital resultante estará representada en 32 bits y en formato de q_{30}, por lo que realizamos corrimientos para recuperar el formato q_{15} y tomamos los 16 bits más representativos.

Entonces, es obvio que tendremos pérdidas en precisión numérica, lo cual es inherente al implantar cualquier algoritmo en una arquitectura de aritmética de punto fijo y es donde el diseñador del sistema debe de ser muy cuidadoso en la implantación de los algoritmos. El siguiente código escrito en lenguaje C ejemplifica el uso de operaciones de suma y multiplicación en aritmética de punto fijo:

```
// Ejemplo de operaciones de multiplicación y suma en
// aritmética de punto fijo en lenguaje C
// variables: x[N], h[N], vectores de entrada con longitud N
// y[N], vector de salida con longitud N.
// Los vectores se suponen en formato de q15

  for (i=0; i<N; i++)
    y[i]=(x[i]*h[i])>>15; \\se realizan 15 corrimientos a la
                          \\derecha para recuperar el formato
                          \\en q15
```

Formato Numérico de punto flotante

Para la representación numérica en formato digital de punto flotante [13] es necesario definir los campos de bits destinados a la parte de la mantisa (M) y el exponente (E). El compilador de C del C5402 utiliza el estándar IEEE 754 (Instituto de Ingenieros Eléctricos y Electrónicos) que representa un número en punto flotante como $X = (-1)^s \cdot 2^{E-127}(M)$, que se muestra en la figura (5.8).

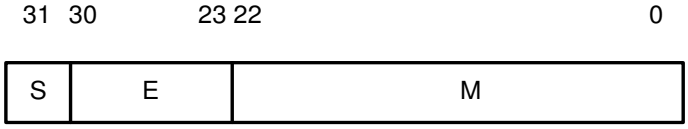

Figura 5.8: Representación del formato en punto flotante (IEEE 754).

Este número tiene las siguientes interpretaciones:

- Si E=255 y M≠0, entonces X no es un número.
- Si E=255 y M=0, entonces X=$(-1)^s \cdot \infty$.
- Si $0 < E < 255$ entonces X=$(-1)^s \cdot 2^{E-127}(1.M)$.
- Si E=0 y M≠0, entonces X=$(-1)^s \cdot 2^{E-126}(0.M)$.
- Si E=0 y M=0, entonces X=$(1)^s \cdot 0$.

Donde 0.M es una fracción y 1.M es un número mixto con un bit entero y 23 fracciones para la mantisa.
Por ejemplo, el número que se muestra en la figura (5.9) tiene el valor X=$(-1)^0$x$2^{130-127}$x1,1010...0, resultando X=2^3x$\frac{13}{8}$ = 13.

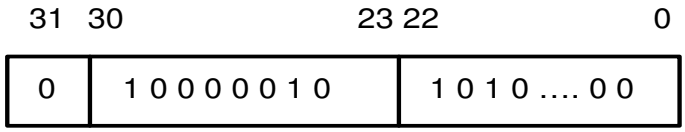

Figura 5.9: Representación del número 13 en punto flotante (IEEE 754).

El intervalo de la magnitud de los números en punto flotante de 32 bits IEEE 754 [13] va desde 2^{-126}x2^{-23} hasta (2-2^{-23})x2^{127}, es decir, desde 1.18x10^{-38} hasta 3.40x10^{38}. El siguiente código escrito en lenguaje C ejemplifica el uso de operaciones de suma y multiplicación en aritmética de punto flotante:

```
// Ejemplo de operaciones de multiplicación y suma en
// aritmética de punto flotante en lenguaje C
// variables: x[N], h[N], vectores de entrada con longitud N
// y[N], vector de salida con longitud N.
// Los vectores se suponen en formato de punto flotante

  for (i=0; i<N; i++)
    y[i]=x[i]*h[i];
```

Del anterior código se observa que la implantación de un algoritmo en aritmética de punto flotante usando un compilador de C para el C5402 no difiere de otras implantaciones de un sistema digital programados en lenguaje C, en este caso el programador se despreocupa del corrimiento del punto. Cabe mencionar que al emplear aritmética de punto flotante en el compilador de C del C5402 se debe incluir en el proyecto la librería rts.lib, la cual define las subrutinas en ensamblador de la conversión entre formato de punto fijo a formato de punto flotante.

5.6. Resumen

A lo largo del presente capítulo se describió el diseño del sistema de síntesis de voz considerando su implantación en tiempo real y empleando una arquitectura dedicada de aritmética de punto fijo (C5402). Para esta implantación, fue necesario establecer una serie de estrategias para la adaptación de los algoritmos usando un formato numérico de punto fijo. Además, al implantar un sistema de síntesis de voz en tiempo real haciendo uso de interrupciones por hardware que integran al DSP, se pueden realizar procesos multitareas como los procesos de adquisición, procesos de análisis - síntesis y de entrega de la voz sintética de manera conjunta.

Al revisar el diseño del sistema de voz en tiempo real, los factores determinantes en su implantación son los tiempos de ejecución de las instrucciones, la cantidad de memoria dato y de memoria programa requeridas y disponibles, y la inclusión de las librerías propias del DSP para la configuración e inicialización del codec.

Esto nos señala que, no sólo requerimos de un buen manejo de los lenguajes de programación en los que se implantó el sistema, sino que es imperativo conocer los algoritmos que se encuentran en todos los procesos y la arquitectura que se proponga. Al proponer una arquitectura DSP nos lleva a conocer su estructura, los mapas de memoria, el modo de programación, etc., además de las herramientas con las que sea necesario diseñar e implantar cualquier sistema que incluya el procesamiento digital de señales en tiempo real. Las nuevas tendencias de programación de dispositivos tales como el DSP TMS320C5402 se orientan al manejo de lenguajes de alto o medio nivel que facilitan la configuración de los periféricos y del mismo DSP, lo cual sería complicado hacer en lenguaje ensamblador.

En el siguiente capítulo se muestran los resultados obtenidos en la implantación del diseño empleando el DSP (aritmética de punto fijo). Además de incluir los tiempos de cálculo de los subprocesos que integran a los procesos de análisis y síntesis.

Capítulo 6

Resultados

En este capítulo se muestran los resultados obtenidos en el trabajo de tesis. Se presentan simulaciones en aritmética de punto flotante y fijo del diseño de sistema de síntesis de voz empleando el entorno de programación de Matlab y el CCS. Además se realiza una simulación empleando el DSP en tiempo congelado de la aplicación del sistema de síntesis de voz. Esta simulación en tiempo congelado (señal de voz almacenada en memoria del DSP) se compara con los resultados obtenidos en aritmética de punto flotante usando una señal de voz real. Como resultados del sistema en tiempo real se incluye la evaluación del desempeño en los tiempos de cálculo para una ventana de análisis de N=240 muestras de los bloques que integran a los procesos de análisis - síntesis en el DSP. Además se incluye una evaluación de un sistema para seguridad telefónica empleando el sistema de síntesis de voz.

6.1. Simulación en Aritmética de Punto Flotante

Tomando en cuenta los diagramas de flujo mostrados en las figuras (4.2) y (4.16) del capítulo cuatro se programó en aritmética de punto flotante el sistema de síntesis usando el entorno de programación que integra Matlab. La señal de voz para la simulación en aritmética de punto flotante es la oración "Hola, ¿cómo estas?". Tal simulación se llevó a cabo en tiempo congelado, es decir, la señal de voz es previamente almacenada en un archivo y posteriormente es procesada para reconstruirla de manera sintética. Para esta simulación sólo se toma en cuenta la calidad obtenida y el error entre la señal original y la señal reconstruida; omitiendo el desempeño de los tiempos de cálculo, carga computacional, y también los requerimientos de memoria.

A continuación presentamos la señal de voz real capturada y la señal de voz sintética en el dominio temporal para distintos valores de tamaño de la ventana: 160, 200, 240 y 256; lo que implica una duración de ventana de: 20, 25, 30 y 32.5 ms, respectivamente, con un traslape de un tercio de la longitud de ventana para cada caso en las figuras (6.1), (6.2), (6.3) y (6.4).
Para esta simulación fue necesario considerar que los valores de los parámetros para cada ventada son constantes, con valores de p=10, K=0.8, α=0.9375 y de 0.3 para el umbral de recorte central.

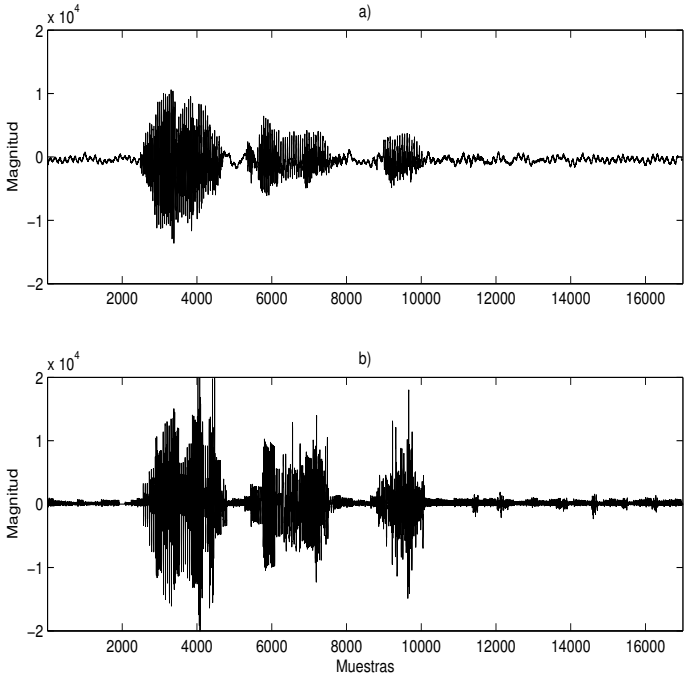

Figura 6.1: Señal de voz " Hola, ¿cómo estas? " a) Original, b) Señal sintetizada con longitud de ventana de 160 muestras.

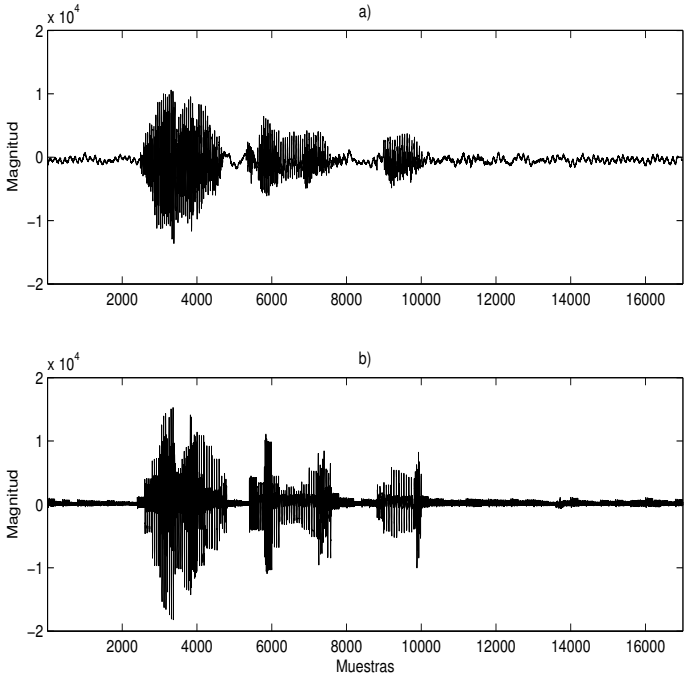

Figura 6.2: Señal de voz " Hola, ¿cómo estas? " a) Original, b) Señal sintetizada con longitud de ventana de 200 muestras.

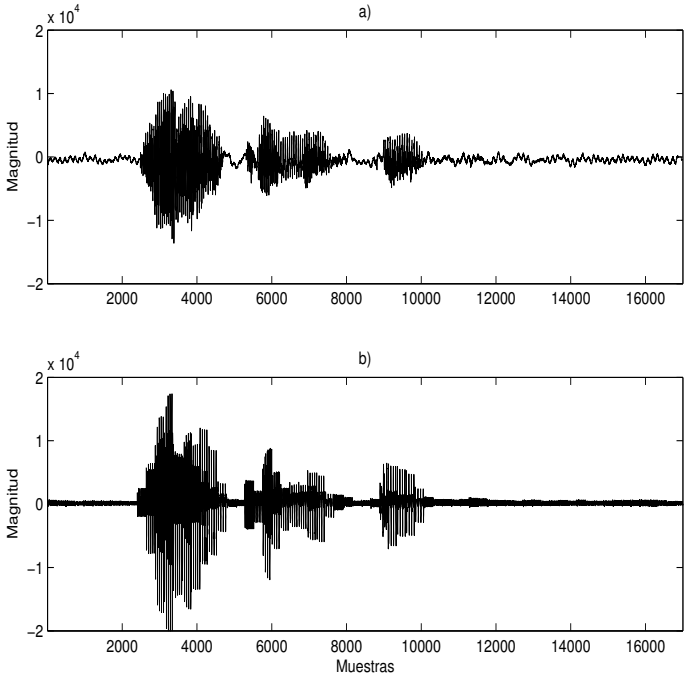

Figura 6.3: Señal de voz " Hola, ¿cómo estas? " a) Original, b) Señal sintetizada con longitud de ventana de 240 muestras.

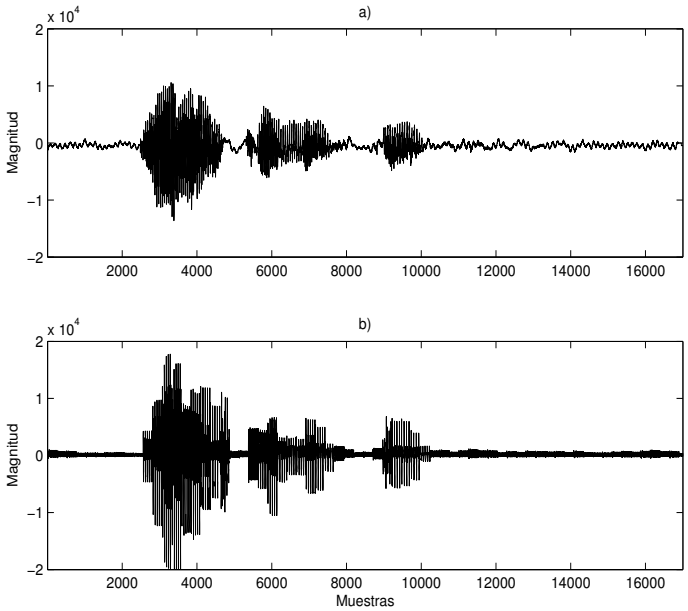

Figura 6.4: Señal de voz " Hola, ¿cómo estas? " a) Original, b) Señal sintetizada con longitud de ventana de 256 muestras.

Con valores de $p > 10$, la aproximación de los coeficientes para estimar el segmento es mejor; pero la carga computacional se ve incrementada en gran manera, siendo esto inpráctico. Al ir

modificando los valores de K, el cual determina el grado de recorte en la señal, en el método de recorte central, se observa que la propiedad de periodicidad se mantiene. Si los valores del umbral de decisión del método de recorte central son mayores a 0.3 (por ejemplo 0.6), la tendencia a considerar segmentos no voceados será alta. Esto contrasta en el caso en que los valores del umbral de decisión del método de recorte central son menores a 0.3, por lo que la tendencia de decisión se verá inclinada a considerar los segmentos de naturaleza voceada.
De lo anterior se puede concluir que se tendrían que modificar de manera muy drástica o extrema, los valores de estos parámetros, por lo que es necesario mencionar que los intervalos y valores de los parámetros involucrados en la síntesis de voz, deberán estar dentro de las consideraciones presentadas en las secciones anteriores.

Así, vemos que la calidad de los resultados de la simulación depende, en gran medida, de los valores del tamaño de la ventana y de la longitud del traslape entre ellas.

Con longitud de ventana igual a 240 muestras, observamos que la calidad en la reproducción del segmento de voz sintética, era bastante aceptable e inteligible.

Debido a ello, presentamos su respectivo espectro (figura 6.5) y espectrograma (figura 6.6), comparado con la señal de voz original. Al comparar los espectrogramas de la señal de voz original con la señal de voz sintética se observa que la riqueza espectral de los segmentos bajo análisis se conservan y son similares para ambos casos.

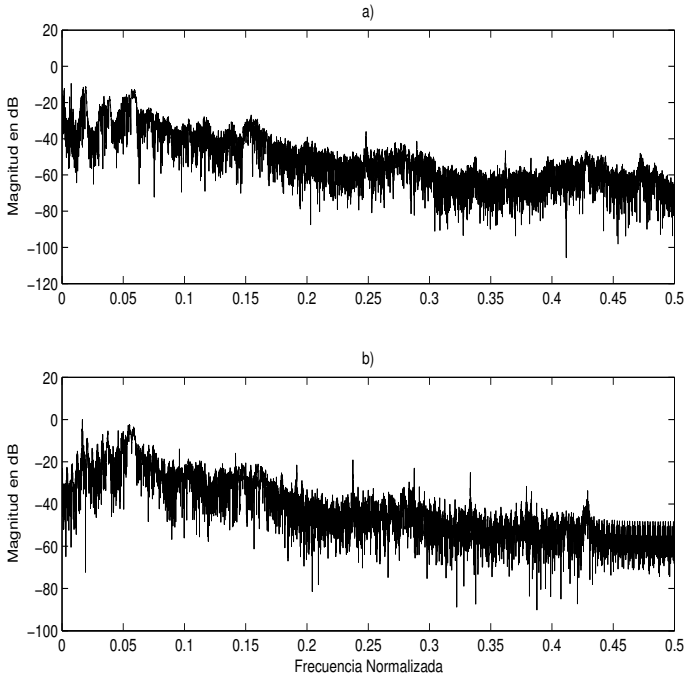

Figura 6.5: Espectro de la Señal de voz “ Hola, ¿cómo estas? ” a) Original, b) Señal sintetizada con longitud de ventana de 240 muestras.

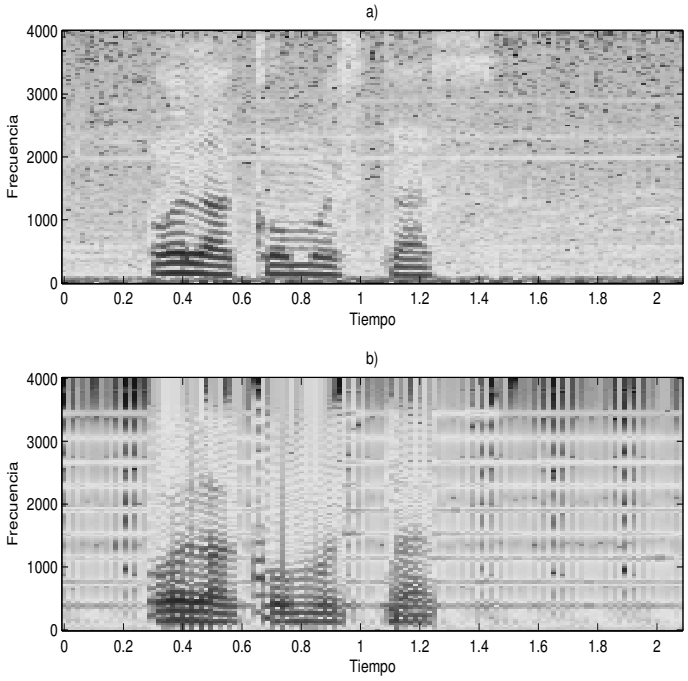

Figura 6.6: Espectrograma de la Señal de voz " Hola, ¿cómo estas? " a) Original, b) Señal sintetizada con longitud de ventana de 240 muestras.

6.2. Simulación en Aritmética de Punto Fijo

Para la simulación en aritmética de punto fijo se implantó el sistema de síntesis de voz empleando el DSP y el CCS para la visualización de los resultados. Se almacenó en memoria DARAM del DSP una señal de voz real de la palabra "rojo" con una longitud de 4000 muestras con una frecuencia de muestreo de f_s=8000 kHz. Con base a los resultados obtenidos en la simulación en aritmética flotante, se presentan 12 pruebas (cuadro 6.1) que consisten en variar los parámetros[1] N y, $N_{traslape}$, manteniendo constantes los parámetros K, α, mostrando así la calidad[2] de la reconstrucción de la señal sintética.

Después de haber efectuado múltiples pruebas con la simulación en punto flotante, se llegó a la conclusión que para el sistema de síntesis de voz a realizar, se requiere una longitud de ventana de N=240, un traslape de un 25 % y los parámetros K=0.8, α=0.9375 y p=10 para obtener una calidad aceptable y óptima, ya que el sistema ofrece mejores desempeños en cuanto a tiempo de cálculo y no existen conflictos en los niveles de pipeline del DSP[3]. En la figura (6.7) se presenta la señal sintetizada al emplear los valores de N=240, traslape de un 25 %, K=0.8, α=0.9375 y p=10 de la señal de voz real "rojo".

Cabe mencionar que para valores de $p > 10$, la aproximación en las frecuencias formantes en

[1]Explicados en el capítulo cuatro.
[2]Entendiendo el término calidad como una calificación subjetiva de la inteligibilidad de la voz reconstruida.
[3]El DSP C5402 tiene seis niveles pipeline dado que es una arquitectura tipo Harvard. Ver apéndice A.

la señal es mejor, pero la carga computacional aumenta[4]. De igual manera, los parámetros K y α no determinan en gran medida la calidad en la señal reconstruida, siempre y cuando estén en los intervalos de $0.6< K <0.8$ y $0.9< \alpha <0.95$.

Prueba	N	$N_{traslape}$	K	α	p	Calidad
1	160	0	0.8	0.9375	10	baja
2	160	40	0.8	0.9375	10	baja
3	160	60	0.8	0.9375	10	baja
4	200	0	0.8	0.9375	10	baja
5	200	40	0.8	0.9375	10	baja
6	200	60	0.8	0.9375	10	baja
7	240	0	0.8	0.9375	10	regular
8	240	40	0.8	0.9375	10	regular
9	240	60	0.8	0.9375	10	buena
10	256	0	0.8	0.9375	10	regular
11	256	40	0.8	0.9375	10	buena
12	256	60	0.8	0.9375	10	buena

Cuadro 6.1: Variación de los parámetros N y $N_{traslape}$ en la simulación en punto fijo.

6.3. Evaluación del Desempeño del Sistema de Voz en Tiempo Real.

En el cuadro (6.2) se muestran los tiempos de ejecución para cada subproceso que integra a los procesos de análisis y síntesis, para un segmento de voz de $N = 240$ muestras. El ciclo de instrucción del C5402 es de 10 ns por lo que si muestreamos a una f_s=8000 Hz, el tiempo en adquirir 240 muestras es de 30 ms. Entonces del cuadro (6.2) se observa que los procesos de análisis y síntesis tienen un tiempo de ejecución de 1.0473 ms y 3.5474 ms, respectivamente. Así pues, los procesos de análisis - síntesis se ejecutan en un 15.31 % del tiempo de adquisición para N=240 muestras. Lo anterior representa que no habrá conflictos entre los tiempos de cálculo de los procesos de análisis - síntesis entre ventanas sucesivas y que la arquitectura seleccionada es suficiente para esta aplicación.

La estimación de los tiempos de cálculo se efectuó con ayuda de una aplicación de medición de ciclos de instrucción integrada en el CCS.

En el cuadro (6.3) se presenta la evaluación del desempeño de la carga computacional del sistema de síntesis de voz a partir del número de multiplicaciones necesarias al emplear diferentes longitudes de ventana y ordenes en la predicción (Levinson - Durbin), así como el número de multiplicaciones utilizadas en el filtro Todo Polo. Del cuadro (6.3) se observa que al aumentar el orden p aumenta la calidad en la señal reconstruida, sin embargo la carga computacional es incrementada, y por cuestiones prácticas es preferible optar por un orden menor (p=10) que de cualquier manera resulta en una calidad aceptable.

[4]En la siguiente sección se presenta el cuadro (6.3) que ejemplifica esto.

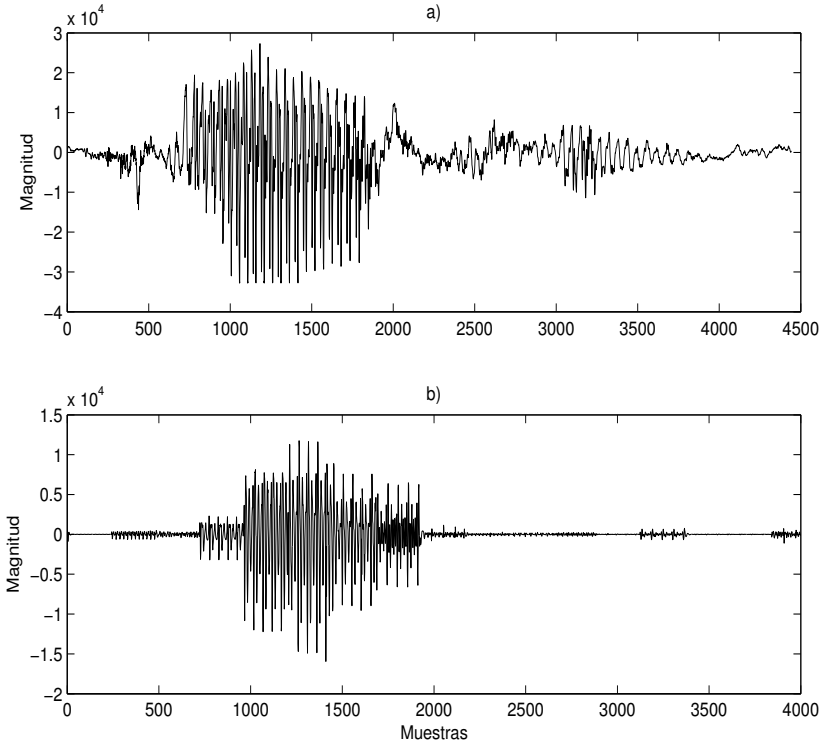

Figura 6.7: Comparación entre a) señal de voz real de la palabra rojo y b) señal sintética en punto fijo con calidad aceptable.

Subproceso	Tiempo (ms)
proceso de ventaneo	0.0762
proceso de estimación de pitch	0.527
Filtrado de preénfasis	0.0835
Vector de Autocorrelación	0.0356
Recursión de Levinson - Durbin	0.325
Generador de tren de impulsos y lectura de ruido	0.051
Filtro Todo Polo	3.42
Filtrado de deénfasis	0.0764

Cuadro 6.2: Tiempos de ejecución para cada subproceso en el DSP.

6.4. Comparación entre aritmética de punto flotante y fijo

Se presenta una prueba para comparar el desempeño del sistema de síntesis de voz empleando el DSP con respecto al resultado obtenido al emplear aritmética de punto flotante (Matlab). La señal de voz real usada fue la palabra "rojo" muestreada a una frecuencia de f_s= 8 kHz. En la figura (6.9) se presenta la simulación tanto en punto flotante como en punto fijo, donde se observa la gran similitud de la voz sintética obtenida.

El criterio para la estimación del error entre la simulación en aritmética de punto flotante y

N	p	# de Multiplicaciones (Levinson - Durbin) p^2	# de Multiplicaciones (Filtro Todo Polo) pN	Calidad
160	8	64	1280	baja
160	10	100	1600	baja
160	12	144	1920	baja
160	24	576	3840	regular
200	8	64	1600	baja
200	10	100	2000	regular
200	12	144	2400	regular
200	24	576	4800	regular
240	8	64	1920	regular
240	10	100	2400	buena
240	12	144	2880	buena
240	24	576	5760	óptima
256	8	64	2048	regular
256	10	100	2560	buena
256	12	144	3072	buena
256	24	576	6144	óptima

Cuadro 6.3: Evaluación del desempeño del sistema en relación con la longitud de la ventana N y el orden de la predicción p.

fijo fue el criterio del error cuadrático promedio mostrado en la figura (6.8). Por tanto, de la figura (6.8) se observa que el error cuadrático promedio es del orden de 6 % entre la señal sintética en punto flotante y fijo, lo que garantiza un buen desempeño en la precisión numérica al emplear aritmética de punto fijo.

En la figura (6.10) se presenta el espectro obtenido de la simulación en punto flotante y el DSP (punto fijo); y se observa una gran similitud en la información frecuencial de la señal de prueba, en donde la diferencia principal del espectro obtenido en punto flotante y el DSP se debe a la precisión numérica usada, es decir, el espectro obtenido en la simulación en el DSP presenta un menor rango dinámico debido al formato en punto entero q_{15} empleado.

La variación de la información frecuencial con respecto al tiempo se observa en la figura (6.11); en dicha figura se presenta el espectrograma obtenido tanto de la señal original como de la simulación realizada en punto flotante y fijo. El espectrograma obtenido, tanto en punto flotante como en el DSP, guarda gran parecido con el espectrograma de la señal original, presentando sólo discontinuidades debido a que la respuesta en frecuencia de los parámetros estimados por cada ventana suavizan la envolvente espectral con respecto a la ventana en análisis de la señal original.

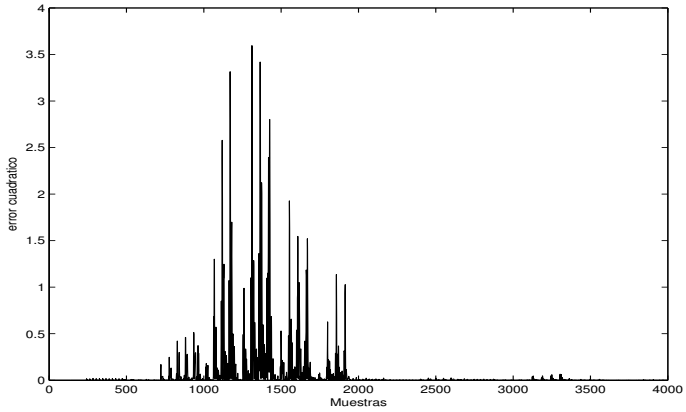

Figura 6.8: Error cuadrático promedio entre la señal en aritmética de punto flotante y fijo.

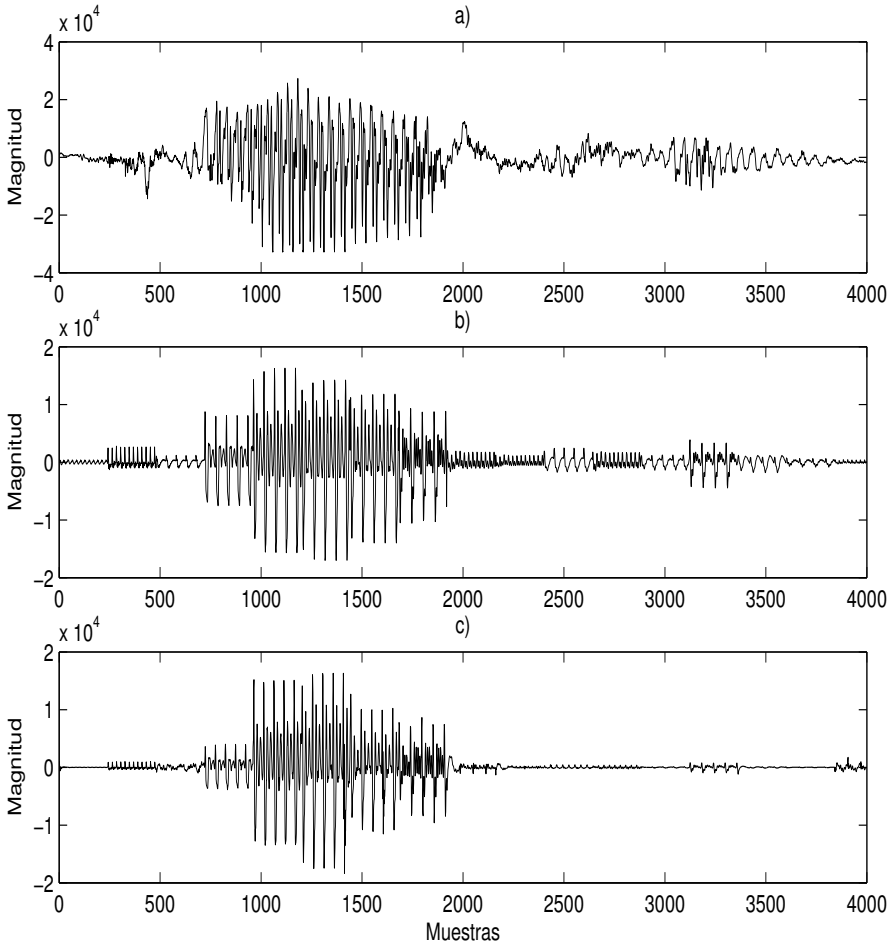

Figura 6.9: Simulación entre aritmética de punto flotante y fijo. a) señal original, b) señal sintetizada de punto flotante, c) señal sintetizada de punto fijo.

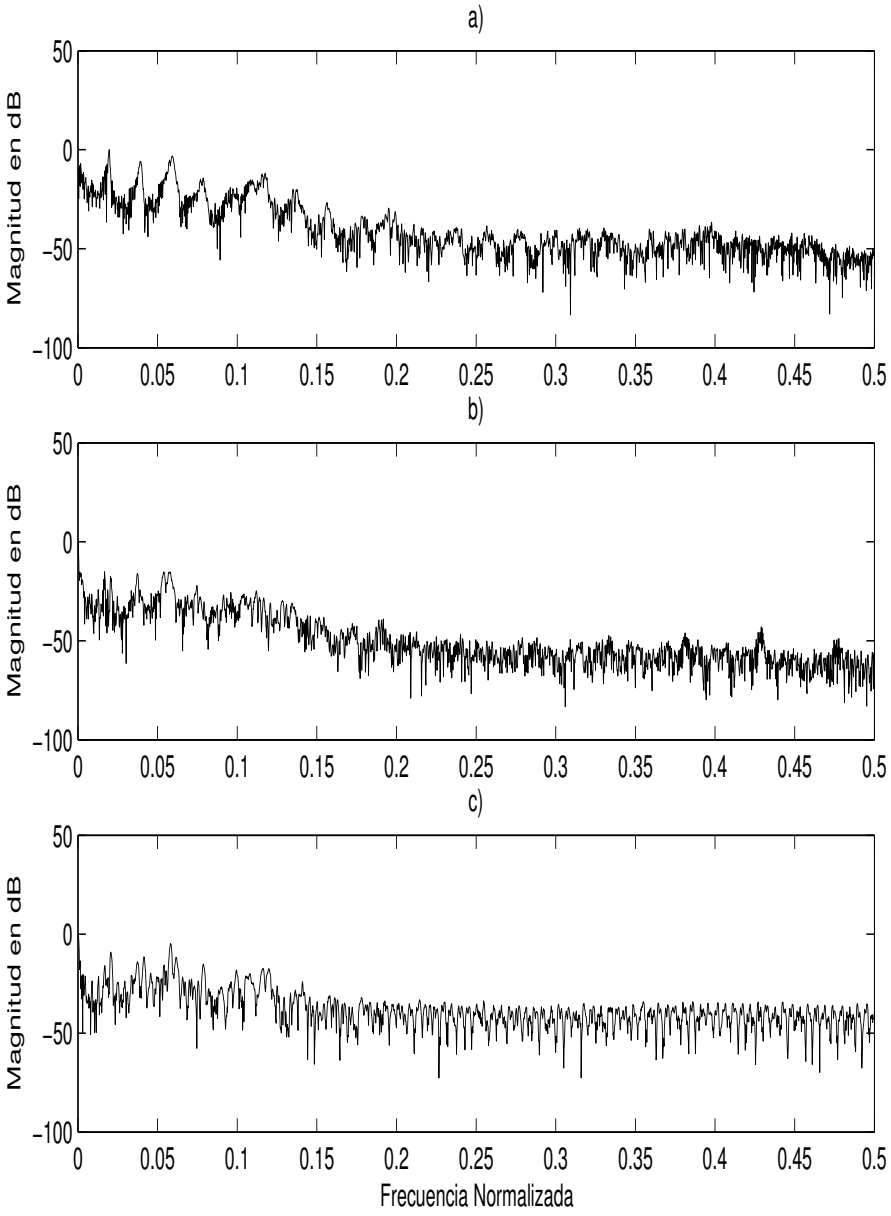

Figura 6.10: Espectro de la señal. a) señal original, b) señal sintetizada de punto flotante, c) señal sintetizada de punto fijo.

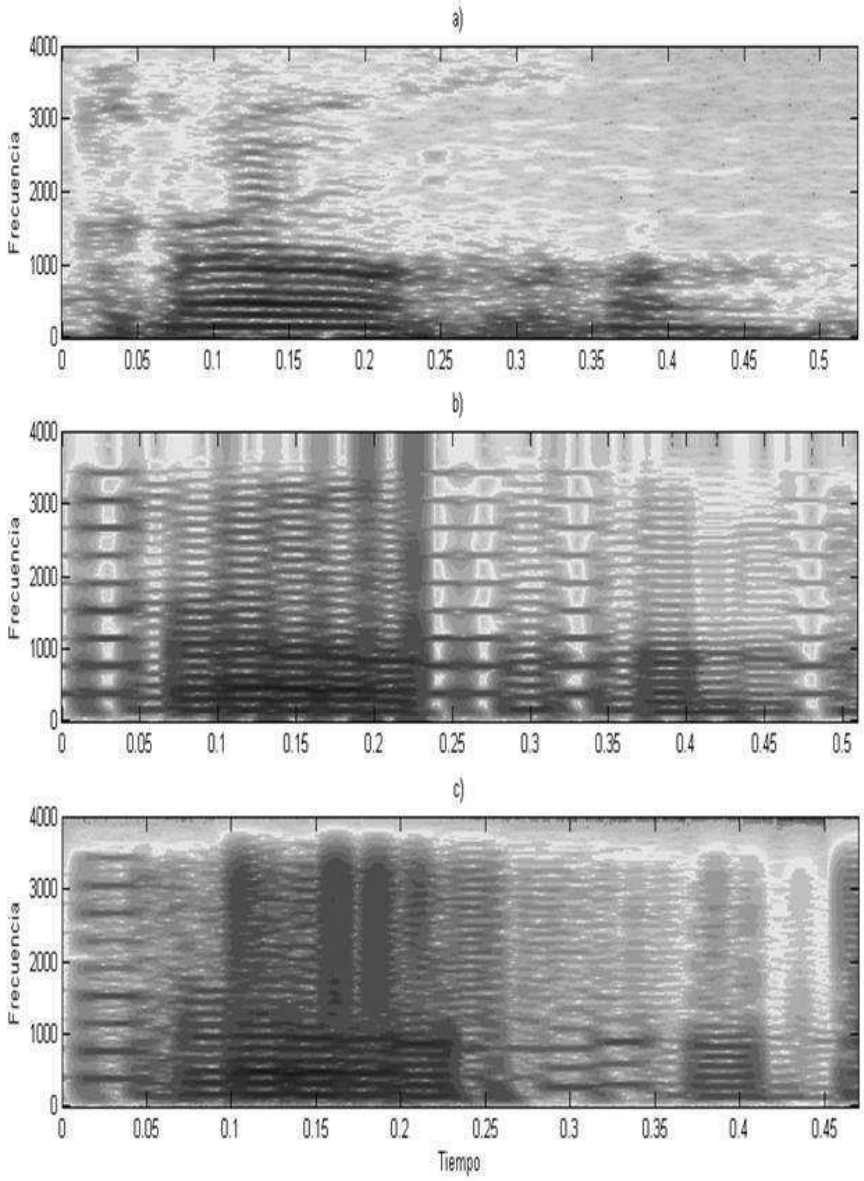

Figura 6.11: Espectrograma de la señal. a) señal original, b) señal sintetizada de punto flotante, c) señal sintetizada de punto fijo.

6.5. Evaluación de un sistema para seguridad telefónica empleando el sistema de síntesis de voz

Como hemos mencionado en el presente trabajo de tesis, uno de los objetivos particulares de la misma es que el Sistema de síntesis de Voz en tiempo real propuesto y diseñado sea adaptado e integrado a sistemas de mayor dimensión, por lo que en ésta sección proponemos la implantación y evaluación de un sistema de seguridad telefónica, empleando nuestro diseño.

Actualmente, las aplicaciones que involucran en sus procesos a la voz se han extendido de tal modo que podemos contemplarlas en sistemas, como:

- Robots controlados por voz
- Reconocimiento de voz
- Transmisión de voz en Internet (Voice Over IP)
- Teléfonos celulares
- Control de procesos mediante la voz.

Según el reporte del Instituto Nacional de Estadística, Geografía e Informática, en el año 2000 el porcentaje nacional de disponibilidad de abonados telefónicos por vivienda en México fue del 36.2 %, es decir, 21,858,085 viviendas particulares habitadas, sin mencionar la disponibilidad de abonados en el sector privado, ni en la administración pública [18]. Debido al crecimiento en cuanto a su uso, vemos que "la red telefónica es la de mayor cobertura geográfica, la que mayor número de usuarios tiene, y ocasionalmente se ha afirmado que es el sistema más complejo del que dispone la humanidad" [2]. Además, nos permite establecer una llamada entre dos usuarios en cualquier parte del planeta de manera automática y prácticamente instantánea. Sin embargo, conforme al crecimiento de los sistemas de comunicación, el número de los casos reportados de espionaje telefónico en nuestro país se han venido incrementando, por lo que uno de las motivos de esta sección, es la describir de manera breve, una propuesta de implantación de un sistema de seguridad telefónica en la que se pueda observar la integración del sistema de síntesis de voz diseñado e implantado en el presente trabajo de tesis [19] [20] [21].

Inicialmente, la red telefónica, también conocida como Red Telefónica Conmutada Pública (Public Switched Telephone Network), se concibió para comunicaciones verbales; pero dada la gran infraestructura existente, su uso, y el desarrollo tecnológico en los últimos tiempos, es el medio más práctico disponible para la comunicación de datos, a pesar de no estar diseñada ni ser adecuada para ello [22] [23][24] [25]. Cabe mencionar que este crecimiento demanda una mayor integridad y seguridad, en cuanto a la información que se manipula [26].

6.5.1. Descripción del problema y propuesta de solución

La gran mayoría de las intervenciones a los aparatos, a las líneas (enlaces entre los abonados y las centrales) y a los sistemas telefónicos, se basan en el alambrado de circuitos y en el empleo de sistemas de transmisión en alta frecuencia que son difícilmente detectables [19][20][21]. Es debido

a lo anterior, que en esta sección proponemos un sistema de seguridad telefónica que se enfoca en trasmitir sobre la red telefónica los parámetros con los cuales es posible generar voz sintética. Sin embargo, para transmitir dichos parámetros sobre la infraestructura telefónica, es necesario adecuar las señales de tal modo que puedan ser enviadas por el emisor sobre la línea telefónica y ser recuperadas en el otro extremo por el receptor. Por lo tanto, proponemos utilizar un *modem*, (término que viene de la contracción de las palabras modulador y demodulador). El módem es la interfaz analógica - digital que se encarga de establecer los protocolos necesarios, modular las señales a transmitir, determinar la velocidad de transmisión, el sistema de detección de errores, etc. para poder transmitir o recibir datos a través de la línea telefónica [22][23][24][25].

Otra razón por la cual se propone un modem para la transmisión de los parámetros de la voz sintética es que la salida del *codec*, que está conectada al *DAA* (interfaz telefónica de la tarjeta de desarrollo DSK[5]), es de tipo analógica y el diseño y la implantación de un modem en el *DSP* es una tarea que al implantarla consumiría la gran mayoría de los recursos de memoria y la carga computacional en el *DSP*. En la figura (6.12) se muestra el esquema general del sistema de seguridad telefónica que se propone.

De la figura (6.12) se observan las tareas a realizarse de manera general:

- La tarjeta 1 se encarga de transmitir la información a la tarjeta 2. La voz en la tarjeta 1 es adquirida por el codec que está integrado en ella y ejecuta el proceso de análisis. Posteriormente, mediante una aplicación del CCS, llamada RTDX (intercambio de datos en tiempo real) los parámetros son transferidos a una terminal PC (localidades de memoria temporales en la PC) para su transmisión a la red pública telefónica de manera modulada. Dicha modulación es efectuada por el modem.
- Mientras la tarjeta 1 envía los parámetros de la voz, la tarjeta 2 se encarga de recibirlos y reconstruir la voz (proceso de síntesis), mientras adquiere, por medio del codec, la voz real que el usuario de la tarjeta 2 para aplicarle el proceso de análisis y enviarle datos al usuario de la tarjeta 1.
- Las tareas anteriormente citadas, se aplican a las dos tarjetas de manera indistinta para realizar una comunicación full - duplex.

Como ya hemos mencionado, la herramienta con en la que se implantó el Sistema de síntesis de Voz (en el DSP TMS320C5402) es controlada por medio de la interfaz gráfica de programación *Code Composer Studio*. Dicha interfaz contiene una herramienta conocida como RTDX[6] con la cual es posible transmitir en *tiempo real* datos entre una terminal (PC) y el DSP. Una de las características del RTDX es que es posible desarrollar aplicaciones en lenguaje orientado a eventos (Visual Basic de Microsoft) o en lenguaje de alto nivel (C++).

En los diagramas de flujo presentados en las figuras (6.13) y (6.14) se observan las modificaciones que se hacen al programa principal del sistema de síntesis de Voz para su integración al sistema de seguridad telefónica. Al procedimiento de análisis se le denomina *proceso de codificación*, mientras que al proceso de síntesis se le denomina *proceso de decodificación*.

[5]Ver apéndice A.
[6]Ver apéndice B.

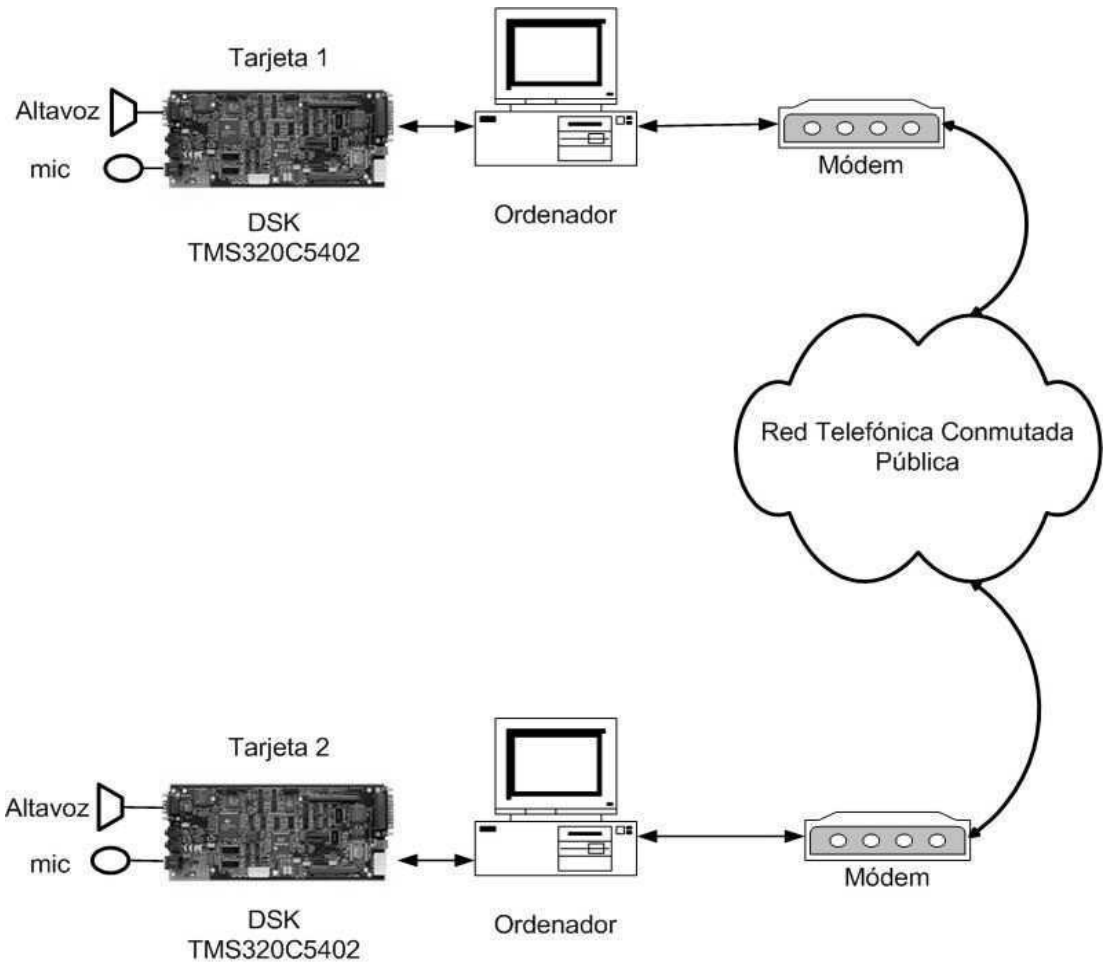

Figura 6.12: Esquema General del Sistema de Seguridad Telefónica propuesto.

En la figura (6.13) se muestra el diagrama de flujo para los procesos de codificación y transmisión del sistema de seguridad. A continuación se da una breve descripción de los bloques que integran a este procedimiento.

- Definición e inicialización de variables: Se definen e inicializan las variables y los arreglos necesarios para la ejecución de las tareas en el procedimiento de codificación y transmisión
- Configuración e inicialización del Codec y DAA: Es necesario la inicialización del Convertidor A/D y D/A y la interfaz de puerto telefónico.
- Configuración de Interrupciones: Se habilitan las interrupciones de recepción por hardware de los puertos serie que están conectados al codec y al DAA.
- Se espera a que una llamada sea realizada, por lo que se descuelga y se efectúa la marcación.
- Bandera de sincronía 1: Esta variable (flag1) funge como bandera para determinar si el arreglo está lleno (adquisición) dependiendo de la longitud del segmento que sea empleada.

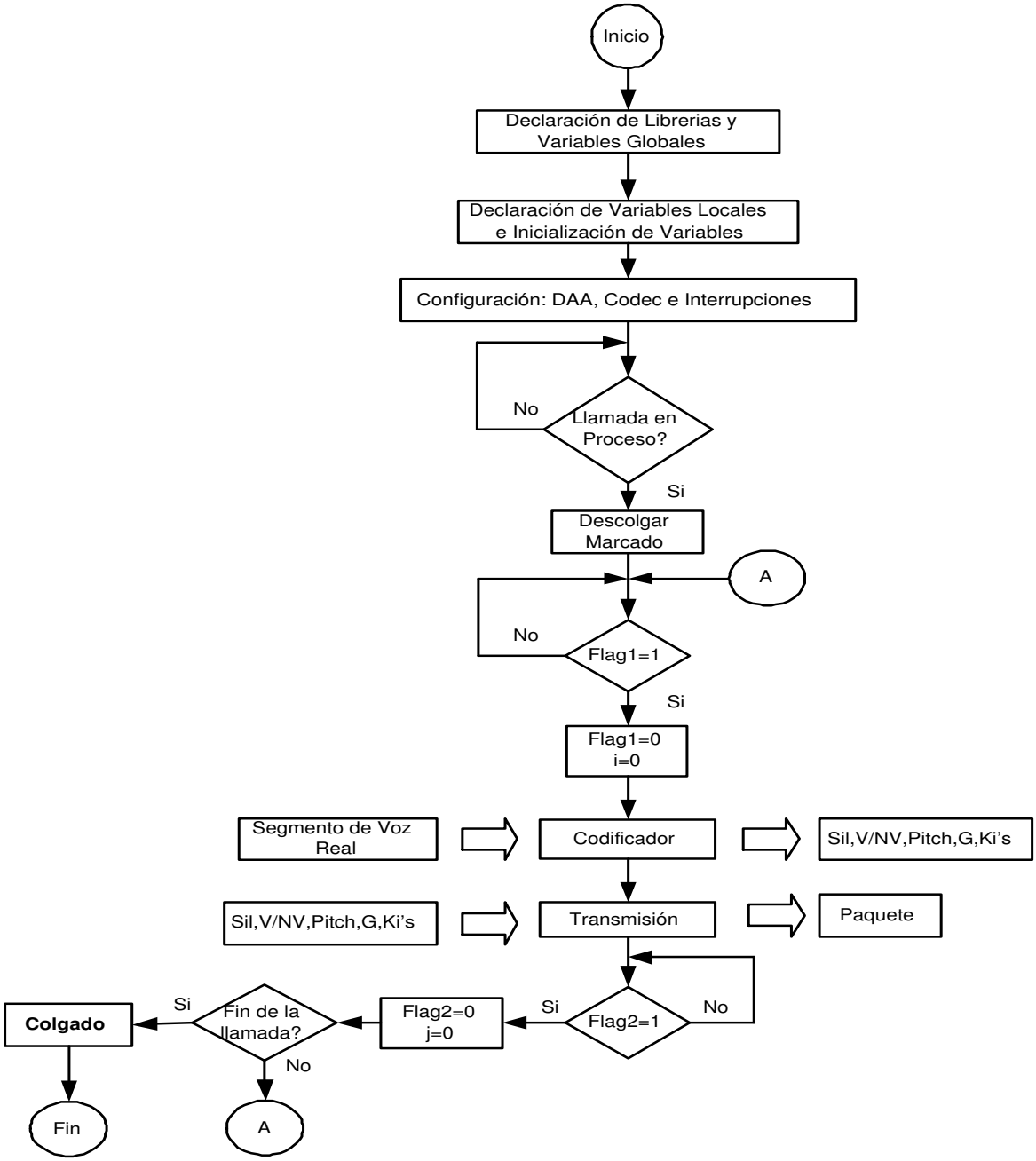

Figura 6.13: Diagrama de Flujo del para los procesos de codificación y transmisión.

- Proceso de codificación: Se ejecutan los bloques correspondientes al proceso de análisis.

- Proceso de Transmisión: los parámetros estimados definen un paquete de datos que por medio de la aplicación RTDX son transferidos al modem para su transmisión a través de la línea pública telefónica.

- Bandera de sincronía 2: Esta variable (flag2) funge como bandera para determinar si el arreglo fue totalmente transmitido a la aplicación RTDX.

- Se verifica si la llamada no ha terminado, de lo contrario se adquiere la siguiente ventana de voz para la estimación y trasmisión de sus parámetros.

En la figura (6.14) se muestra el diagrama de flujo para los procesos de decodificación y recepción del sistema de seguridad. A continuación se da una breve descripción de los bloques que integran a este procedimiento.

- Definición e inicialización de variables: Se definen e inicializan las variables y los arreglos necesarios para la ejecución de las tareas en el procedimiento de codificación y transmisión.

- Configuración e inicialización del Codec y DAA: Es necesario la inicialización de los convertidores A/D y D/A y la interfaz de puerto telefónico.

- Configuración de Interrupciones: Se habilitan las interrupciones de recepción por hardware de los puertos serie que están conectados al codec y al DAA.

- Se espera a que una llamada sea realizada, por lo que se descuelga.

- Bandera de sincronía 3: Esta variable (flag3) funge como bandera para determinar si el arreglo auxiliar correspondiente a la recepción de parámetros está lleno. La recepción de parámetros es por medio de un enlace con la aplicación RTDX (de los parámetros demodulados anteriormente por el modem) y el DSP.

- El paquete de datos que contiene los parámetros estimados en los procedimientos de codificación se lee y se almacena en memoria del DSP.

- Proceso de decodificación: Se ejecutan los bloques correspondientes al proceso de síntesis.

- Se verifica si la llamada ha terminado, de lo contrario se adquiere el siguiente arreglo de datos correspondientes a los parámetros recibidos por la aplicación RTDX.

Cabe mencionar que los procedimientos mostrados en los diagramas de flujo (6.13) y (6.14) son ejecutados de manera simultánea en cada DSP para establecer un sistema de tipo full - duplex.

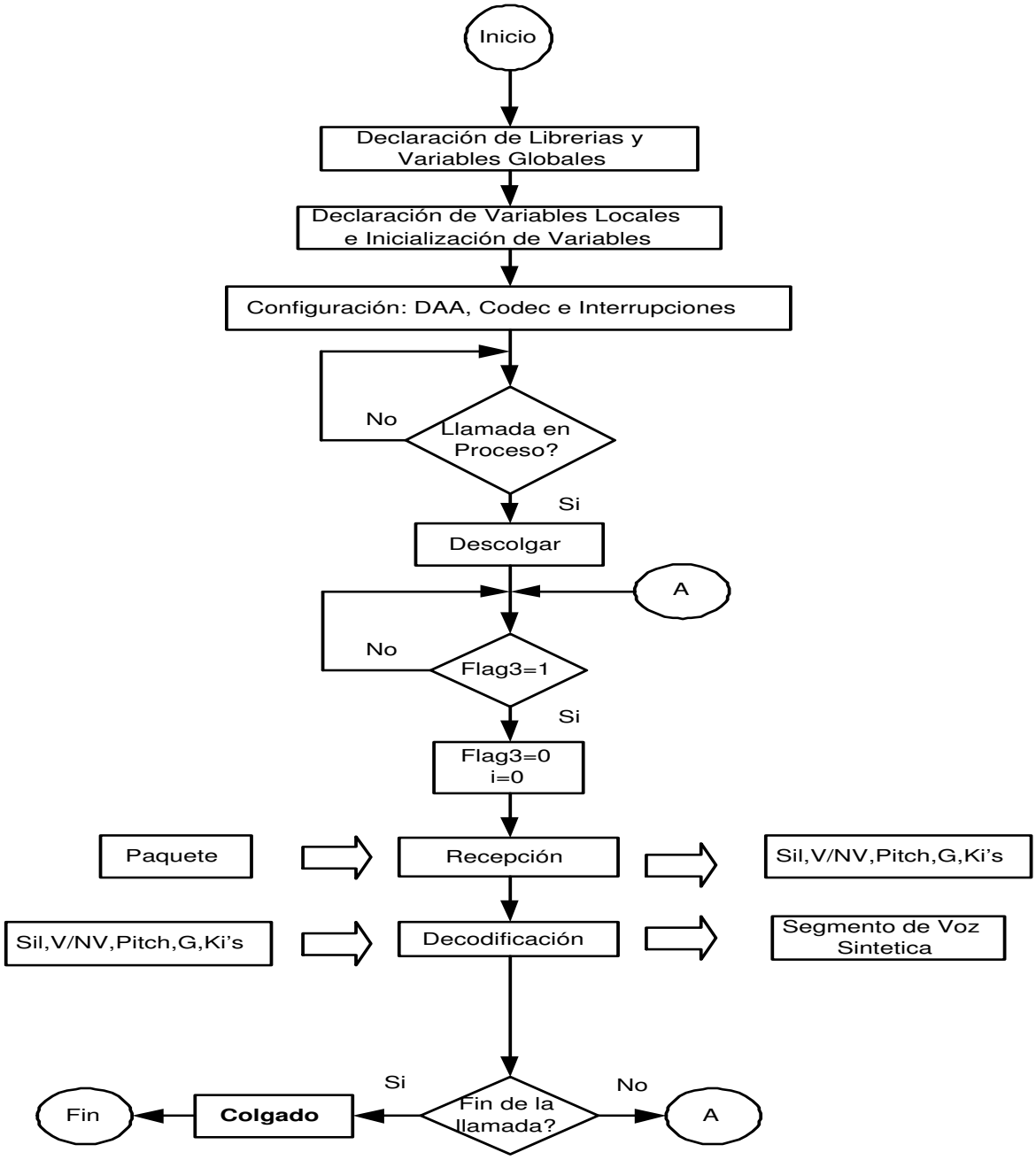

Figura 6.14: Diagrama de Flujo del para los procesos de decodificación y recepción.

6.6. Resumen

En este capítulo se mostraron los resultados obtenidos del actual trabajo de tesis. Se presentaron, los resultados de las simulaciones tanto en punto flotante como en punto fijo para los procesos de análisis y síntesis de voz. Las consideraciones que se hicieron para tales simulaciones es que los segmentos de voz se encontraban almacenados tanto en un archivo (simulación en punto flotante) como en memoria del DSP (simulación en punto fijo). Se observa que los resultados de la comparación entre aritmética de punto flotante y fija no difieren en gran manera, sólo existe un error cuadrático promedio del 6 % por cuestiones de precisión numérica, que no afecta la calidad de la voz al momento de reconstruirse. También se presentaron los resultados concretos en cuanto al desempeño de tiempos de cálculo y carga computacional, así como una calificación de la voz sintética al variar los parámetros más significativos. Por tanto, el DSP TMS320C5402 resultó ser una herramienta adecuada para la implantación de nuestro sistema. En esta arquitectura es viable implantar procesos adicionales al sistema principal, con las cuales es posible ofrecer un valor agregado a cualquier sistema que se implante, por ejemplo, integrar el sistema de síntesis de voz a un sistema más complejo. En el caso particular de este capítulo, se propuso la integración de nuestro sistema de síntesis de voz a un sistema de seguridad telefónica. Para lograr esta integración es necesario efectuar cambios mínimos a la estructura principal del sistema de síntesis de voz. Así, observamos que el sistema de síntesis de voz cumple con los estándares más comunes de integración a otros sistemas, cumpliendo con el objetivo de ser un bloque modular de sencilla adaptación e integración a otros sistemas. Además, se hizo una breve descripción, para un caso real, de los componentes y procesos que se requerirían para efectuar un sistema de seguridad telefónica en tiempo real viable, que incluso, podría ser comercializable.

Capítulo 7

Conclusiones

A lo largo del presente trabajo de tesis, presentamos las estrategias y el diseño de un sistema de síntesis de voz en tiempo real, considerando su implantación en una arquitectura *DSP* de punto fijo (TMS320C5402); tomando como punto de partida el procesamiento digital de señales y, en el caso particular, enfocándonos en el procesamiento digital de la voz.

Se mostraron los tópicos relacionados con el sistema de producción de voz humana para establecer un modelo digital completo de producción de voz en tiempo discreto llamado modelo terminal - análogo.

Hicimos una descripción teórica sobre los procesos, algoritmos, y métodos de los vocoders comúnmente usados; decidiendo por su fácil implantación y aceptable desempeño, el Vocoder LPC. Para la realización de este Vocoder es necesario estimar los parámetros para cada segmento de voz (Bloque de análisis), que son: la existencia de silencio, la naturaleza del segmento de voz, el período de pitch, los coeficientes LPC y la ganancia. En la reconstrucción de la señal (Bloque de síntesis), por medio de éstos parámetros, utilizamos un filtro Todo Polo, el cual es excitado por un tren de impulsos con período igual al período de pitch o ruido blanco, dependiendo de la naturaleza del segmento bajo análisis (voceado - no voceado).

Para lograr la implantación del sistema de síntesis de voz, en primera instancia, desarrollamos e implantamos dicho sistema en un ambiente gráfico de programación y simulación de punto flotante en tiempo congelado, de tal manera que pudimos valorar y evaluar, tomándolo como punto de partida, los desempeños del sistema en punto flotante. Con las simulaciones efectuadas en punto flotante, llegamos a la conclusión de que se requiere una longitud de ventana de N=240, un traslape de un 25 % y los valores de los parámetros K=0.8, α=0.9375 y p=10 para obtener una calidad de voz aceptable y óptima.

Posteriormente realizamos la implantación del sistema en punto fijo (DSP), en un entorno de programación (CCS), tanto en tiempo congelado como en real. Al hacer esto pudimos determinar que la complejidad de la implantación del sistema en ambas plataformas depende, no sólo de los algoritmos a programar, sino también de la carga computacional, tiempos de cálculo y precisión numérica. Con los recursos del DSP, en cuanto a su tiempo de ejecución (100 MIPS) y memoria dato - programa (16k words), pudimos implantar el sistema de síntesis de voz en tiempo real; con una duración en los procesos análisis - síntesis de 4.5947 ms y usando 3364 localidades de memoria dato - programa del DSP.

El diseño del sistema en tiempo real se efectuó con un tiempo de ejecución de todos los procesos empleando el 15.31 % (4.594 ms.) del tiempo necesario para la adquisición de N=240 muestras, garantizando así un óptimo desempeño de nuestra implantación, que valida su posible implantación como un bloque modular partícipe en sistemas de mayor complejidad.

De los resultados obtenidos, observamos que la comparación entre aritmética de punto flotante y fija, no difieren en gran manera, estimando un error cuadrático promedio del 6 % debido a cuestiones de precisión numérica, que no afecta la calidad de la voz al momento de reconstruirse.

La compresión de la señal de voz original utilizando el método de síntesis, se alcanzó hasta en un 7.7 %, es decir, de 8000 muestras de voz adquiridas en un segundo, sólo se requerirían 616 parámetros/s para su posible transmisión y/o almacenamiento.

Con la arquitectura elegida (TMS320C5402) se tiene la opción de integrar nuestro sistema de síntesis de voz a sistemas de reconocimiento, seguridad o almacenamiento masivo puesto que el 85 % del tiempo de ejecución por ventana está libre (25.41 ms), con lo que no existirían conflictos en tiempos, carga computacional y precisión numérica para implantar lo anterior.

Este trabajo ha mostrado la implantación de un sistema de síntesis de voz en tiempo real en una arquitectura de procesamiento digital de señales DSP, comprobando la posibilidad real de su implantación y los desempeños del sistema en el DSP.

Apéndice A

Introducción al DSP TMS320C5402

El DSP TMS320VC5402, perteneciente a la familia C54x de Texas Instruments, es un procesador digital de señales de punto fijo, diseñado para la implantación de algoritmos y aplicaciones en tiempo real. Está construido bajo una arquitectura tipo Harvard modificada con un alto grado de paralelismo y bajo consumo de potencia, además de tener modos de direccionamiento versátiles y un conjunto de instrucciones que mejoran su desempeño. En este apéndice se describe la arquitectura del DSP, sus principales características y los bloques que lo constituyen.

Para mayores referencias en cuanto a la arquitectura y periféricos internos que integran al DSP TMS320C5402, así como el lenguaje en ensamblador y C, se recomienda consultar la bibliografía en [15], [16], [27], [28], [29], [30], [31], [32].

A.1. Características Generales

- Arquitectura de Multibus avanzado con tres buses separados para memoria dato (16 bits) y un bus para memoria programa.
- Unidad Aritmética lógica de 40 bits, incluyendo 2 acumuladores independientes de 40 bits y un bloque de corrimiento de 40 bits.
- Multiplicador en paralelo de 17 x 17 bits acoplado a un Sumador dedicado de 40 bits para la operación de Multiplicación - Acumulación (MAC) en un sólo ciclo de instrucción.
- Unidad de Comparación, Selección y Almacenamiento (CSSU) para codificación Viterbi.
- Codificador de Exponente para el cálculo del exponente de un valor de 40 bits en el acumulador en un sólo ciclo.
- Dos generadores de dirección con ocho registros auxiliares y dos unidades aritméticas de registros auxiliares (ARAU).
- Buses de datos con característica de retención ("Holding").
- Modo extendido de direccionamiento para direccionar hasta 1M x 16 bits.
- 4k x 16 bits en ROM interna.

- 16k x 16 bits en DARAM.
- Instrucciones:
  - para operaciones de repetición y bloques de repetición de código programa.
  - para el movimiento de bloques de memoria y para un manejo eficiente de datos y programa.
  - para operaciones con palabras de 32 bits.
  - para lectura de dos o tres operandos.
  - aritméticas con almacenamiento y carga en paralelo.
  - de almacenamiento condicional.
  - de retornos rápidos desde llamadas a interrupciones.
- Periféricos internos;
  - Generador de estados de espera programados por software y un banco de interrupción programable.
  - Generador de reloj con oscilador interno o reloj externo.
  - Circuito de malla de fase enlazada (PLL) programada por software.
  - 2 Puertos seriales bufereados multicanal (McBSPs).
  - Interfaz de puerto paralelo de tipo Huésped de 8 bits mejorado (HPI8)
  - 2 Temporizadores de 16 bits con preescalador de 4 bits.
  - Controlador de seis canales de acceso directo de memoria (DMA).
- Control de bajo consumo de potencia con instrucciones IDLE1, IDLE2 e IDLE3.
- Emula el estándar 1149.1 (JTAG) de IEEE.
- Velocidad a (100 MHz - 100 MIPS) y período de ciclo de reloj de 10 ns.
- Maneja seis niveles de pipeline: prebúsqueda, búsqueda, decodificación, acceso, lectura y ejecución.
- Voltaje de operación a 3.3 /1.8 V.

A.2. Arquitectura

El C5402 (figura A.1) usa una arquitectura avanzada tipo Harvard que optimiza el procesamiento por medio de sus tres buses independientes de memoria dato y un bus de memoria programa. Estos procesadores constan de una unidad aritmética lógica que ha sido diseñada para un alto grado de paralelismo, incluyendo unidades para aplicaciones especificas mediante Hardware (MAC,CSSU,ARAU, etc.), memoria interna y periféricos internos.

El separar la memoria programa y la memoria dato permite acceso simultáneo a las instrucciones del programa y datos teniendo con ello un alto grado de paralelismo Por ejemplo, se pueden realizar dos operaciones de lectura y una operación de escritura en un sólo ciclo. Por tanto, tal paralelismo efectúa en un solo ciclo de instrucción operaciones aritméticas, lógicas, y de manipulación de bits.

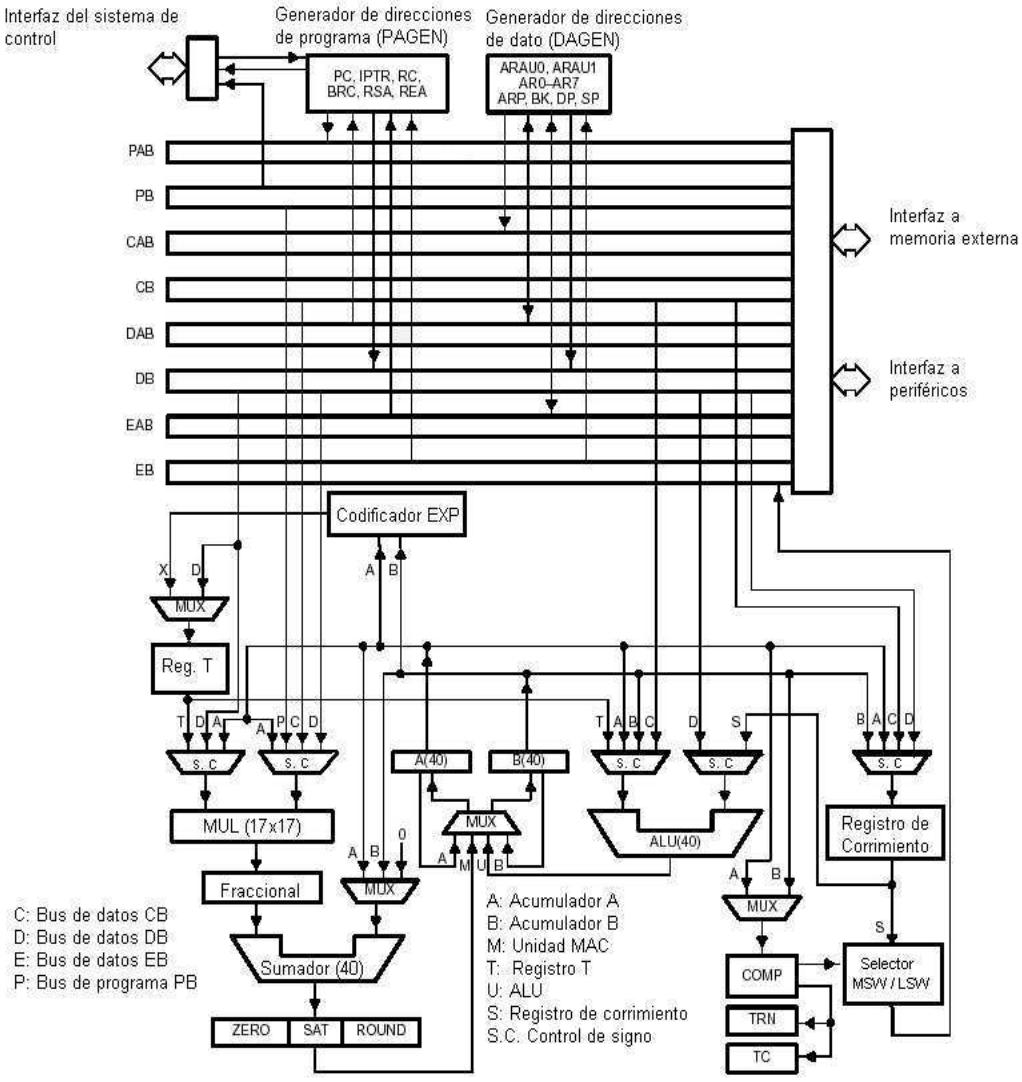

Figura A.1: Arquitectura del DSP TMS320c5402.

A.3. Estructura de Bus

La arquitectura del C5402 consta de 8 buses de 16 bits, 4 buses para datos y programa, y 4 buses para direcciones:

- El bus de programa (PB) transporta el código de instrucción y los operandos inmediatos desde la memoria programa.
- Tres buses de datos (CB, DB, EB) interconectan varios elementos, tales como el CPU, los periféricos internos y la memoria dato.
- Los buses CB y DB transportan los operandos que son leídos desde la memoria dato.
- El bus EB transporta los datos para ser escritos en memoria.

- Cuatro buses de direcciones (PAB, CAB, DAB y EAB) transportan las direcciones necesarias para la ejecución de las instrucciones.

A.4. Organización de Memoria

El DSP C5402 dispone de memoria ROM y RAM para ayudar en el rendimiento e integración del sistema.

Memoria ROM Interna

EL C5402 contiene 4k-palabras x 16 bits de memoria ROM interna. Un programa bootloader está disponible para el C5402 desde las direcciones F800h - FBFFh de la ROM, de tal manera que puede ser configurada dicha opción para transferir automáticamente código del usuario desde una fuente externa en cualquier posición del mapa de memoria y ejecutarse a la hora de resetear al DSP. Si el pin MP/MC está puesto a cero durante un reset de hardware, la ejecución del código empezará desde la localidad FF80h de la memoria ROM. En esta localidad contiene una instrucción de salto hacia el inicio del programa del bootloader. La opción bootloader del C5402 proporciona diferentes maneras para cargar el código alojando varios requerimientos del sistema:

- En paralelo, desde localidades de 8 o 16 bits de una memoria EPROM externa.
- En paralelo, desde espacios I/O de 8 o 16 bits.
- En arranque serial, desde puertos seriales con modo de 8 o 16 bits.
- En arranque de interface de puerto huésped.

La distribución de las localidades en la memoria ROM (figura A.2) es la siguiente:

- Un programa bootloader que se carga desde puerto serie, memoria externa, puertos I/O o interfaces de puertos huésped
- Una tabla de la ley μ de 256 valores.
- Una tabla de la ley A de 256 valores.
- Una tabla de la función seno de 256 valores.
- Una tabla de vectores de interrupción.

Memoria RAM Interna

En cuanto a memoria RAM interna, el C5402 contiene 16k palabras x 16 bits localidades del tipo de doble acceso (DARAM). La DARAM esta compuesta de 2 bloques de 8k palabras cada uno. Cada bloque puede soportar 2 operaciones de lectura en un ciclo, o una operación de lectura y una de escritura en un ciclo. La DARAM está localizada en el intervalo de direcciones de 0060h-3FFFh en memoria dato, y puede ser mapeada dentro de la memoria programa/dato al poner en uno al bit OVLY (registro PMST).

La distribución del mapa de memoria se muestra en la figura (A.3).

Direcciones	Descripción
F000h – F7FFh	Reservado
F800h – FBFFh	Programa Bootloader
FC00h – FCFFh	Tabla de Ley mu
FD00h – FDFFh	Tabla de ley A
FE00h – FEFFh	Tabla de senos
FF00h – FF7Fh	Reservado
FF80h – FFFFh	Tabla de vectores de interrupción

Figura A.2: Disposición de la memoria ROM interna.

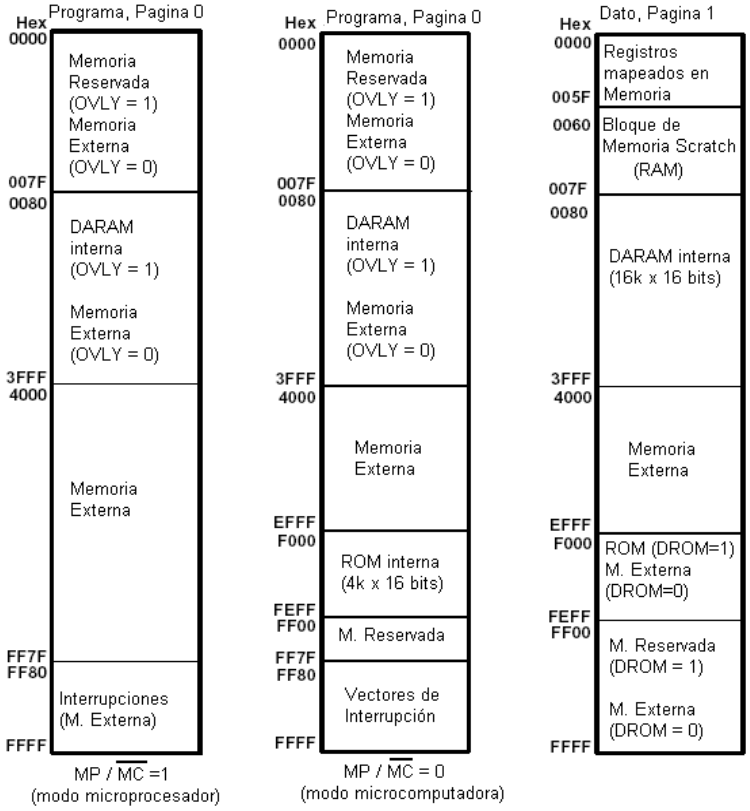

Figura A.3: Mapa de Memoria.

Registros mapeados en Memoria

El C5402 consta de 27 registros de CPU mapeados en memoria (MMRs), los cuales se localizan en memoria dato desde la localidad 0h hasta 1Fh, y se muestran en la figura (A.4). Además consta

de otros registros mapeados en memoria asociados con los periféricos mostrados en la figura (A.5).

NOMBRE	DIRECCIÓN		DESCRIPCIÓN
	DEC	HEX	
IMR	0	0	Registro de Interrupción Enmascarable
IFR	1	1	Registro de Bandera de Interrupción
–	2–5	2–5	Reservado para pruebas
ST0	6	6	Registro de Estado 0
ST1	7	7	Registro de Estado 1
AL	8	8	Parte Baja de la palabra del Acumulador A (15-0)
AH	9	9	Parte Alta de la palabra del Acumulador A (31-16)
AG	10	A	Bits de Guarda del Acumulador A (39-32)
BL	11	B	Parte Baja de la palabra del Acumulador B (15-0)
BH	12	C	Parte Alta de la palabra del Acumulador B (31-16)
BG	13	D	Bits de Guarda del Acumulador B (39-32)
TREG	14	E	Registro Temporal
TRN	15	F	Registro Transitorio
AR0	16	10	Registro Auxiliar 0
AR1	17	11	Registro Auxiliar 1
AR2	18	12	Registro Auxiliar 2
AR3	19	13	Registro Auxiliar 3
AR4	20	14	Registro Auxiliar 4
AR5	21	15	Registro Auxiliar 5
AR6	22	16	Registro Auxiliar 6
AR7	23	17	Registro Auxiliar 7
SP	24	18	Registro de Apuntador de Pila
BK	25	19	Registro de Tamaño para Buffer Circular
BRC	26	1A	Contador de Repetición de Bloque
RSA	27	1B	Dirección Inicial de Repetición de Bloque
REA	28	1C	Dirección final de Repetición de Bloque
PMST	29	1D	Processor mode status (PMST) register
XPC	30	1E	Registro de Paginación de Programa Extendido
–	31	1F	Reservado

Figura A.4: Registros Mapeados en Memoria.

Tabla de Vectores de Interrupción

Al resetear al DSP, las interrupciones y los vectores "trap"son direccionados en memoria programa. Al efectuarse la llamada de servicio de interrupción, el contador de programa (PC) es cargado con la dirección de la localidad del vector de interrupción en cuestión. Por tanto, se reservan cuatro localidades para cada vector de interrupción y de esta manera se efectúa la instrucción de salto con retardo, realizando con ello, apropiadamente, la rutina de servicio a la interrupción. Después del reset, las interrupciones y los vectores "trap"son mapeados a partir de la localidad FF80h en memoria programa. Sin embargo estos vectores pueden ser remapeados para empezar en cualquiera de las páginas definidas en memoria programa después de haber sido reiniciado el DSP. Lo anterior se realiza cargando en los bits (15-7) del apuntador del vector de interrupción (IPTR) que se encuentra en el registro PMST con la dirección apropiada de la página en cuestión.

NOMBRE	DIRECCIÓN	DESCRIPCIÓN	TIPO
DRR20	20h	Registro 2 de Recepción de Datos McBSP0	McBSP #0
DRR10	21h	Registro 1 de Recepción de Datos McBSP0	McBSP #0
DXR20	22h	Registro 2 de Transmisión de Datos McBSP0	McBSP #0
DXR10	23h	Registro 1 de Transmisión de Datos McBSP0	McBSP #0
TIM	24h	Registro TIMER0	Timer0
PRD	25h	Contador de Periódo TIMER0	Timer0
TCR	26h	Registro de Control TIMER0	Timer0
–	27h	Reservado	
SWWSR	28h	Registro de Estados de Espera por software	Bus Externo
BSCR	29h	Registro de Control de Banco de Interrupción	Bus Externo
–	2Ah	Reservado	
SWCR	2Bh	Registro de Control de Estados de Espera por Software	Bus Externo
HPIC	2Ch	Registro de Control HPI	HPI
–	2Dh–2Fh	Reservado	
TIM1	30h	Registro TIMER1	Timer1
PRD1	31h	Contador de Periódo TIMER1	Timer1
TCR1	32h	Registro de Control TIMER1	Timer1
–	33h–37h	Reservado	
SPSA0	38h	Registro de Direcciones del subBanco McBSP0	McBSP #0
SPSD0	39h	Registro de Datos del subBanco McBSP0	McBSP #0
–	3Ah–3Bh	Reservado	
GPIOCR	3Ch	Registro de Control de pines de I/O de Propósito General	GPIO
GPIOSR	3Dh	Registro de Estado de pines de I/O de Propósito General	GPIO
–	3Eh–3Fh	Reservado	
DRR21	40h	Registro 2 de Recepción de Datos McBSP1	McBSP #1
DRR11	41h	Registro 1 de Recepción de Datos McBSP1	McBSP #1
DXR21	42h	Registro 2 de Transmisión de Datos McBSP1	McBSP #1
DXR11	43h	Registro 1 de Transmisión de Datos McBSP1	McBSP #1
–	44h–47h	Reservado	
SPSA1	48h	Registro de Direcciones del subBanco McBSP1	McBSP #1
SPSD1	49h	Registro de Datos del subBanco McBSP1	McBSP #1
–	4Ah–53h	Reservado	
DMPREC	54h	Registro de Control de habilitación y Prioridad de Canal DMA	DMA
DMSA	55h	Registro de Direcciones de subBanco DMA	DMA
DMSDI	56h	Registro de Datos de subBanco DMA con Autoincremento	DMA
DMSDN	57h	Registro de Datos de subBanco DMA	DMA
CLKMD	58h	Registro de Modo de Reloj	PLL
–	59h–5Fh	Reservado	

Figura A.5: Registros Mapeados en Memoria asociados a Periféricos.

El reset por hardware no podrá ser remapeado debido a que cuando éste se presente, se cargará con valor de "1" los bits del apuntador del IPTR dado que el vector asociado al reset siempre está en la localidad FF80h en memoria programa.

Memoria programa en Modo Extendido

El C5402 utiliza un esquema de memoria extendida paginada en memoria programa para permitir un direccionamiento de 1024k localidades de memoria. La memoria programa está organizada en 16 páginas de 64k-palabras en cada página y mediante un registro extra mapeado en memoria (XPC) se define la selección de página a emplear. La disposición de la memoria externa se muestra en la figura (A.6).

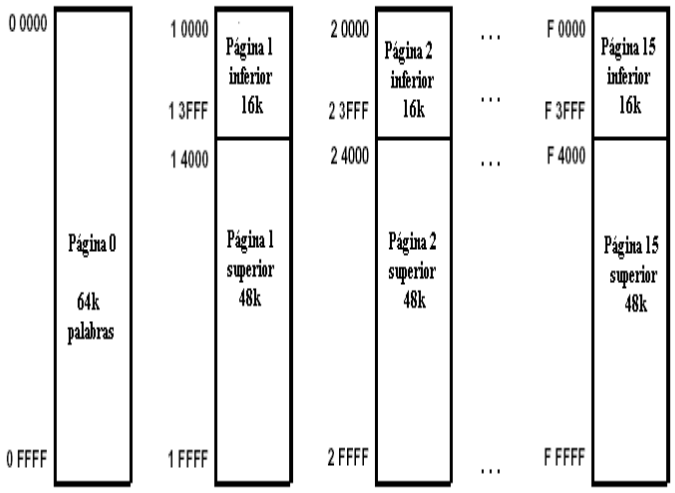

Figura A.6: Mapa de Memoria Programa en modo Extendido.

A.5. Unidad Central de Proceso (CPU)

El CPU está caracterizado por tener:

- Una unidad aritmética lógica (ALU) de 40 bits.
- Dos acumuladores de 40 bits.
- Un bloque de corrimiento de 40 bits.
- Un Multiplicador / Sumador de 17 x 17 bits.
- Una unidad de comparación, selección y almacenamiento (CSSU).
- Un registro temporal de 16 bits (T).
- Un registro de transición de 16 bits (TRN).
- Un codificador de exponente.

Registros de Estado y de Control del CPU.

El C5402 tiene 3 registros de estado y control:

- Registro de Estado 0 (ST0).
- Registro de Estado 1 (ST1).
- Registro de Estado de Modo de Procesador (PMST).

Los registros ST0 y ST1 nos indica el estado de los modos y condiciones de operación; el registro PMST contiene el estado de la configuración de memoria e información de control.

A.6. Interrupciones

El C5402 consta de un conjunto de interrupciones tanto internas como externas, además de incluir interrupciones por software y para los periféricos internos.
Las localidades de los vectores de interrupción y sus respectivas prioridades para todas las interrupciones internas y externas se muestran en la figura (A.7).
Las interrupciones descritas anteriormente se pueden configurar mediante el registro de banderas de interrupción (IFR) y el registro de interrupciones mascarables (IMR). En las figuras (A.8) y (A.9) se muestran la disposición de bits de los registros antes mencionados, como también, una breve descripción de sus respectivos campos de bits.

Sólo se hace mención del puerto serial bufureado multicanal como periférico interno empleado en el diseño del sistema de síntesis de voz en tiempo real.

Puertos Seriales Bufereados Multicanal.

El C5402 incluye de 2 puertos seriales bufereados multicanal (McBPs) que permiten realizar interfaces directas con dispositivos externos como DSP's de la familia C54x, codecs, y otros dispositivos. Los McBPs están basados en las interfaces de puertos serie estándar que se encuentran en los DSP's de la familia C54x. Los McBPs constan de las siguientes características:

- Comunicación Full-Duplex.
- Registros de datos de doble Buffer que permiten una comunicación de datos continua.
- "Framing" y "Clocking" independiente para la recepción y transmisión.

Además los McBPs tienen capacidades para:

- Interface directa con:
  - Framers T1/E1.
  - Interrupción compatible MVIP y dispositivos conforme a ST-BUS.
  - Dispositivos conforme a IOM-2
  - dispositivos de interfaz periférico serial.
- Transmisión Multicanal y recepción de 128 canales.
- Una amplia selección de tamaños de datos incluyendo 8,12,16,20,24 o 32 bits.
- "Compansión" ley μ y ley A.
- Polaridad programable para la sincronización de "frames" y reloj.
- Reloj interno programable y generación de "frames".

Los McBPs consisten de canales separados para la recepción y transmisión que operan de manera independiente. La interface externa de cada McPB consiste de los siguientes pines externos:

NOMBRE	LOCALIDAD DECIMAL	LOCALIDAD HEX	PRIORIDAD	FUNCIÓN
$\overline{RS}$, SINTR	0	00	1	Reset (Por Hardware y Sotfware)
$\overline{NMI}$, SINT16	4	04	2	Interrupción no enmascarable
SINT17	8	08	—	Interrupción por Software #17
SINT18	12	0C	—	Interrupción por Software #18
SINT19	16	10	—	Interrupción por Software #19
SINT20	20	14	—	Interrupción por Software #20
SINT21	24	18	—	Interrupción por Software #21
SINT22	28	1C	—	Interrupción por Software #22
SINT23	32	20	—	Interrupción por Software #23
SINT24	36	24	—	Interrupción por Software #24
SINT25	40	28	—	Interrupción por Software #25
SINT26	44	2C	—	Interrupción por Software #26
SINT27	48	30	—	Interrupción por Software #27
SINT28	52	34	—	Interrupción por Software #28
SINT29	56	38	—	Interrupción por Software #29
SINT30	60	3C	—	Interrupción por Software #30
$\overline{INT0}$, SINT0	64	40	3	Interrupción Externa de Usuario #0
$\overline{INT1}$, SINT1	68	44	4	Interrupción Externa de Usuario #1
$\overline{INT2}$, SINT2	72	48	5	Interrupción Externa de Usuario #2
TINT0, SINT3	76	4C	6	Interrupción Tmer0
BRINT0, SINT4	80	50	7	Interrupción de Recepción McBPs #0
BXINT0, SINT5	84	54	8	Interrupción de Transmisión McBps #0
Reservado (DMAC0), SINT6	88	58	9	Reservado (default) o Interrupción del canal DMA #0. La selección es hecha en el registro DMPREC
TINT1(DMAC1), SINT7	92	5C	10	Interrupción Timer1 (default) o Interrupción del canal DMA #1. La selección es hecha en el registro DMPREC
$\overline{INT3}$, SINT8	96	60	11	Interrupción Externa de Usuario #3
HPINT, SINT9	100	64	12	Interrupción HPI
BRINT1(DMAC2), SINT10	104	68	13	Interrupción de Recepción McBPs #1 (default) o Interrupción del canal DMA #2. La selección es hecha en el registro DMPREC
BXINT1(DMAC3), SINT11	108	6C	14	Interrupción de Transmisión McBPs #1 (default) o Interrupción del canal DMA #3. La selección es hecha en el registro DMPREC
DMAC4,SINT12	112	70	15	Interrupción del canal DMA #4
DMAC5,SINT13	116	74	16	Interrupción del canal DMA #5
Reservado	120–127	78–7F	—	Reservado

Figura A.7: Localidades de Interrupciones y sus prioridades.

15–14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	DMAC5	DMAC4	BXINT1 o DMAC3	BRINT1 o DMAC2	HPINT	INT3	TINT1 o DMAC1	RES o DMAC0	BXINT0	BRINT0	TINT0	INT2	INT1	INT0

Figura A.8: Registros de banderas de Interrupción (IFR) y de mascaras de Interrupción (IMR).

- BCLKX: Reloj de referencia para la transmisión.
- BDX: Transmisión de datos.
- BFSX: "Frame" de sincronía para la transmisión.
- BCLKR: Reloj de referencia para la recepción.

BIT		FUNCTION
NUMBER	NAME	
15–14	–	Reservado para futura expansión
13	DMAC5	Bit de bandera/enmascaramiento de Interrupción del canal DMA #5
12	DMAC4	Bit de bandera/enmascaramiento de Interrupción del canal DMA #4
11	BXINT1/DMAC3	Bit de bandera/enmascaramiento de Interrupción de Transmisión MCBP1 o de Interrupción del canal DMA #3. La selección es hecha en el registro DMPREC
10	BRINT1/DMAC2	Bit de bandera/enmascaramiento de Interrupción de Recepción MCBP1 o de Interrupción del canal DMA #2. La selección es hecha en el registro DMPREC
9	HPINT	Bit de bandera/enmascaramiento de interrupción Huésped a 54x
8	INT3	Bit de bandera/enmascaramiento de interrupción externa #3
7	TINT1/DMAC1	Bit de bandera/enmascaramiento de Interrupción de Timer1 o de Interrupción del canal DMA #1 La selección es hecha en el registro DMPREC
6	DMAC0	El Bit puede configurarse como Reservado o como Bit de bandera/enmascaramiento de Interrupción del canal DMA #0. La selección es hecha en el registro DMPREC
5	BXINT0	Bit de bandera/enmascaramiento de interrupción de transmisión McBP0
4	BRINT0	Bit de bandera/enmascaramiento de interrupción de recepción McBP0
3	TINT0	Bit de bandera/enmascaramiento de interrupción Timer0
2	INT2	Bit de bandera/enmascaramiento de interrupción externa #2
1	INT1	Bit de bandera/enmascaramiento de interrupción externa #1
0	INT0	Bit de bandera/enmascaramiento de interrupción externa #0

Figura A.9: Campos de Bits de los registros IFR y IMR.

- BDR: Recepción de datos.

- BFSR: "Frame" de sincronía para la recepción.

En el transmisor, el "frame" de sincronía de transmisión y el reloj para la transmisión son señalizados por los pines BFSX y BCLKX, respectivamente. El CPU o los canales de DMA pueden iniciar la transmisión de datos escribiendo al registro de transmisión de datos (DXR). Los datos escritos en el registro DXR son corridos hacia el pin BDX a través del registro de corrimiento de transmisión (XSR). Esta estructura permite al DXR recargarse con la siguiente palabra a ser transmitida mientras la palabra presente está en progreso.
En el receptor, el "frame" de sincronía de recepción y el reloj para la recepción son señalizados por los pines BFSR y BCLKR, respectivamente.

EL CPU o los canales de DMA pueden leer el dato recibido desde el registro de recepción de datos (DRR). Los datos recibidos en el pin BDR son corridos en el registro de corrimiento de recepción (RSR) para después ser guardados en el registro Buffer de recepción (RBR). Si el registro DRR está vacío, el contenido del RBR es copiado dentro del DRR. La operación contraria, el RBR contiene el dato hasta que el DRR esté disponible. Esta estructura permite el almacenamiento de dos palabras previas mientras la recepción de una nueva palabra está en progreso. El diagrama de Bloques del McBP se muestra en la figura (A.10).

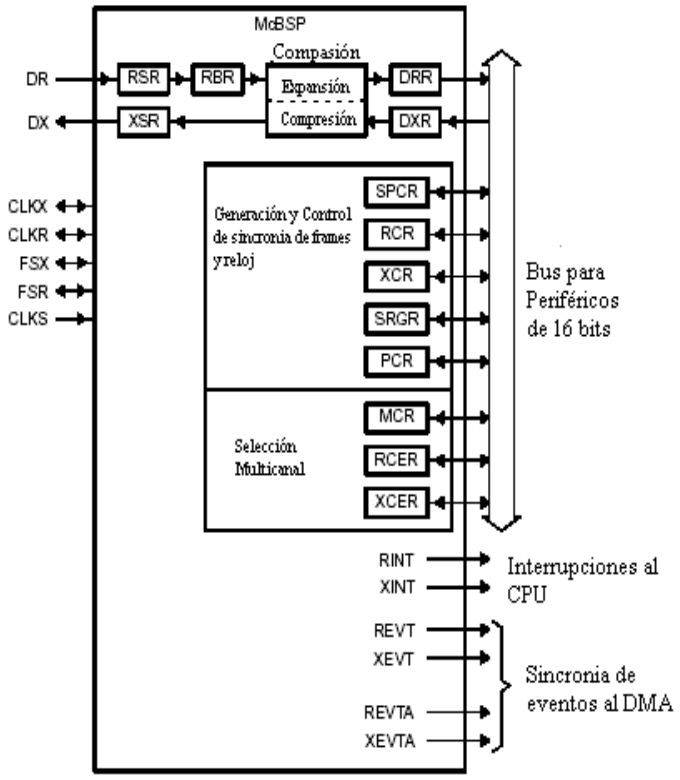

Figura A.10: Diagrama de Bloques del Puerto Serial Bufereado.

A.7. Convertidor A/D y D/A TLC320AD50

Para implantación del sistema de síntesis de voz propuesto en la presente tesis es necesario comprender el funcionamiento de la herramienta que nos permiten efectuar las conversiones analógicas - digitales y digitales - analógicas. A continuación se presenta una descripción de ésta herramienta, mejor conocida como *Codec*. El término *Codec* se deriva de la contracción de las palabras provenientes del inglés "*Co*der" y "*Dec*oder" (Codificador y Decodificador).

Interfaz Micrófono-Altavoz

Esta interfaz está integrada por dos conectores de 3.5 mm, de los cuales uno es usado para conectarse al micrófono (señal de entrada) y el otro para el altavoz (señal de salida). El conector de entrada tiene una ganancia fija de 10 dB y una etapa de conversión que permite que la señal pase de un estado *"single-ended"* (o de una sola terminación) a un estado diferencial, antes de ser digitalizada por el TLC320AD50C. El codec del DSK maneja un voltaje *"Bias"* para ser usado en aplicaciones que utilicen baterías o micrófonos tipo *"electret"*. También efectúa un filtrado pasivo entre los conectores y el codec. La entrada del micrófono está diseñada para micrófonos tipo *"electret"* que requieren para su funcionamiento un voltaje de *"Bias"*. Los micrófonos de tipo dinámico pueden ser conectados a este dispositivo siempre y cuando el voltaje de *"Bias"* sea suprimido mediante el uso de uno o varios capacitores. El nivel máximo de voltaje de la señal de entrada

que el codec puede tolerar es de 500 mV (350 mV_{RMS}); pero existe una amplificación a la entrada del *DSK* de 10 dB. El canal de salida puede ser configurado de tal modo que la señal de salida tenga una ganancia, en el intervalo desde los +0 hasta los -12 dB con incrementos de 6 dB. Ésta ganancia de salida es controlada desde el *CCS*; tanto para la señal de salida como para la señal de entrada.
El codec, adicionalmente, cuenta con una característica llamada *hardware strap* que permite accesar a la salida directa del AD50 y proporcionar la corriente necesaria para el funcionamiento de una carga de 600 ohms o de una etapa de potencia para conectar altavoces de baja impedancia (8 ohms, por ejemplo).
El TLC320AD50 es un circuito analógico (AIC) que funciona como interfaz entre las señales de entrada y de salida al *DSK* con las suficientes características que lo hacen adecuado para aplicaciones en tiempo real implantadas en procesadores digitales de señales.
Las características más importantes, en cuanto a su operación y a su construcción, se mencionan a continuación:

- Fuente de poder que suministra , ya sean 5 V únicamente, o fuente dual que suministra 5 V de manera analógica y 3 V de manera digital.
- Posee una resolución de 16 bits para la conversión analógica a digital (ADC, por sus siglas en inglés) y digital a analógica (DAC, por sus siglas en inglés)
- El valor de la reducción señal a ruido es de 85 dB (min).
- Contiene un filtrado inherente *"antialiasing"* y de compensación
- Posee una alta impedancia de entrada.
- Soporta hasta 4 dispositivos de manera serial.
- Modo de bajo consumo de potencia (7.5 mW).
- Disipación de potencia de 120mW en el modo de operación típica.
- Modo de operación de baja potencia con un valor máximo de 175 mW.
- Tasa de conversión configurable (muestreo) máxima de 22.05 kHz .
- Modo de apagado (20 mW).
- Circuito de interfaz analógica de propósito general para módems que usen el estándar V.43+ y superiores; y para aplicaciones para audio.
- Conversión analógica a digital (ADC) y digital a analógica (DAC) sigma delta de 16 bits con sobremuestreo.
- Interfaz para puerto serial
- Relación de señal a ruido con un valor típico de 89-dB SNR (signal-to-noise ratio) tanto del proceso de conversión analógico a digital (ADC) como de digital a analógico (DAC).
- Valor de distorsión total armónica con valor típico de 90-dB THD (signal to total harmonic distortion) tanto en el proceso de conversión analógico a digital (ADC) como en el de digital a analógico (DAC).

- Rango dinámico con valor de 88 dB.
- Modo de Auto-prueba que incluye una prueba de bucle digital y bucle analógico.
- Control de ganancia de salida o de entrada programable y configurable.
- Voltaje de referencia interna.
- Arquitectura Diferencial.
- Al TLC320AD50C es posible agregarle un sólo circuito esclavo.
- Formato de datos del tipo *complemento a dos.*
- Monitoreo de datos.
- Amplificación del monitoreo de los datos de entrada.
- Circuito de malla de fase cerrada (On-chip phase locked loop (PLL)).

El convertidor A/D TLC320AD50C proporciona una alta resolución en cuanto a conversión de señales, tanto desde el dominio analógico al dominio digital y viceversa, empleando la tecnología sigma-delta con sobremuestreo. Este dispositivo consiste en un par de rutas de comunicación, que son seriales, síncronas y de un tamaño de 16 bits, (una para cada dirección del flujo de datos). Incluye un filtro de intepolación antes de que la señal entre al convertidor digital - analógico (DAC) y un filtro de decimación posterior al convertidor analógico - digital (ADC). Este chip tiene la ventaja de que su *temporizador* puede ser configurado de tal modo que la tasa de muestreo, el retraso en la señal de sincronía y las funciones en aplicaciones relacionadas al control de procesos (filtro de ganancia programable, PLL, protocolos de comunicación) no se vean afectadas. La arquitectura sigma-delta produce una conversion analógica a digital (ADC) y digital a analógica (DAC) con una alta resolución a un bajo costo.
Las funciones programables de este dispositivo, pueden ser seleccionadas a través de su interfaz serial. De entre las opciones que ofrece, se incluyen las de *Reset*, apagado (*power down*), protocolos de comunicación, tasa de muestreo de señales, control de ganancias de entrada y de salida, y modos de sistemas de pruebas internas.
Para una operación óptima es recomendable que la temperatura del A/D TLC320AD50C se encuentre dentro del intervalo de 0 ^oC a 70 ^oC.

Los registros del chip A/D TLC320AD50C.

Existen siete registros de control que se describen a continuación:

- Registro 0 : Registro de no Operación (The No-Op register) Accesando este registro se permite la comunicación secundaria hacia otro dispositivo sin notificar ni alterar los demás registros.
- Registro 1: Registro de Control (Control register 1), los datos en este registro controlan:
  - "Reset" por medio de Software
  - Apagado por medio de Software

- Habilitación de las entradas normales o auxiliares
- Monitoreo de las entradas normales o auxiliares
- Selección del monitor del amplificador de ganancia de salida.
- Selección del bucle digital
- Selección del modo de operación del DAC ya sea en 16 bits o en (15+1) bits.

▪ Registro 2: Registro de Control (Control register 2), los datos en éste registro:

- Contienen el valor de salida de la bandera o *"FLAG"*
- Seleccionan el modo de operación telefónica.
- Contienen la bandera de salida (*output flag*) que indican el desborde del filtro FIR de decimación.
- Seleccionan el modo de operación del ADC ya sea en 16 bits o en (15+1) bits.
- Habilitan el bucle analógico.

▪ Registro 3: Registro de Control (Control register 3), los datos en éste registro:

- Establecen el número de SCLK retrasos entre FS y FSD
- Informan al dispositivo maestro cuántos elementos esclavos están conectados.

▪ Registro 4: Registro de Control (Control register 4), los datos en este registro:

- Establecen el valor de la ganancia tanto de salida como de entrada del amplificador de ganancia.
- Establecen la tasa de muestreo dependiendo del valor seleccionado de N, el cual es desde $N = 1$ hasta $N = 8$, en donde por medio de las ecuaciones (A.1) y (A.2) se pueden calcular la tasa de muestreo:

$$f_s = \frac{MCLK}{128N} \tag{A.1}$$

$$f_s = \frac{MCLK}{512N} \tag{A.2}$$

  Donde MCLK es el reloj maestro del DSP.

- Selecciona la malla de fase enlazada (PLL). Si el PLL es seleccionado, entonces la tasa de muestreo se establece por medio de la ecuación (A.3):

$$f_s = \frac{MCLK}{128N} \tag{A.3}$$

  Si el PLL es puenteado, la tasa de muestreo se establece por la ecuación (A.4):

$$f_s = \frac{MCLK}{512N} \tag{A.4}$$

  .

Los Registros 5 y 6 están reservados para pruebas de fabricación por lo que es recomendable abstenerse de escribir cualquier valor a estos dos registros.

Funciones del codec

El codec optimiza las funciones necesarias para los canales del convertidor analógico a digital y digital a analógico, el filtrado paso bajas, el control de las ganancias analógicas de entrada y salida, el sobremuestreo interno (realizado por medio de un filtro de decimacion e interpolación), y dos interfaces de puerto serial, de 16-bits. Los puertos seriales son independientes entre sí y son capaces de trabajar con diferentes frecuencias de muestreo. La máxima frecuencia de muestreo para ambos canales, es de 22.05 kHz.

Funciones híbridas

El circuito híbrido en el canal de datos, incluye amplificadores integrados cuya ganancia y polos en la frecuencia del filtro pasa-bajas, se establecen por medio de resistencias y capacitores externos. Esto permite una gran flexibilidad para ajustarse tanto a diversas tarjetas y como a diferentes estándares internacionales, proporcionando integración de funciones. Los estados de amplificación para el filtro, en el canal de datos, van seguidos de una ganancia amplificada programable que alimenta a los controladores del canal de salida.

Frecuencias de operación y filtro de control

La frecuencia de muestro es controlada por el registro número 4. cuando se habilita la malla de lazo cerrado (D7=0). La frecuencia de muestreo se establece por la ecuación (A.5):

$$f_s = \frac{MCLK}{128N} \tag{A.5}$$

en donde N es un número natural entre 1 y 8 y MCLK es el valor del reloj maestro (master clock). Cuando la malla interna de lazo cerrado (PLL) se deshabilita (D7=1), la frecuencia de muestreo se establece por la ecuación (A.6):

$$f_s = \frac{MCLK}{512N} \tag{A.6}$$

Este registro divide el reloj maestro del sistema, hasta obtener la frecuencia necesaria para alimentar el codec. Si la frecuencia de muestreo es inferior a 7 kHz, la frecuencia de muestreo se deriva del reloj principal (MCLK) empleando la ecuación número dos, por lo que el PLL interno debe de ser puenteado. La frecuencia de corte del filtro de control (paso banda) no puede ser controlada mediante la programación de los registros. La respuesta del filtro se muestra en la siguiente figura para una frecuencia de muestreo de 8 kHz. Este filtro paso bandas se adecúa de manera lineal con la frecuencia de muestreo del canal de conversión analógica a digital (ADC). La señal de entrada es amplificada y se redirecciona a la entrada del ADC. El ADC convierte la señal en una señal discreta en palabras digitales en el formato de datos complemento a dos, correspondiendo al valor instantáneo de la señal analógica de a cuerdo al período de muestreo. Estas palabras de 16 bits (o 15 bits + 1), representan los valores muestreados de la señal analógica de entrada después del registro PGA, que se disponen fuera del puerto serial (DOUT) en el borde positivo del SCLK durante el intervalo de sincronía. La programación del tamaño de las palabras ya sean de 16-bits o de (15 + 1) del ADC se efectúa usando el registro de control 2. El valor de fábrica es de (15 + 1) bits.

DAC Signal Channel

El registro de datos de entrada ("data in", DIN) recibe la palabra de datos de 16-bit (complemento a dos) de manera serial del *"host"* (DSP y demás periféricos) durante el primer intervalo de las comunicaciones. Estas palabras digitales, representan la señal de salida analógica antes del registro PGA, y se registran en el puerto serial del DIN en el borde de caída del SCLK durante el intervalo de sincronía. Los datos son convertidos en un tren de pulsos por el DAC por medio de la tecnología sigma-delta, lo cual consiste en un filtro digital de interpolación y de un modulador digital. La salida del modulador es direccionada a un filtro paso-bajas interno para completar la reconstrucción de la señal analógica. Finalmente, la señal analógica se introduce al amplificador de ganancia programable. Este amplificador es capaz de conducir una carga 600W de manera difrencial a través de los registros OUTP y OUTM.

Sigma-Delta ADC

El convertidor analógico - digital del tipo sigma - delta es un modulador sigma-delta con un sobremuestreo a razón de 64 veces. El ADC tiene un funcionamiento de alta resolución, de poco ruido debido a las técnicas de sobremuestreo que emplea. Por ello simplemente se requiere de filtros "antialiasing" de un sólo - polo en las entradas analógicas.

Filtro de decimación

Los filtros de decimación reducen la tasa del flujo de datos digitales a la tasa de muestreo. Esto se logra por medio de la decimación con una relación de 1:64. La salida del filtro de decimación es una palabra de datos de 16-bits con formato de complemento a dos de acuerdo a la tasa de muestreo seleccionada para un canal de datos en particular. El ancho de banda del filtro es 0.439 veces la frecuencia de muestreo y se adecúa por si sólo de manera lineal de acuerdo a la tasa de muestreo.

Sigma-Delta DAC

El convertidor digital - analógico del tipo sigma - delta es un modulador del sigma - delta con un sobremuestreo a razón de 256 veces. El DAC proporciona funcionamiento de alta resolución y poco ruido debido a las técnicas de sobremuestreo que emplea.

Filtro de Interpolación

El filtro de interpolación realiza un nuevo muestreo de la señal digital con una velocidad de 256 veces la tasa de muestreo. La velocidad de la salida de los datos del filtro de interpolación es muy alta, por lo que es necesario aplicarle la técnica sigma-delta a la entrada del DAC. El ancho de banda de este filtro es de 0.439 veces la frecuencia de muestro y se adecúa por sí sólo de manera lineal de acuerdo a la tasa de muestreo.

Los bucles (loopback) digitales y analógicos

Los bucles análogos y digitales proporcionan varios medios para probar los canales de los datos ADC/DAC del módem y se pueden utilizar para las pruebas a nivel sistema del circuito. El

bucle analógico direcciona la salida del filtro paso-bajas del DAC a la entrada analógica donde es convertido por el ADC en una palabra digital. El bucle digital, el cual se habilita fijando el valor del bit D1 en el registro de control 1 a "1", direcciona la salida del ADC a la entrada del DAC. El bucle análogo se habilita al fijar el valor del bit D3 en el registro del control 2 a "1".

Bandera de desborde del filtro tipo FIR

El filtro de decimación es un filtro de tipo FIR al cual se le asocia una bandera de desbordamiento (bit D5) del registro de control 2 para indicar que la señal analógica de la entrada ha excedido los límites de los cálculos internos del filtro de decimación. Una vez que la bandera de desborde se levanta en el registro, permanece en este estado hasta que el usuario lee este registro. La lectura de este valor reajusta la bandera de desborde a su estado original. Si ocurre un desbordamiento en el filtro FIR, la señal de entrada se debe atenuar por medio del PGA o por cualquier otro método.

Apéndice B

Interfaz Code Composer studio (CCS)

La interfaz gráfica Code Composer Studio (CCS) permite a los programadores agilizar y optimizar los procesos de desarrollo e implantaciones de aplicaciones inscrustadas para el procesamiento digital de señales en tiempo real. EL CCS extiende sus capacidades en un entorno de desarrollo al incluir herramientas para el análisis en tiempo real en conjunción con una tarjeta de desarrollo para un DSP (DSK).

En este apéndice se dará una breve explicación de la configuración, manejo y de las herramientas que constituyen al ambiente de desarrollo CCS. Para mayores referencias en cuanto a lo relacionado con el manejo del ambiente Code Composer Studio, se recomienda la consulta de [33], [34], [35], [36], [37].

B.1. Generalidades del CCS

El CCS es una aplicación de Texas Instruments que permite a los usuarios programar, editar, depurar y analizar programas del procesamiento digital de señales. El ambiente de desarrollo CCS soporta diversos DSP's de Texas Instruments para la creación de proyectos, donde el CCS consta de un gestor de proyectos que permite manejar tareas de programación. Para propósitos de depuración consta de puntos de interrupción (breakpoints), observadores de variables (watches), ventanas de memorias, registros y pila, puntos de prueba para el flujo de datos hacia y desde la tarjeta de evaluación (DSK), análisis gráfico, ejecución personalizada y capacidad para la programación y visualización de código mixto en ensamblador y lenguaje C. Una opción muy importante y útil es poder manejar grandes proyectos desde un ambiente gráfico para el usuario. Las características más importantes del CCS se resumen en las siguientes secciones:

- Entorno integrado de desarrollo con editor, depurador, gestor de proyectos, generador de perfiles de programación, puntos de interrupción, observadores de variables, puntos de prueba, ventanas de registros/memoria/pila.
- Herramientas de generación de código: Compilador de C, optimizador de ensamblador y ligador de archivos y código fuente.
- Sistema operativo de tiempo real (DSP/BIOS)
- Intercambio de datos en tiempo real entre la terminal y el DSP (RTDX, Real-Time Data eXchange).

- Herramientas de análisis y visualización de datos en tiempo real.

B.2. El Ambiente Integrado de Desarrollo CCS

El ambiente integrado de desarrollo CCS está diseñado para permitir la edición, creación y compilación de programas en la tarjeta de evaluación del DSP.

Características de Edición del Código Programa.

El CCS permite editar código en lenguaje ensamblador y C, además de poder observar el código en lenguaje C seguido de sus correspondientes instrucciones en lenguaje ensamblador.

Características de Creación de Aplicaciones

Usando el CCS, se crean aplicaciones mediante la adición de archivos en un proyecto. Los archivos incluidos en el proyecto pueden ser código en lenguaje C y en lenguaje ensamblador, archivos objeto, librerías y archivos de ligado de comandos. Se tiene la opción de compilar individualmente los respectivos archivos o realizar una compilación general del proyecto.

Características de Depuración de Aplicaciones

Para la depuración de aplicaciones, el CCS incluye ventanas para la visualización de los registros, memoria, observadores variables, puntos de prueba, gestores de compilación y gráficas de señales.

B.3. Herramientas de Generación de Código

Las herramientas de generación de código permiten la configuración de los programas mediante un proceso de ensamblado, compilación, ligado y referencias de librerías a usar. Dicho proceso puede realizarse de manera transparente al usuario gracias a las opciones de compilación del proyecto, o bien, efectuarlo de modo manual.

A continuación se da una breve descripción de las herramientas de generación de código:

- El compilador de lenguaje C, acepta código fuente en C produciendo código fuente en ensamblador.
- El ensamblador, traslada archivos o código fuente escritos en lenguaje ensamblador a archivos objeto de lenguaje máquina, donde el lenguaje máquina está basado en formatos de archivos de objetos comunes (COFF).
- El ligador, combina archivos objetos en un módulo único de objetos ejecutables. Al crear el módulo ejecutable realiza una relocalización de símbolos e inserta las referencias externas al proyecto.
- El ligador de archivos, permite conjuntar los diferentes archivos del proyecto en un único archivo en modo de librería.

- La utilidad de traducción de mnemónicos de ensamblador a modo algebraico, convierte el código escrito en lenguaje ensamblador a un código descrito de forma algebraica (instrucciones algebraicas).
- Podemos utilizar la utilidad de generación de librerías para la creación de librerías propias del usuario.
- La utilidad de conversión a sistema hexadecimal, convierte archivos del tipo COFF a archivos con formato objeto de TI, ASCII-HEX, Intel, Motorola y Tektronic. Por tanto, se puede cargar dichos archivos a un programador de memoria EPROM.
- El listado de referencia cruzada usa los archivos objetos para la referencia cruzada de símbolos, sus definiciones, y sus referencias en los archivos fuente ligados.
- El listado absoluto acepta archivos objeto ligados como archivos de entrada y crea archivos con extensión ".abs" como archivos de salida, de tal manera que al ensamblar los archivos con extensión ".abs" se produce una lista que contiene las direcciones absolutas.

B.4. Desarrollo de programas mediante el CCS

Para realizar un programa en el CCS lo habitual es la creación de un proyecto en el cual se tendrá que realizar los siguientes pasos para su ejecución:

- Creación del Proyecto.
- Creación de los códigos fuente.
- Ligar el código fuente con un fichero de ligado de comandos (con extensión ".cmd") el cual definirá los espacios de memoria permisibles de la tarjeta de Evaluación (DSK).
- Especificar la localización de los ficheros "include" y librerías a utilizar.
- Y por último, Compilar el proyecto y cargarlo al DSK.

Para la creación del proyecto se escoge la opción "Project" de la barra de herramientas para después seleccionar el tópico "New", de tal manera que se abrirá un cuadro de dialogo donde se nombrará al proyecto con extensión ".mak". De igual manera para la creación y edición de los códigos fuente se escoge la opción "File" de la barra de herramientas con la selección de "New" donde se nombrará al código con la extensión debida (".asm", ".C", ".cmd").

Es muy importante definir bien el fichero de ligado de comandos (".cmd") dado que en él se encuentran los segmentos de memoria a usar. Por ejemplo:

```
/Especificación de los segmentos de memoria  Memoria.cmd

MEMORY{ PAGE 1:           /página de memoria dato

STACK: origin = 0x1180, length = 0x0560 /memoria destinada a la
                                        /pila
```

```
 DATA : origin = 0x1f00, length = 0x4000 /memoria definida
                                         /para datos

PAGE 0:  /página de memoria programa
  PROG    : origin = 0x0080, length = 0x3f70  /memoria definida
                                              /para código fuente
  VECS    : origin = 0xff80, length = 0x007f  /vectores de
                                              /interrupción
   }

   SECTIONS
   {
     vectors > VECS    PAGE 0  /sección de vectores de
                               /interrupción
     .text   > PROG    PAGE 0  /sección del código
                               /fuente
     .const  > DATA    PAGE 1  /asignación para la memoria
                               /a destinada a constantes
     .data   > DATA    PAGE 1  /sección de memoria dato
     .stack  > STACK   PAGE 1  /sección para la memoria de la pila
   }
```

B.5. Intercambio de Datos en Tiempo Real (RTDX).

De entre las muchas funcionalidades de programación con las que cuenta el Code Composer Studio, existe una que nos permite implantar, ejecutar y monitorizar la actividad de las aplicaciones diseñadas en el DSK TMS320C5402. A ésta herramienta se le conoce como Intercambio de Datos en Tiempo Real, o bien Real-Time Data Exchange (RTDX, por sus siglas en inglés).

El RTDX proporciona una visualización continua y en tiempo real de las aplicaciones ejecutándose en el DSP y el desenvolvimiento de éstas aplicaciones en la realidad.
De igual modo, el RTDX, permite la transferencia de datos entre una computadora (conocida como *"Host"*) y el Starter KIT del DSP TMS320C5402 anexado a dicha computadora; sin interferir en el desempeño del DSP ni en el de la aplicación. Los datos transferidos, pueden ser analizados y visualizados en la computadora empleando cualquier cliente de aplicación tipo COMM de la compañía Microsoft.

El RTDX consiste en una librería que se aloja en memoria del DSP. Las aplicaciones que se ejecutan en el DSP, llaman a las funciones que se encuentran en esta librería para poder enviar o recibir datos desde la computadora que hospeda al DSK, mediante el uso de un emulador de la interfaz JTAG.

En la plataforma en que se hospeda el DSK (o computadora anfitriona) existe una librería del RTDX conocida como librería anfitriona del RTDX (RTDX Host Library), que opera en conjunto con el Code Composer Studio y con una Interfaz de Aplicación de Programa (API, por sus siglas en inglés) tipo COMM, las cuales permiten a las herramientas de despliegue de gráficas e información

establecer una comunicación directa con las librerías del RTDX, para así obtener o enviar datos al DSK.
El uso de esta Interfaz API ofrece la posibilidad de desarrollar cualquier aplicación en otras plataformas, aparte del Code Composer, tales como:

- LabVIEW de la compañía National Instruments.
- Herramientas gráficas en tiempo real de la compañía Quinn - Curtis.
- Microsoft Excel.
- Lenguajes de programación como Visual Basic o Visual C++.
- En el caso en que se desee la transferencia de datos desde el DSP a un modem, será necesario establecer una aplicación empleado los lenguajes descritos anteriormente en conjunción con el RTDX. Para ello, es necesario almacenar los datos en localidades temporales de memoria de la terminal (PC).

El flujo de datos del RTDX.

El Code Composer Studio de encarga de controlar el flujo de datos entre la plataforma anfitriona (computadora personal) y el objetivo (DSP o DSK). EL RTDX forma una estructura tipo pipeline (tubería) entre la PC y el DSK. Los datos pueden ser enviados desde el DSK a la aplicación COM y viceversa. Esta Tubería se puede considerar como un conjunto de varias tuberías mas delgadas, que son conocidas como canales de transmisión de datos o simplemente canales. Los canales, al ser unidireccionales, permiten diferenciar los distintos tipos de datos, asociados a un canal.
La aplicación incrustada en el DSP envía los datos al "host" llamando las funciones que se encuentran en la librería del RTDX para el Target o DSK (RTDX Target Library). Estas funciones inmediatamente envían los datos a un almacén de datos intermediario (buffer) y posteriormente atienden las llamadas de la aplicación. La "RTDX Target Library" transmite los datos almacenados en el buffer hacia el "host" por medio de la interfaz JTAG de modo que no se interfiera con la aplicación que se ejecute en el DSK.

El "host" almacena los datos recibidos ya sea en un buffer de memoria o en un archivo perteneciente al mismo RTDX conocido como archivo bitácora (log file). Los datos almacenados pueden ser recuperados por medio de cualquier cliente COMM.
El proceso de enviar datos a la aplicación que se haya en el DSP se hace por medio del cliente COMM. Los datos que se vayan a enviar al DSK, son escritos a un buffer de memoria, que se haya en la "RTDX Host Library". Cuando la "RTDX Host Library" recibe una solicitud de envío de datos desde el DSK, los datos son enviados por medio de la interfaz JTAG y son escritos en la localidad que ha sido especificada en el DSK, sin intervenir en la ejecución de la aplicación. El "host" notifica a la RTDX Target Library cuando ha terminado de enviar los datos solicitados.

Flujo de datos desde el DSK a la computadora "host".

Para almacenar datos en el "host" provenientes del DSK, es necesario declarar un canal de salida en el DSP, y escribir en el utilizando las rutinas definidas en la Interfaz de Usuario (pegar

figura). Estos datos son almacenados en un buffer de memoria dentro del DSK definido en el RTDX Target Library y posteriormente, son enviados al "host" por medio de la interfaz de emulación JTAG. El RTDX Host Library recibe los datos desde la interfaz JTAG, y los almacena, ya sea en un buffer de memoria en el "host" o en el archivo log del RTDX o en la ubicación que se especifique al configurar el funcionamiento del RTDX en el Code Compose Studio.

Finalmente, estos datos pueden ser recuperados, por medio de cualquier aplicación tipo COMM como por ejemplo:

- Aplicaciones en Visual Basic.
- Aplicaciones en Visual C++.
- Lab View.
- Microsoft Excel.

Flujo de datos desde la computadora "host" al DSK.

Para que la tarjeta del DSK pueda recibir datos desde la computadora "host", es necesario declarar un canal de entrada de datos, y solicitar los datos via el RTDX . Existe un buffer que guarda los datos llamado RTDX target buffer, que se encarga de almacenar los datos enviar por la interfaz JTAG.

Por medio de un cliente COMM es posible enviar los datos, utilizando el siguiente método. Los datos a enviar se almacenan en un buffer de memoria dentro de la RTDX Host Library, la cual al recibir una solicitud de envío de datos hecha por el DSK, envía los datos a través de la interfaz JTAG. Los datos son escritos en tiempo real en la localidad especificada en el DSK. La computadora "host" indica a la RTDX Target Library cuando el envió de datos ha finalizado.

Bibliografía

[1] Cole R. A. *A Survey of the State of the Art in Human Language Technology.* Center for Speaking Language Understanding, EUA, 1996.

[2] Concheiro A. Kuhlmann, F. *Información y Telecomunicaciones.* Fondo de Cultura Económica, México, 1997.

[3] Kleijn W. B. & Paliwal K. K., editor. *Speech Coding and Synthesis.* Elsevier Science, 1995.

[4] Papamichalis E. Panos. *Practical Approaches to Speech Coding.* Processor Series. Prentice - Hall, EUA, 1987.

[5] Proakis John & Hansen John Deller John R. *Discrete-Time Processing of Speech Signals.* MacMillan, EUA, 1993.

[6] Parsons Thomas. *Voice and Speech Processing.* Mc Graw Hill Inc., EUA, 1987.

[7] Owens F. J. *Signal Processing of Speech.* Mc Graw Hill Inc., EUA, 1993.

[8] Rabiner Lawrence & Schafer Ronald W. *Digital Processing of Speech Signals.* Prentice-Hall Inc, EUA, 1978.

[9] Flanagan J. L. *Speech Analysis, Synthesis and Perception.* Springer-Verlag, EUA, 1972.

[10] Makhoul J. Linear prediction: A tutorial review. *Proc. IEEE*, 63:561–580, Abril 1975.

[11] J. D. & Gray A. H. Markel. *Linear Prediction of Speech.* Springer - Verlag, EUA, 1976.

[12] Hayes H. Monson. *Statistical Digital Signal Processing and Modeling.* John Wiley & Sons Inc., EUA, 1996.

[13] Proakis John & Manolakis Dimitris. *Tratamiento Digital de Señales: Principios, algoritmos y aplicaciones.* Prentice Hall, 1998.

[14] Furui Sadaoki. *Digital Speech Processing, Synthesis, and Recognition.* Marcel Dekker, EUA, 2001.

[15] Texas Instruments. *TMS320C54x DSP Reference Set, Volume 1: CPU and Peripherals.* literature number SPRU131.

[16] Texas Instruments. *TMS320C54x DSP Reference Set, Volume 5: Enhanced Peripherals.* literature number SPRU302.

[17] Escobar S. L. & Alcantara S. R. Fixed point arithmetic using digital signal processors. pages 22–25, Acapulco, México, Mayo 2000. International Conference on Telecommunications (ICT2000).

[18] Instituto Ñacional de Estadística Geografía e Informática. www.inegi.gob.mx.

[19] Privacy Rights Clearinghouse. www.privacyrights.org.

[20] Communication Security Inc. www.bugsweep.com.

[21] Charles L & Udovich Richard J Taylot. *Telephone eavesdropping and detection.* Sams Publishing, EUA, 1998.

[22] Bigelow Stephen J. *Understanding Telephone Electronics.* Sams Publishing, EUA, 1994.

[23] Michael A. Noll. *Introduction To telephones and telephone systems.* Artech House Inc., EUA, 1991.

[24] M. Harb. *Modern Telephony.* Prentice - Hall Inc., EUA, 1989.

[25] Winston A. Gayler. *Telephone Voice Transmission. Standars and measurements.* Prentice - Hall Inc., EUA, 1989.

[26] Sousa F. & Felix P. *Developing Secure Communication Systems Using the TMS320C50 DSP.* Texas Instruments Incorporated, EUA, 1997.

[27] Escobar Salguero Larry. *Laboratorio de DSP´s. Familias TMS320C5x y TMS320C54x.* Facultad de Ingeniería UNAM, México, 2002.

[28] Texas Instruments. *TMS320C54x DSP Reference Set, Volume 2: Mnemonic Instruction Set.* literature number SPRU172.

[29] Texas Instruments. *TMS320C54x DSP Reference Set, Volume 3: Algebraic Instruction Set.* literature number SPRU179.

[30] Texas Instruments. *TMS320C54x DSP Reference Set, Volume 4: Applications Guide.* literature number SPRU173.

[31] Texas Instruments. *TMS320C54x Optimizing C Compiler User's Guide.* literature number SPRU103.

[32] Texas Instruments. *TMS320C54x C Source Debugger User's Guide.* literature number SPRU099.

[33] Texas Instruments. *Code Composer User's Guide.* literature number SPRU296.

[34] Texas Instruments. *Code Composer Studio User's Guide.* literature number SPRU328.

[35] Texas Instruments. *TMS320C54x DSP/BIOS User's Guide.* literature number SPRU326b.

[36] Texas Instruments. *TMS320C54x RTDX Help.* literature number SPRH053.

[37] Texas Instruments. *TMS320C54x Real-Time Data Exchange (RTDX) Tutorial RTDX Help.* literature number SPRH108.