TESIS
IT 200
B5
v.1

UNIVERSIDAD CATÓLICA ANDRÉS BELLO
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA DE TELECOMUNICACIONES

DISEÑO DE UN TELÉFONO IP

TRABAJO ESPECIAL DE GRADO

presentado ante la

UNIVERSIDAD CATÓLICA ANDRÉS BELLO

como parte de los requisitos para optar al título de

INGENIERO EN TELECOMUNICACIONES

REALIZADO POR *Blanco Ascencion, Luis Fernando*
Suárez Rodríguez, Victor Eduardo

PROFESOR GUIA *Alvarado Robles, Luis*

FECHA Caracas, 21 de Febrero de 2007

ÍNDICE GENERAL

AGRADECIMIENTOS V

ÍNDICE GENERAL VI

ÍNDICE DE TABLAS VIII

ÍNDICE DE FIGURAS VIII

INTRODUCCIÓN IX

CAPÍTULO I. PLANTEAMIENTO DEL PROYECTO 1

I.1 PLANTEAMIENTO 1
I.2 OBJETIVO GENERAL 2
I.3 OBJETIVOS ESPECÍFICOS 2
I.4 JUSTIFICACIÓN 2
I.5 LIMITACIONES 3
I.6 ALCANCES 3

CAPÍTULO II. MARCO REFERENCIAL 5

II.1 VOZ SOBRE IP 5
II.2 H.323 5
II.3 PROTOCOLO DE INICIO DE SESIÓN "SESSION INITIATION PROTOCOL" (SIP) 8
II.3.i Peticiones SIP *9*
II.3.ii Mensajes de respuesta SIP *10*
II.3.iii Estructura de un mensaje SIP *11*
II.4 PROTOCOLO DE DESCRIPCIÓN DE SESIÓN "SESSION DESCRIPTION PROTOCOL" (SDP) 13
II.5 PROTOCOLO DE TRANSPORTE EN TIEMPO REAL "REAL TIME PROTOCOL" (RTP) 15
II.5.i Paquetes RTP *16*
II.6 PROTOCOLO DE CONTROL EN TIEMPO REAL "REAL TIME CONTROL PROTOCOL" (RTCP) 18
II.6.i Formato de paquetes RTCP *18*
II.6.ii Reglas de Tiempo *19*
II.7 CODEC 20
II.7.i Estándar G.711. PCM *20*
II.7.ii Estándar G.729. CS-CELP *22*
II.8 WATTCP 23

CAPÍTULO III. METODOLOGÍA 28

III.1 METODOLOGÍA Y DESARROLLO 28

CAPÍTULO IV. RESULTADOS 32

IV.1 SELECCIÓN DE CODEC 32
IV.1.i G711 PCM *32*
IV.1.ii G.729 CELP *35*
IV.2 SELECCIÓN DEL PROTOCOLO DE SEÑALIZACIÓN 39
IV.3 DISEÑO DE LA INTERFAZ DE CONVERSOR 41
IV.4 DESCRIPCIÓN DEL DISPOSITIVO IMPLEMENTADO 42
IV.4.i Programas residentes para manejo de interfaz con la tarjeta de red *42*
IV.4.ii Programa residente de conversión analógico/digital y digital/analógico: *43*
IV.4.iii Librería RTP *45*
IV.4.iv Librería SIP *46*
IV.4.v Funcionamiento General *47*

CAPÍTULO V. CONCLUSIONES Y RECOMENDACIONES 50

V.1 CONCLUSIONES Y RECOMENDACIONES 50
V.2 BIBLIOGRAFÍA 53

ANEXO A. ANÁLISIS DE LA IMPLEMENTACIÓN MÍNIMA DEL CONJUNTO DE PROTOCOLOS H323 55

ANÁLISIS DE LA IMPLEMENTACIÓN MÍNIMA DEL CONJUNTO DE PROTOCOLOS H323 56
A.1 CAPACIDADES DE MEDIOS 56
A.2 CAPACIDADES DE CONTROL 56
A.3 CAPACIDAD DE AUDIO 58
A.4 MENSAJES DE SEÑALIZACIÓN DE CANAL LÓGICO 59

ANEXO B IMPLEMENTACIÓN MÍNIMA DEL PROTOCOLO SIP 61

SIP IMPLEMENTACIÓN MÍNIMA 62
B.1 PROTOCOLOS MÍNIMOS NECESARIOS 62
B.2 COMUNICACIÓN SIP 64

ANEXO C ESTABLECIMIENTO DE UNA LLAMADA UTILIZANDO EL PROTOCOLO SIP 66

ESTABLECIMIENTO DE UNA LLAMADA UTILIZANDO EL PROTOCOLO SIP 67
C.1 LLAMADA SIN PROXY 67
C.2 REGISTRO EN SERVIDOR PROXY 70

APÉNDICE 1. MANUAL TÉCNICO DEL DISPOSITIVO 72

MANUAL TÉCNICO DEL TELÉFONO IP 73
1.1 HARDWARE 73
1.1.i Interrupción 8, Interrupción del timer 75
1.2 SOFTWARE 76
1.2.i Rutina de interrupción 76
1.2.ii Módulo RTP 78
1.2.iii Módulo SIP 81
1.2.iv Programa Principal 82
1.3 PROGRAMAS VARIOS 86

ÍNDICE DE TABLAS

TABLA NRO 1: COMPONENTES DEL ESTÁNDAR H.323 6
TABLA NRO 2: CAMPOS DEL PROTOCOLO SDP. 14
TABLA NRO 3: PRINCIPALES TIPOS DE CARGA PARA AUDIO EN RTP 17
TABLA NRO 4: METODOLOGÍA DE LA INVESTIGACIÓN 28
TABLA NRO 5: COMPARACIÓN DE ARCHIVO DE ENTRADA A CODEC G711 LEY A Y LEY μ CON LOS ARCHIVOS DE SALIDA CORRESPONDIENTES 33
TABLA NRO 6: ANCHO DE BANDA UTILIZADO USANDO EL CODEC G711 35
TABLA NRO 7: COMPARACIÓN DE ARCHIVO DE ENTRADA AL CODEC G729 CON EL ARCHIVO DE SALIDA CORRESPONDIENTES 38
TABLA NRO 8: ANCHO DE BANDA UTILIZADO USANDO EL CODEC G729 38

ÍNDICE DE FIGURAS

FIGURA NRO 1: CAPAS DEL CONJUNTO PROTOCOLO H.323 7
FIGURA NRO 2: PAQUETES RTP 16
FIGURA NRO 3: FORMATO DE PAQUETES RTCP 19
FIGURA NRO 4: DIAGRAMA DE BLOQUE DE CODIFICADOR CS-CELP 23
FIGURA NRO 5: ARCHIVO DE ENTRADA, IN.HEX 33
FIGURA NRO 6: ARCHIVO DE SALIDA CON LEY A, OUTA.HEX 34
FIGURA NRO 7: ARCHIVO DE SALIDA CON LEY M, OUTU.HEX 34
FIGURA NRO 8: ARCHIVO CODIFICADO EN CELP, CHANNELCELP.HEX 37
FIGURA NRO 9: ARCHIVO DE SALIDA, OUTCELP.HEX 37
FIGURA NRO 10: CONVERSORES DIGITAL/ANALÓGICO Y ANALÓGICO/DIGITAL 41
FIGURA NRO 11: DIAGRAMA DE ESTADOS SIP 48
FIGURA NRO 12: ESQUEMA DE ARREGLOS IBUF Y OBUF Y SU INTERACCIÓN CON RTP 49
FIGURA NRO 13: CAPAS DEL CONJUNTO PROTOCOLO SIP 62
FIGURA NRO 14: RED SIP 64
FIGURA NRO 15: LLAMADA SIN SIP PROXY 67
FIGURA NRO 16: REGISTRO EN SERVIDOR SIP PROXY 71
FIGURA NRO 17: INTERCONEXIÓN DEL HARDWARE 73
FIGURA NRO 18: DIAGRAMA DE BLOQUES DEL CONVERSOR DIGITAL A ANALÓGICO 74
FIGURA NRO 19: DIAGRAMA DE BLOQUES DEL CONVERSOR ANALÓGICO A DIGITAL 75
FIGURA NRO 20: DIAGRAMA DE FLUJO DE LA RUTINA DE INTERRUPCIÓN 77
FIGURA NRO 21: DIAGRAMA DE FLUJON DE LA FUNCIÓN GENERADORA DE PAQUETES RTP 79
FIGURA NRO 22: DIAGRAMA DE FLUJO DE LA FUNCIÓN RECEPTORA DE PAQUETES RTP 80
FIGURA NRO 23: DIAGRAMA DE ESTADOS DEL MÓDULO SIP 81
FIGURA NRO 24: DIAGRAMA DE PROCEDIMIENTOS DE INICIO DEL PROGRAMA PRINCIPAL 84
FIGURA NRO 25: DIAGRAMA DE FLUJO DEL CICLO INFINITO DEL PROGRAMA PRINCIPAL 85

INTRODUCCIÓN

El hombre siempre ha tenido la necesidad de comunicarse y constantemente ha buscado nuevas formas para lograrlo, desde las más primitivas hasta los nuevos sistemas de comunicación que existen actualmente. En 1976 Alexander Graham Bell inventó el teléfono iniciándose así la "Era de la comunicación"; en la misma se han incorporado de manera progresiva nuevas tecnologías que a su vez se traducen en nuevos servicios comunicacionales.

El perfeccionamiento tecnológico ha conducido a diferentes sistemas de comunicación, como son las redes telefónicas y de datos. El uso de las primeras, al mejorar progresivamente en el tiempo la fidelidad de la transmisión de la voz, se ha difundido ampliamente en la población. La segunda también se ha popularizado porque mejora y amplia la comunicación a distancias gracias a las ventajas que ofrece y las nuevas tecnologías que usa.

Las ventajas técnicas y económicas que actualmente presentan las redes de conmutación de paquetes han dirigido los esfuerzos en desarrollar soluciones para adaptar en estas redes los servicios de voz.

La tecnología de voz sobre protocolo IP (VoIP), ofrece nuevos y distintos servicios a los ofrecidos por la telefonía tradicional y su funcionamiento se basa en paquetes de voz transmitidos sobre redes de datos, que se encapsulan bajo diferentes protocolos para lograr la señalización de la comunicación y la transmisión en tiempo real.

Existen diversas especificaciones técnicas que definen las características a cumplir en una red de datos para ser capaz de brindar servicios de VoIP y en especial los componentes que la conforman, tenemos los Gateway, servidores, terminales entre otros. Con respecto a los terminales se posee información correspondiente al procesamiento de los datos y la señalización, pero en sistemas bastante elaborados que hacen complejo su estudio. Razón por la que el propósito de este proyecto es el diseño e implementación de un teléfono IP de mínimos requerimientos que permita una conversación en tiempo real y un mejor

entendimiento de su proceso en general. El equipo resultante de esta investigación podría utilizarse con fines educativos en la Escuela de Ingeniería Telecomunicaciones de nuestra universidad.

Con este propósito, inicialmente se ha realizado un estudio de los distintos protocolos para la transmisión de telefonía sobre IP, como también un análisis de diferentes Codecs existentes.

Para facilitar la comprensión del trabajo especial de grado éste se estructuró en 5 Capítulos: Capítulo I que se refiere al problema, el planteamiento del problema, el objetivo general, los específicos, la justificación, limitaciones y alcances del trabajo; Capítulo II que incluye marco teórico con antecedentes de la investigación, bases teóricas. Capítulo III se refiere a la metodología aplicada, comprende los métodos, técnicas y procedimientos requeridos para llevar a cabo la investigación, como también los problemas encontrados durante cada fase del trabajo; Capítulo IV explica cada uno de los resultados de la investigación; Capítulo V incluye las conclusiones y recomendaciones. Finalmente se incluyen las referencias bibliográficas, apéndices y anexos.

CAPÍTULO I. PLANTEAMIENTO DEL PROYECTO

El presente capítulo expone el planteamiento del proyecto el cual está constituido por el problema, el objetivo general, los objetivos específicos, la justificación, las limitaciones y alcances del trabajo. Se han incluido algunos cambios respecto a la propuesta original como producto del aprendizaje obtenido durante el proceso de investigación y sugerencia del tutor.

I.1 PLANTEAMIENTO

Hoy día las redes por las que primordialmente conversamos son: la Red de Telefonía Conmutada (RTC) y las de telefonía móvil (PLTS). La RTC es la más confiable debido a que usualmente podemos comunicarnos a través de ella, pero es a su vez ineficiente en lo que respecta a la cantidad de información enviada, puesto que una vez conmutada una llamada el circuito está constantemente enviando un flujo de datos incluso en los momentos de silencio de las conversaciones. Además, en una RTC el costo de la llamada se mide por el tiempo de duración de la misma, aunque nunca se haya transmitido información. Por otro lado estas redes transmiten voz, y la transmisión de datos o video resulta ineficiente, puesto que no están contemplados en su diseño.

Las nuevas tecnologías persiguen la integración del envío de voz, datos y video en una única red; una de éstas es la llamada "Voz sobre Protocolo IP" (VoIP, por sus siglas en inglés). Este nuevo tipo de tecnología facilita esta convergencia abriendo una amplia gama de nuevos servicios y al mismo tiempo reduciendo los costos de infraestructura y de operación, además busca integrar a las RTC a fin de permitir que se originen o terminen llamadas en teléfonos tradicionales según sea necesario, y permite la transmisión de señales de voz en paquetes comprimidos que son transportados a través de redes IP existentes en lugar de líneas telefónicas tradicionales, utilizando a la Internet y la pila de protocolos TCP/IP.

VoIP necesita de una nueva infraestructura en la arquitectura de la red, que permita a los paquetes de voz ser dirigidos correctamente a los terminales que deben estar en capacidad de emplear los protocolos TCP/IP para poder realizar el intercambio de información con la red.

Hoy día, existen varias investigaciones de VoIP con un código completamente abierto, como por ejemplo tenemos: Asterix, OpenH323, OpenZoep, SIPxchange PBX, SIP Express Router (SER), entre otros; todos orientados a la creación de aplicaciones multimedia "softphones" o Central telefónica digital PBXs de uso comercial o personal. Ninguno hasta ahora ofrece la posibilidad de estudiar el sistema completo de un hardphone, de una forma sencilla, que facilite la capacidad de entendimiento y de adaptación a nuevas aplicaciones; este es el propósito de este proyecto.

I.2 OBJETIVO GENERAL

Diseñar un teléfono IP, utilizando una Licencia Pública de uso General.

I.3 OBJETIVOS ESPECÍFICOS

- Analizar información de la tecnologia de VoIP.
- Diseñar un teléfono IP con características básicas de funcionamiento.
- Implementar el teléfono IP.

I.4 JUSTIFICACIÓN

La telefonía tal y como la conocemos está sumamente limitada a la comunicación analógica de voz a 8KHz. Hoy en día, con los avances tecnológicos y la evolución de Internet el servicio de telefonía se ha hecho anticuado, por ello nació la Telefonía IP, dando lugar a una serie de posibilidades mucho más amplias, gracias a la red de Internet, todo ello a través de VoIP.

Por lo anteriormente expuesto surgió el interés de desarrollar este proyecto "Diseño de un Teléfono de Voz Sobre IP" con la idea de profundizar acerca de la tecnología de VoIP en lo que se refiere a protocolos, programas "software" y equipo físico "hardware", fortaleciendo los conocimientos adquiridos durante la carrera y dando base para orientar a otras investigaciones que estén relacionados con una red de VoIP.

I.5 LIMITACIONES

- Se realizó el diseño utilizando las capacidades de un 486 embebido facilitado por el tutor. Estableciendo ciertas limitaciones sobre todo en la velocidad de procesamiento y el manejo de señales de audio (analógico/digital).
- La interfaz del conversor analógico/digital se realizó de la manera más sencilla posible, debido a limitaciones como, necesidad de una fuente bipolar para los integrados (ADC, DAC) que utilizan valores de ± 15 voltios (los integrados que manejan solo 5 voltios no se encontraron el mercado). Otra limitación fue sobre el sincronismo entre un reloj externo que requerían los integrados para la conversión A/D y D/A y la obtención o envío de la muestra por parte del 486 embebido resultaba en tiempos de pérdida para procesamiento de hasta 37 μsegundos (37×10^{-6} segundos).
- La cantidad de bits utilizados en las conversiones digital-analógica y analógica-digital (8 bits) no permite realizar una compresión puesto que para ello se necesitan al menos 12 bits de muestra, según las indicaciones de la serie de recomendaciones G.729, G711 de la UTI-T

I.6 ALCANCES

- El diseño del teléfono IP implica su conceptualización, implementación, prueba del dispositivo y creación de manual técnico.

- El trabajo incluye el diseño de una arquitectura del teléfono apta para cubrir las señales mínimas de aviso al usuario, y ejecutar los protocolos TCP/IP, y los protocolos superyacentes que permitan la comunicación.
- No se realizó un teléfono de VoIP comercial, que pueda competir con los teléfonos IP ya existentes en el mercado, ni con los servicios que estos ofrecen. Se buscó que el mismo sea funcional, entendiéndose por esto que maneje los mínimos requerimientos de un teléfono de VoIP para realizar una transacción de datos que de lugar a conversaciones telefónicas sobre una red IP.
- No se implementaron todos los servicios de los protocolos utilizados para el teléfono debido a su complejidad y a la limitación de tiempo. Se utilizaron soluciones ya implementadas para los protocolos TCP/IP sobre el sistema operativo DOS (librería WATTCP) y el resto de los protocolos se les dio una implementación mínima requerida, a fin de garantizar que el teléfono IP funcione en el plazo de tiempo establecido.
- Se busca poner en funcionamiento un teléfono IP, o hardphone, y no está contemplado un segundo teléfono IP. Para fines de prueba es necesario utilizar un programa en Visual Basic que simula los mensajes de señalización. Otra forma de prueba es la utilización del programa del teléfono IP sobre una computadora personal, que ejecute el sistema operativo DOS en modo real.

Una vez definido estos aspectos que son la base de nuestro proyecto se da inicio a concretar los fundamentos teóricos que nos ayudaran a entender y desarrollar la investigación.

CAPÍTULO II. MARCO REFERENCIAL

En este capítulo se describen dos de los más importantes protocolos de señalización para VoIP, como los diferentes codecs aplicables a estas tecnologías, además se hace una descripción del WATTCP TCP/IP y sus servicios.

II.1 VOZ SOBRE IP

Voz sobre IP, o VoIP, es una tecnología que busca la integración de las redes de voz y datos al permitir que se envíen paquetes de voz sobre una red inicialmente destinada a enviar datos (Davidson, J. & Peters, J., s.f.). Existen varios protocolos en su marco estructural, entre los más conocidos y los primeros a ser usados, se encuentra la pila de protocolos H.323 sugeridos por la Unión Internacional de Telecomunicaciones (UIT-T) (Khasnabish, B., 2003), conformado por el Protocolo de Transporte de Tiempo Real "Real Time Transport Protocol" (RTP), protocolo RAS, H225 y H245; luego se encuentra el Protocolo de Inicialización de Sesiones "Session Initiation Protocol" (SIP) definido por el Grupo de Trabajo en Ingeniería de Internet "Internet Engineering Task Force" (IETF) en el RFC 2543 y actualizado por los RFCs 3261, 3262, 3263, 3264, (http://www.rfc-editor.org) que se utiliza para iniciar sesiones interactivas multimedia entre usuarios de redes IP y que junto a RTP, y RTCP (RTP de control) descritos en los RFCs 3550 y 3551, definen otro conjunto de protocolos destinados a la señalización y transmisión de paquetes de voz sobre redes IP.

A continuación se explicará con más detalle el funcionamiento de los protocolos participantes en una comunicación de VoIP.

II.2 H.323

Según la UTI, la pila de protocolos H323 es una especificación para las comunicaciones de audio, video y datos a través de una red IP, la conforman un conjunto de protocolos de la serie H los cuales se van a encargar de dirigir la

señalización, control de llamada, transporte, control multimedia y de ancho de banda para conferencias punto a punto y multipunto. H.323 es una parte del estándar de sistema audio visual y multimedia "Audiovisual and Multimedia Systems", de series H.32X, donde se define la transmisión multimedia a través de diferentes tipos de medios como:

- H.320: Comunicaciones multimedia sobre RDSI.
- H.310 y H.321: Comunicaciones multimedia sobre ATM.
- H.324: Comunicaciones multimedia sobre RTC.
- H.323: Comunicaciones multimedia sobre IP.

En la siguiente Tabla se especifican algunos de los componentes que conforman el estándar H.323.

FUNCIÓN	PROTOCOLO
Señalización de llamadas	H.225
Control de medios	H.245
Codecs de audio	G711, G722, G723, G728, G729
Codecs de video	H.261, H.263
Compartir datos	T.120
Transporte de medios	RTP/RTCP

Tabla Nro 1: Componentes del estándar H.323
Fuente: Davidson, J. & Peters, J.

Como se puede observar, H.323 consiste en un conjunto de protocolos separados por funciones los cuales engloban todo el sistema y además utilizan la red de paquetes, como son redes locales (LAN), redes metropolitanas (MAN), redes empresariales e Internet, y están conformados por elementos como son: terminales, Gateways, Gatekeepers y unidades de control multipunto " Multipoint Control Units" (MCU), siendo los terminales los puntos finales de la red, los Gateways son traductores entre redes de paquetes con la Red Pública de Telefonía Conmutada

"Public Switched Telephone Network" (PSTN), los Gatekeepers proporcionan el control de admisión y servicios de traducción de direcciones para terminales o Gateways y las MCU permiten que dos o más terminales o Gateways realicen conferencias con sesiones de audio y/o video.(Rec. UIT-T H.323)

En la siguiente Figura se presentan varias capas y los protocolos en los que estos se encuentran.

<table>
<tr><td rowspan="2">H.245</td><td colspan="2">H.225</td><td colspan="2">Flujos de audio/video</td></tr>
<tr><td>Control de llamadas</td><td>RAS</td><td>RTCP</td><td>RTP</td></tr>
<tr><td colspan="2">TCP</td><td colspan="3">UDP</td></tr>
<tr><td colspan="5">IP</td></tr>
<tr><td colspan="5">Capas física /datos</td></tr>
</table>

Figura Nro 1: Capas del conjunto protocolo H.323
Fuente: Khasnabish

- RTP/RTCP (Real-Time Transport Protocol / Real-Time Transport Control Protocol) Protocolos de transporte en tiempo real que proporcionan servicio de entrega punto a punto de datos.

- RAS (Registro, admisión y estatus): Señalización de registro, admisiones y estado entre los equipos participantes.

- H225: Protocolo de control de llamada que permiten establecer una conexión y una desconexión.

- H.245: Protocolo de control usando el establecimiento y control de una llamada.

- Q.931: (señalación del suscriptor digital "Digital Subscriber Signalling") Este protocolo se define para la señalización de accesos red digital de servicios integrados (RDSI) básico.

- RSVP (Resource ReserVation Protocol): Protocolo de reserva de recursos en la red para cada flujo de información de usuario

Además de H.323, existe el conjunto de protocolos SIP el cual se presenta a continuación

II.3 PROTOCOLO DE INICIO DE SESIÓN "SESSION INITIATION PROTOCOL" (SIP)

Según la IETF, SIP es un protocolo de capa de aplicación que se utiliza para el control de la señalización en la creación, modificación, y culminación de sesiones entre agentes usuarios – aquellos dispositivos o terminales de datos capaces de utilizar el conjunto de protocolos SIP se denominan agentes usuarios. Estas sesiones incluye las llamadas telefónicas a través de Internet. Sus funciones principales son las siguientes (RFC 3261, 2002)

- Localización de un punto final "end point", también llamado agente usuario "user agent".

- Contactar al end point para determinar si está dispuesto a iniciar la sesión.

- Intercambio de la información media para permitir el establecimiento de la sesión, esto permite conocer las capacidades de cada dispositivo que se comunica y establecer un acuerdo de cómo se realizará la comunicación.

- Modificación de las sesiones medias existentes, como por ejemplo la forma en que se intercambia la información entre los puntos de comunicación.

- Finalización de sesiones.

El protocolo utiliza TCP o UDP, y nuevas extensiones han permitido que utilice medios más seguros como Seguridad para Capa de Transporte (TLS) o Protocolo de Transmisión de Control del Flujo de Datos SCTP. (Johnston, 2004).

El protocolo permite a los usuarios mantener un único identificador aunque cambie de red. Estos identificadores se denominan Identificador Uniforme de Recursos "Uniform Resource Identifier" (URI). Ejemplo: pedro_perez@ucab.edu.ve ó 12345@sipserver.com

El protocolo SIP es utilizado junto a otros protocolos para construir una arquitectura multimedia completa (RFC 3261, 2002), así, SIP sólo se encarga de la señalización de la sesión, pero para el intercambio de información entre los agentes usuarios se utiliza el protocolo RTP.

En el establecimiento, modificación y culminación de sesiones, SIP utiliza dos tipos de mensajes: peticiones y mensajes informativos.

Los mensajes informativos son aquellos que se utilizan para intercambiar información de los eventos ocurridos luego de la petición, por ejemplo la ocurrencia de errores.

II.3.i PETICIONES SIP

Las peticiones SIP, son aquellos mensajes que piden al otro equipo SIP ejecutar una acción específica (Johnston, 2004) a través de cadenas de caracteres denominadas métodos. Los métodos principales explicados por Johnston y en el RFC 3261 son: INVITE, REGISTER, BYE, ACK, CANCEL y OPTIONS.

- INVITE: Es el método que se utiliza entre los agentes usuarios para establecer la sesión media, usualmente el cuerpo del método contiene la información media que es capaz de utilizar el agente usuario que inicia la llamada. Este método, además, envía dos identificadores únicos al destino para hacer la

identificación única del dialogo, un identificador se llama Tag, un número de al menos 32 bits de aleatoriedad que se envía asociado al URI del agente usuario de origen, y el otro identificador, asociado a la sesión, es llamado Call-ID.

- REGISTER: Este método lo manejan los Agentes usuarios para notificar a una red SIP su URI de contacto actual.
- BYE: Este método se envía por cualquiera de los agentes usuarios para finalizar su sesión media.
- ACK: para confirmar que la conexión ha sido establecida con éxito.
- CANCEL: El método se usa para terminar búsquedas pendientes o intentos de sesión pendientes. Si la sesión ya ha sido establecida, el método CANCEL no genera ninguna acción, y se deberá terminar la sesión mediante el método BYE.
- OPTIONS: Con este método se descubren las capacidades y disponibilidad de un servidor o agente usuario. Como respuesta se envía una lista de las capacidades del agente usuario o servidor receptor. La respuesta a este método se maneja de la misma manera que cuando se responde a un método INVITE, es decir con mensajes informativos del estado en que se encuentra, pudiendo agregar información extra indicando sus capacidades y opciones soportadas.

II.3.ii MENSAJES DE RESPUESTA SIP

Los mensajes de respuesta se dividen en seis categorías, dependiendo del primer dígito en el mensaje (RFC 3261, 2002), así:

- Los mensajes del 100 al 199 son informativos.
- Los mensajes del 200 al 299 son de éxito
- Los mensajes del 300 al 399 son de redirección
- Los mensajes del 400 al 499 indican que ha ocurrido un error en el cliente
- Los mensajes del 500 al 599 indican un error en el servidor

- Los mensajes del 600 al 699 indican un error global

II.3.iii ESTRUCTURA DE UN MENSAJE SIP

Todos los mensajes SIP se remiten en texto plano(sin formato codigo ASCII), y envían una línea de petición o de estado, dependiendo de si se trata de una petición o respuesta, y seguidamente una serie de campos de cabeceras separados por caracteres de retorno de carriage "Carry Return" y Nueva Línea "Line Feed" dentro del mensaje para informar respecto a la petición o la respuesta.

La línea de petición se crea de la siguiente manera:

Método SP (URI que crea la petición) SP (Versión SIP) CRLF

Donde SP representa el carácter de espacio y CRLF representa la combinación de caracteres 'Carry Return' y 'Line Feed'.

La línea de estado colocada en las respuestas posee el siguiente formato

(Versión SIP) SP (Código de estado) SP (Razón o frase) CRLF

Donde nuevamente SP representa el carácter de espacio y CRLF representa la combinación de caracteres 'Carry Return' y 'Line Feed', y el código de estado es uno de los posibles números de tres dígitos que identifican la respuesta, y la razón o frase es una explicación corta de la respuesta, por ejemplo: Ocupado.

Los campos de cabecera se definen como 'Cabecera: Campo', donde ambos son independientes del tipo de letra, la 'Cabecera' es utilizado para representar el nombre del campo, y 'Campo' es donde se encuentra la información. El orden de estos campos no es importante en el mensaje, y la mayoría no es obligatorio, pero se requiere una información mínima en el mensaje dependiendo del tipo de petición que se esté ejecutando (RFC 3261). Así se tiene que la petición INVITE requiere de los campos de cabecera Call-ID, CSeq, From, To, Via, Contact, Max-Forwards, y todos

los demás mensajes de petición requieren de al menos los mismos campos de cabecera excepto el campo Contact.(Johnston, 2004)

Según el RFC 3261 cada campo obligatorio de un mensaje SIP posee el siguiente significado

CALL-ID: Como parte de identificación del diálogo entre dos agentes usuario se utiliza el campo Call-ID. Este debe ser único entre llamadas excepto en las peticiones de registro, en cuyo caso se debe mantener constante. Usualmente para crear un Call-ID se utiliza un identificador local, que debiera ser un número aleatorio aunque esto no es obligatorio, o también es usual encontrar el identificador local, el símbolo @, y el nombre del host o su dirección IP.

- CONTACT: El campo de contacto es utilizado para colocar el URI que identifica al destino o al origen, dependiendo de si el mensaje está presente en una petición o respuesta.

- CSEQ: Este campo contiene un número decimal que se incrementa cada vez que se realiza una petición. Normalmente se incrementa en uno, excepto cuando la petición es un CANCEL o un ACK, en cuyos casos el campo Cseq utiliza el mismo número que estuvo presente en el INVITE. Estos campos se usan para determinar las peticiones fuera de orden o para diferenciar entre una petición nueva o una retransmisión. Cada punto en la comunicación inicializa su propio número de secuencia cuando realiza una petición.

- FROM: En este campo se coloca el URI del agente usuario que genera la petición y no desde donde se está originando el mensaje, puesto que en la respuesta a una petición el campo From no varía. También se agrega un Tag, o número aleatorio, que se utiliza para identificar al dialogo. Se da como opción indicar un nombre representativo (display name) en este campo; el URI debe ser colocado entre los símbolos '<>'.

- TO: El URI destino de una petición se señala en este campo. Un Tag se agrega en un mensaje de respuesta '200 OK' y se mantiene durante todos los mensajes siguientes para identificar el dialogo.

- VIA: Se utiliza este campo para grabar la ruta que ha transitado una petición y para enrutar de regreso la respuesta al origen. El orden que se mantiene dentro de este campo es muy importante, cuando la petición está siendo enviada a su destino el agente usuario que genera el mensaje agrega su URI en el campo y sucesivamente cada servidor que recibe el mensaje agrega su dirección al principio de la lista, en el enrutamiento de regreso de la respuesta cada servidor que recibe el mensaje revisa la primera dirección dentro de este campo y si corresponde con la suya elimina esa línea y reenvía el mensaje a la siguiente dirección de la lista; cuando no hay más direcciones en la lista el agente usuario asume que la respuesta es para él.

- MAX-FORWARDS: En este campo se coloca un número decimal que indica la cantidad máxima de saltos que puede dar el mensaje antes de ser descartado. El valor es decrementado por cada servidor Proxy que reenvíe el mensaje, y si en algún punto un servidor Proxy recibe un mensaje con el campo Max-Forwards igual a cero, descarta el mensaje y envía una respuesta '483 demasiados saltos' al origen del mensaje.

II.4 PROTOCOLO DE DESCRIPCIÓN DE SESIÓN "SESSION DESCRIPTION PROTOCOL" (SDP)

El protocolo utilizado para el intercambio de las capacidades de comunicación es el SDP; un paquete SDP va dentro de un paquete SIP durante el establecimiento de la comunicación y es utilizado para indicar al protocolo subsiguiente de intercambio de paquetes de voz en que formato debe enviar la información. El protocolo al igual que SIP utiliza texto plano y cabeceras que representa con un carácter alfabético

minúsculo, seguido del carácter '=' para el envío de la información. Existen 16 campos posibles:

Campo	Descripción	Tipo
v=	Número de versión del protocolo	Obligatorio
o=	Creador de la sesión	Obligatorio
s=	Nombre de la sesión	Obligatorio
c=	Información de la conexión	Obligatorio
t=	Tiempos de inicio y fin de sesión	Obligatorio
m=	Información media	Opcional
a=	Atributos media	Opcional

Tabla Nro 2: Campos del protocolo SDP.
Fuente: Johnston

A continuación se explican los campos obligatorios de SDP y los campos opcionales más relevantes

- VERSIÓN DE PROTOCOLO (V): Contiene el número de versión del protocolo. Puesto que la versión actual de SDP es cero, un mensaje SDP válido comienza con v=0.

- ORIGEN (O): El campo contiene la información acerca del originador de la sesión e identificadores de sesión. Este campo se utiliza para identificar a la sesión. La estructura del campo es: o= (Nombre de usuario) (identificador de sesión) (versión) (tipo de red) (tipo de dirección) dirección.

- NOMBRE DE SESIÓN (S): Contiene un nombre para la sesión. Debe contener al menos un carácter.

- INFORMACIÓN DE CONEXIÓN (I): Este campo contiene la información de la conexión media. El formato del campo es: c=(tipo de red) (tipo de dirección) (dirección de conexión)

El tipo de red está definido como IN para Internet. La dirección tipo está definida como IP4 para direcciones IP versión 4, e IP6 para direcciones IP versión 6. La dirección de conexión será la dirección IP que enviará los paquetes con la información media.

- TIEMPOS DE INICIO Y FIN DE SESIÓN (T): En este campo se coloca el tiempo en que inicia la sesión y el tiempo en que termina. Se puede indicar un inicio y fin de sesión con valores igual a cero, lo que indica que la sesión es indefinida.

- INFORMACIÓN MEDIA (M): El formato de este campo viene dado por: m=media puerto transporte (lista de formatos).

El parámetro en 'media' puede ser 'audio, video, application, data, telephone-event, o control'. El parámetro puerto contiene el número del puerto. 'Transporte' contiene el protocolo de trasporte utilizado, el cual puede ser 'RTP', 'AVP' o 'UDP'. Finalmente el campo de 'lista de formatos' contiene más información de la información media. Usualmente contiene la carga de información que puede ser llevada en los paquetes de RTP.

- ATRIBUTOS MEDIA (A). Este campo contiene los atributos de la sesión media que se encuentra en el campo anterior. Puede haber uno o más campos de atributos por cada tipo de carga listada en el campo de información media.

Como se ha afirmado previamente, H.323 y SIP necesitan de los servicios del Protocolo de Transporte en Tiempo Real (RTP) el cual es descrito en las siguientes líneas.

II.5 PROTOCOLO DE TRANSPORTE EN TIEMPO REAL "REAL TIME PROTOCOL" (RTP)

El protocolo RTP provee funciones para transporte en tiempo real, como audio, video y datos, sobre una red unicast o multicast (RFC 3550). Fue diseñado por

el grupo de trabajo de Transporte de Audio/video del IETF y ha sido adoptado como parte de las recomendaciones H.323 del ITU.

Según la IETF, RTP consiste de dos partes: el protocolo RTP encargado de transportar la información media y el protocolo de control RTP (RTCP), encargado de monitorear la calidad de servicio como también enviar información a los participantes de la sesión.

II.5.i PAQUETES RTP

Un paquete RTP se muestra en la siguiente Figura. Los primeros 12 bytes de cabecera deben estar presentes en todo paquete RTP, mientras que los siguientes bytes de identificador de otras fuentes contributivas (CSRC) son opcionales para brindar servicios en sesiones multicast, por lo que no serán explicados aquí.

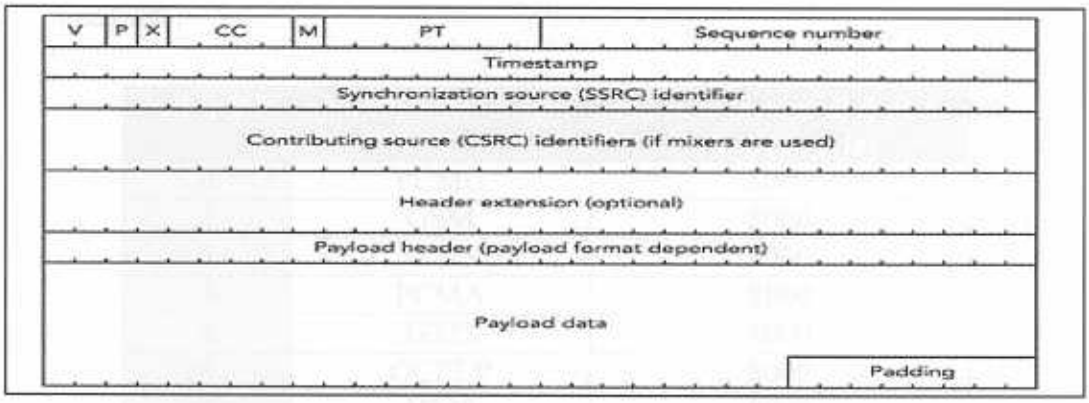

Figura Nro 2: Paquetes RTP
Fuente: Perkins

Los siguientes campos son explicados a continuación:

- VERSIÓN (V): este campo utiliza 2 bits e identifica la versión de RTP utilizada. La versión actual es la 2, y el valor que se encuentra en este campo es $2_{(16)}$.

- PADDING (P): Este bit indica si la carga del paquete contiene uno o más bytes que deban ser ignorados; la cantidad de bytes que deben ser ignorados se indica con el último byte de los datos enviados.

- EXTENSIÓN (X): El bit de extensión indica que luego de los 12 bytes de cabecera obligatorios, continúa una extensión de la cabecera utilizada para permitir implementaciones individuales.

- CONTADOR CSRC (CC): indica la cantidad de identificadores CSRC que siguen luego de la cabecera, este campo utiliza 4 bits.

- MARKER (M): Este bit se utiliza para marcar eventos de interés en un flujo de datos.

- TIPO DE CARGA (PT): Identifica el formato de carga RTP que contiene el paquete.

Valor PT	Nombre de codec	Frecuencia de reloj para muestreo (Hz)
0	PCMU	8000
2	GSM	8000
3	G723	8000
4	PCMA	8000
5	G722	8000
6	QCELP	8000
7	G728	8000
8	G729	8000

Tabla Nro 3: Principales tipos de carga para audio en RTP
Fuente: RFC 3551

- NÚMERO DE SECUENCIA: El número de secuencia se incrementa por cada paquete de datos RTP que se envía, y puede ser utilizado por el receptor para detectar pérdida de paquetes o para reorganizar paquetes recibidos. El valor inicial debe ser aleatorio.

- TEMPORIZADOR " TIMESTAMP": Refleja el instante de muestreo del primer octeto en el paquete RTP, es un valor aleatorio y se incrementa independientemente de cualquier otra fuente de datos en tiempo real, su granularidad debe ser suficientemente pequeña como para permitir una sincronización de los datos que procedan de la misma fuente.

- SSRC: Este campo es un identificador elegido aleatoriamente, que identifica a cada participante en la sesión.

- CARGA "PAYLOAD": los datos útiles de información se encuentran en esta parte del paquete; es posible que al final del mismo existan varios bytes en cero lo cual se indicaría con el bit de padding, y el último byte de la carga indica la cantidad de bytes que se deben ignorar, incluyendo a él mismo.

II.6 PROTOCOLO DE CONTROL EN TIEMPO REAL "REAL TIME CONTROL PROTOCOL" (RTCP)

Provee un reportaje periódico de la calidad de recepción, identificación de los participantes y otras informaciones de descripciones de las fuentes, notificación de cambios en los miembros de la sesión y la información necesaria para sincronizar los flujos de datos (Perkins, 2003)

II.6.i FORMATO DE PAQUETES RTCP

Existen cinco tipos de paquete RTCP: reporte de receptor, reporte de transmisor, descripción de fuente, manejo de miembros, fin de participación y definido por aplicación. Todos poseen una estructura como la siguiente.

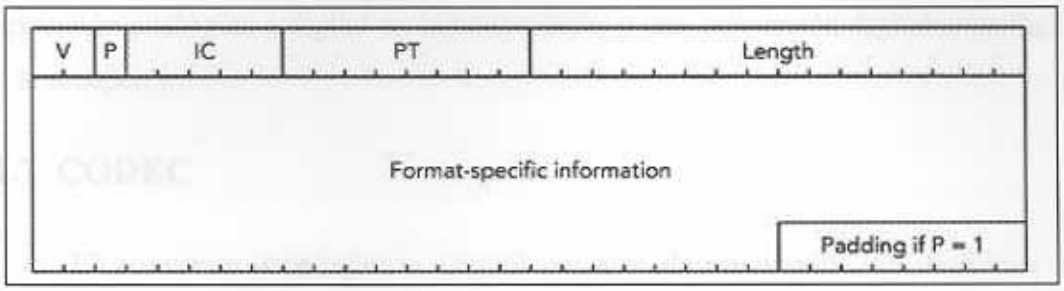

Figura Nro 3: Formato de paquetes RTCP
Fuente: Perkins

Los bits para número de versión y para padding siguen las mismas reglas que las vistas anteriormente en la sección anterior de RTP. Luego:

- CONTEO DE ITEMS (IC): algunos paquetes contienen una lista de items. Este campo indica su cantidad.
- TIPO DE PAQUETE (PT): contiene el valor constante 200 que lo identifica como un paquete RTCP.
- LONGITUD: Denota la longitud del paquete RTCP en palabras de 32 bits, incluyendo la cabecera del paquete y cualquier dato de relleno.

II.6.ii REGLAS DE TIEMPO

La velocidad a la cual cada participante envía paquetes RTCP no es fija, sino que varía de acuerdo al ancho de banda utilizado por la sesión y el formato de media utilizado. Lo ideal es restringir el total de tráfico RTCP a una fracción del total de ancho de banda utilizado por la sesión, y según recomendación en el RFC 3550, este ancho de banda debe situarse en alrededor del 5% del tráfico total utilizado por RTP.

Para que RTP maneje los mensajes de voz, se necesita de una previa transformación, adaptando la voz analógica al sistema digital. El encargado de esta función se denomina "Codec" (codificador/decodificador) el cual consiste de una

conversión analógica a digital en la transmisión, y una conversión digital a analógica en la recepción.

II.7 CODEC

El conversor Analógico a Digital consiste de un circuito de muestreo, un cuantificador y un Modulador por Impulsos Codificados (PCM). El circuito de muestreo provee muestras de voltajes discretos tomados a un intervalo regular de tiempo, de una señal analógica, el cuantificador se encarga de aproximar los voltajes analógicos, y por tanto infinitos, a niveles predeterminados de voltaje. El codificador PCM convierte cada nivel cuantificado a una palabra binaria.

En la recepción, el conversor Digital a Analógico recibe los pulsos de voltajes binarios y los convierte a niveles de voltaje cuantificados que se aproximen al valor original de la señal analógica. (Glover I. & Grant P., 1998)

Los codecs hacen un amplio uso de técnicas de procesamiento de señales para codificar eficientemente la señal antes de transmitirla, y también para decodificarlas en la recepción. Entre los estándares de codificación de telefonía IP normalmente encontramos el estándar G.711 con ley A o ley μ, utilizadas para la compansión, y opcionalmente los estándares, G723.1, G.728, G.729 entre otros.

Principalmente se toma en cuenta para el estudio los codecs G.711 y el G.729 los cuales son explicados seguidamente.

II.7.i ESTÁNDAR G.711. PCM

El estándar G711 establecido por la UIT-T (Rec. UIT-T G.711), tiene como fin principal la codificación de muestras de voz en palabras de 8 bits. (ITU-T Users' Group on Software Tools, 2005). El algoritmo utiliza dos maneras distintas para lograr esto: la ley μ y la ley A, la primera adoptada principalmente en Estados Unidos y Japón, y la segunda en Europa.

La forma más sencilla de alcanzar una codificación de 8 bits es realizarla de manera linear, tomando muestras de la señal en 8 bits, pero esto conlleva al problema de que la relación señal a ruido de las señales varía con respecto a la señal de entrada, es decir, mientras menor es la amplitud de la señal de entrada menor la relación señal a ruido, y esto resulta en una variación de la calidad de servicio. Para evitar esto, se crean las leyes μ y A, descritas a continuación

Ley μ

La ley μ se describe con la siguiente formula: (ITU-T Users' Group on Software Tools, 2005)

$$c(x) = x_{\max} \frac{\ln\left(1 + \mu |x| / x_{\max}\right)}{\ln(1+\mu)} \operatorname{sgn}(x) \tag{1}$$

El estándar G.711 de la UIT-T elige el valor de μ = 255, y como ya se menciono 8 bits de codificación. La relación señal a ruido para señales grandes es: (ITU-T Users' Group on Software Tools, 2005).

$$SNR_{\mu} = 6.02B + 4.77 - 20\log_{10}(\ln(1+\mu)) \tag{2}$$

Donde B es el número de bits utilizados, como el estándar utiliza 8 bits se obtiene:

$$SNR_{\mu} = 6.02B + 9.99 = 38.17dB \tag{3}$$

Ley A

La ley A se puede describir de la siguiente manera: (ITU-T Users' Group on Software Tools, 2005).

$$c(x)=\begin{cases}\dfrac{A|x|}{1+\ln(A)}\operatorname{sgn}(x), & 0\le\dfrac{|x|}{x_{\max}}\le\dfrac{1}{A}\\[2ex] x_{\max}\dfrac{1+\ln(A|x|/x_{\max})}{1+\ln(A)}\operatorname{sgn}(x), & \dfrac{1}{A}\le\dfrac{|x|}{x_{\max}}\le 1\end{cases} \tag{4}$$

Para el cual se eligió en el estándar el valor de A = 87.56. La relación señal a ruido para señales grandes utilizando la compresión ley A: (ITU-T Users' Group on Software Tools, 2005)

$$SNR_A = 6.02B + 4.77 - 20\log_{10}(1+\ln(A)) \tag{5}$$

De la misma forma que en la compresión utilizando ley μ: B = 8, de donde se obtiene:

$$SNR_A = 6.02B + 10.1 = 38.06dB \tag{6}$$

II.7.ii ESTÁNDAR G.729. CS-CELP

El códec CS-ACELP se basa en el modelo de codificación mediante la predicción lineal con excitación por código (CELP) (Rec. UIT-T G.729). Opera con tramas vocales de 10 ms correspondientes a 80 muestras a una velocidad de muestreo de 8000 muestras por segundo. En cada trama de 10 ms se analiza la señal vocal para extraer los parámetros del modelo CELP (coeficientes de filtros de predicción lineal, ganancias e índices de las tablas de códigos adaptivos y fijos). (UIT-T, 1996). El principio de codificación puede observarse en la siguiente Figura.

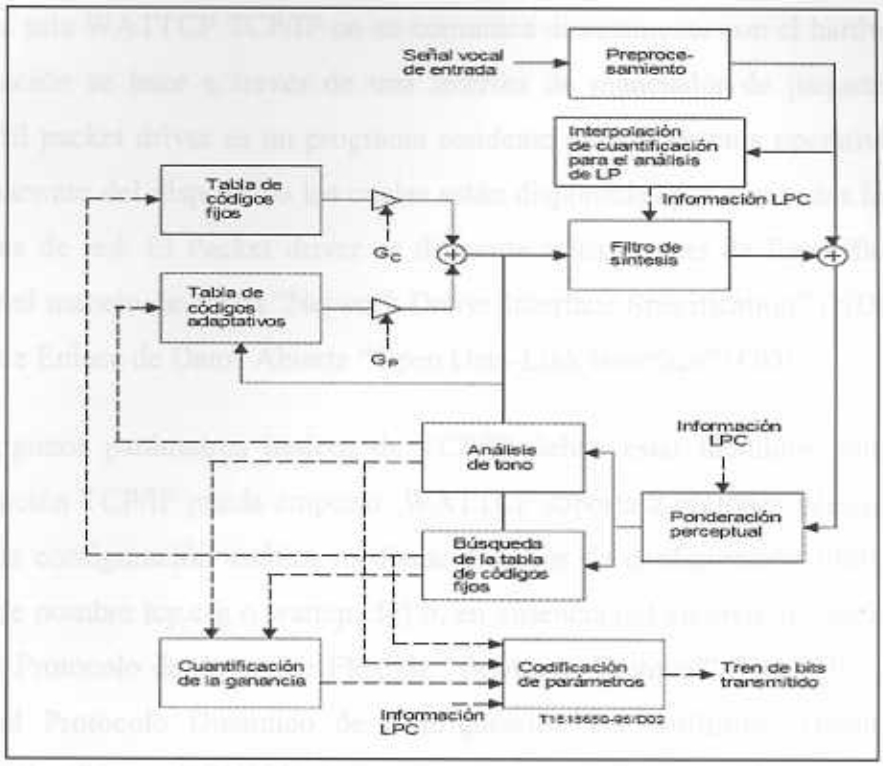

Figura Nro 4: Diagrama de bloque de Codificador CS-CELP
Fuente: Rc. G729

Como ya se ha dicho, todos estos protocolos (SIP, H.323, RTP…) necesitan de los servicios de las capas inferiores para poder establecer la comunicación de VoIP, es por esto que se utiliza las librerías de TCP/IP llamada WATTCP el cual se describe a continuación.

II.8 WATTCP

Waterloo TCP (WATTCP) es una librería de funciones para dar servicios de TCP/IP a los protocolos de capas superiores en el sistema operativo DOS – no es una pila TCP/IP. Cada función de WATTCP está pre-compilada en la librería, haciendo que las funciones solo sean copiadas en el archivo binario ejecutable final. Posee la desventaja de que la computadora corriendo el un programa WATTCP no sea realmente un verdadero host en una red IP. No es posible hacerle ping desde otro host a menos que se este corriendo un demonio.

La pila WATTCP TCP/IP no se comunica directamente con el hardware, esta comunicación se hace a través de una interfaz de manejador de paquetes"packet driver". El packet driver es un programa residente para el sistema operativo DOS o para el harware del dispositivo los cuales están disponibles por casi todas las marcas de tarjetas de red. El Packet driver es diferente a los drivers de Especificación de interfaz del manejo de la red "Network Driver Interface Specification" (NDIS) y a la Interfaz de Enlace de Datos Abierta "Open Data-Link Interface" (ODI).

Algunos parámetros básicos de TCP/IP deben estar incluidos antes que la comunicación TCP/IP pueda empezar. WATTCP soporta 2 métodos básicos, uno de ellos es la configuración estática mediante archivos de configuración (utilizando un archivo de nombre tcp.cfg o wattcp.cfg) o, en ausencia del archivo de configuración, utiliza el Protocolo de Arranque Flexible "Bootstrap Protocol" (BOOTP), antecesor del actual Protocolo Dinamico de Configuración del Anfitrion "Dinamyc Host Configuration Protocol"(DHCP), pero con quien mantiene compatibilidad.

El archivo mínimo de configuración contiene la dirección IP y la máscara de red, su Router/Gateway para posibilidad de conexión con la red WAN, y uno o más Servidores de Resolución de Nombres "Domain Name Server" (DNS).

Si un Router/Gateway no es indicado la computadora solo podrá ser capaz de comunicarse con computadoras dentro de la red local (LAN), y si el nombre del servidor no es suministrado, solo podrá trabajar con direcciones IP y no con el dominio de nombres.

La programación en WATTCP TCP/IP está basada en el concepto de un socket. Es un modelo donde se puede abrir, cerrar, leer y escribir hacia y desde la red como si se trataran escrituras en archivos de computador. El envío y recepciones de los paquetes con el uso de los sockets permiten establecer las comunicaciones orientadas a la conexión usando el Protocolo de Control de Transmisión "Transfer

Control Protocol" (TCP) o no orientadas a la conexión utilizándole Protocolo de Datagrama de Usuario "User Datagram Protocol" (UDP).(Engelke, 2003)

A continuación se presentan las descripciones de las principales funciones de WATTCP utilizadas en el proyecto

- udp_open() abre un puerto UDP.

*Int udp_open(void *s,Word lport, longword ina,Word port, int(*signal_handler));*

Udp_open abre un socket tipo udp para transferir datos de una manera no orientada a la conexión, o recibirlos de la misma manera.

Para especificar el host del cual procederán los paquetes o al cual serán transmitidos los paquetes udp, se utiliza el parámetro dela función 'ina'. Si este parámetro posee un valor de -1 en el momento de la inicialización, se recibirán paquetes de cualquier dirección IP remota y los envíos se realizarán en modo de difusión. Si el parámetro 'ina' posee un valor de cero, el siguiente paquete recibido por el puerto indicado en 'lport' completará la conexión y las siguientes lecturas del socket sólo se harán de la dirección IP obtenida de ese paquete, las siguientes escrituras al socket se enviarán a la dirección IP obtenida. Si se escribe a un socket udp con una dirección 'ina' igual a cero, los paquetes UDP serán enviados a la dirección no válida 0.0.0.0. Si el valor en 'ina' posee una dirección IP válida, cualquiera de los host, tanto el local como el especificado en 'ina', pueden iniciar la transmisión de datos.

Udp_open retornara 0 si el socket no puede ser abierto. Retornará cualquier otro valor en caso contrario.

- sock_recv_init() Inicia arreglos para el almacenamiento temporal dedatagramas udp

*Int sock_recv_init (void *socket,byte *buffer,int bufferlen);*

sock_recv_init() instala un arreglo, por lo general bastante grande, el cual será dividido en segmentos de arreglos más pequeños. Siempre que un datagrama UDP llegue, las funciones de WATTCP lo colocarán en el arreglo 'buffer'; subsecuentes lecturas del socket con la funcion sock_recv() tomarán el datagrama del arreglo 'buffer'.

- sock_recv() lee un datagrama de los buffers instalados por sock_recv_init()

*Int sock_recv(void *socket,byte *buffer,int maxlength,int flags);*

Explora el arreglo instalado por sock_recv_init() en busca de cualquier datagrama recibido por el socket. Si hay un datagrama , la longitud es devuelta y se llena el arreglo del usuario 'buffer'. Si no fueron encontrados datagramas para leer la función retorna cero.

- sock_fastwrite() intenta escribir datos a una secuencia

*Int sock_fastwrite(void *s, byte *data, int len);*

Sock_fastwrite intenta escribir los datos sobre el socket de salida y trasmitir lo más rapidamente posible. Si los datos no pueden ser transmitidos durante la función, el número de bytes transmitidos será retornado.

- tcp_tick realizar el proceso del sistema

Word tcp_tick(void *s);

WATTCP debe constantemente estar haciendo trabajos de chequeo por nuevos paquetes, procesando estos paquetes, y ejecutando retransmisiones de datos perdidos. Para que estos procesos se lleven a cabo, se debe llamar constantemente a la función tcp_tick().

Tcp_tick toma un parámetro, a saber un puntero al socket. Si el socket puntero es NULL, el valor retornado en indefinido. Si el socket puntero señala para un socket valido abierto, tcp_tick retornara uno de dos valores.

0 - Conexión reestablecida o cerrada por otro host

No 0 - Conexión es correcta

Tcp_tick() realiza procesos en todos los sockets sobre cada invocación, el puntero es simplemente una medida de la conveniencia para determinar el estatus de la conexión actual de ese socket.

Lo descrito en este capítulo representa el fundamento y las bases que son utilizadas para desarrollar este proyecto, pero el procedimiento utilizado se explica en el siguiente capítulo.

CAPÍTULO III. METODOLOGÍA

En este capítulo se presenta la metodología y se describen detalladamente los métodos, técnicas, procedimientos y las actividades realizadas en la investigación. Se presentan también los problemas que se consiguieron durante el desarrollo del proyecto y como fueron estos resueltos.

III.1 METODOLOGÍA Y DESARROLLO

Para el desarrollo del Trabajo Especial de grado se utilizo una metodología que se explicará y desarrollará a continuación, en donde se resume cada una de las fases en las que se dividió el estudio.

Fases de la metodología	Actividades realizadas
Recopilación de información	Recopilar teoría de los estándares de VoIP.
	Organizar teoría de los estándares de VoIP.
Selección del conjunto de protocolos	Seleccionar Codec
	Seleccionar protocolos de señalización VoIP
Selección de interfaz de usuario	Diseñar hardware del teléfono IP
Diseño de aplicación	Pruebas con la librería WATTCP
	Implementar programas utilizados en el teléfono IP
Diseño del manual	Escribir manual
Pruebas finales	Realizar pruebas finales del teléfono

Tabla Nro 4: Metodología de la investigación
Fuente: Propia

Fase 1: Recopilación de información

La recopilación comprendió la búsqueda y análisis de información de VoIP. Mediante el cual se obtuvieron los elementos importantes que componen un teléfono IP, como lo son los estándares de los protocolos de señalización, protocolos de comunicación en tiempo real y los codecs.

Se investigó bibliografías, RFC, Papers, investigaciones anteriores que dieron como resultado el fundamento teórico del proyecto, dando base para la selección de los protocolos y los codecs a implementar.

Durante el desarrollo teórico se plantearon las diferentes alternativas posibles que podrían implementarse en el diseño final del teléfono IP. Estas alternativas se presentan a continuación:

- Conjuntos de protocolos de señalización y comunicación: SIP, H.323, IAX
- Codecs de voz: G.711, G.723, G.729, G.729A, IAX

Fase 2: Selección del conjunto de protocolos

Se seleccionó técnicamente entre el conjunto de protocolos existentes para formar un teléfono IP, que pueda utilizar los mínimos requerimientos necesarios para que el dispositivo esté en capacidad de iniciar, procesar y terminar una comunicación. Para esto se realizaron pruebas entre posibles codec a implementar, seleccionando el G.711 como principal codec ya que es obligatorio para establecer cualquier comunicación de VoIP, de igual forma para elegir entre los protocolos de VoIP (H323, IAX ó SIP) se estableció un estudio comparativo dando como resultado la elección del protocolo SIP, el protocolo IAX no fue considerado en este estudio debido a que durante esta fase aún se encontraba estandarizado sino en borradores por parte de la IETF (http://www.ietf.org/internet-drafts/draft-guy-iax-02.txt). Para los servicios TCP/IP se utilizó la librería llamada WATTCP.

Fase 3: Selección de interfaz de usuario

Por otro lado, se seleccionó los mínimos mecanismos que permitan una interfaz adecuada con el usuario, dando la posibilidad que se establezca, termine y atienda una llamada.

Debido a las limitaciones del procesador utilizado, la cantidad de puertos mediante el cuál se podían establecer interfaces con el usuario son reducidas, siendo la principal forma de comunicación con medios externos el uso del puerto paralelo. Debido a que el puerto paralelo fue principalmente destinado a la conversión digital/analógica y analógica/digital la interfaz con el usuario se llevó a cabo solo con algunos pines del puerto paralelo, sin utilizar teclados o pantallas digitales; la mínima interfaz posible es utilizando un pin para la indicación de una llamada entrante y un pin para cuando se desee iniciar una llamada, atenderla o finalizarla. En un computador personal existe la posibilidad de enviar mensajes al usuario a través del monitor y recibir los mensajes del usuario mediante la lectura del teclado.

Fase 4: Diseño de aplicación

Se diseñó la aplicación que se ejecuta en el dispositivo y que es la encargada del control entre la interfaz del usuario y la pila de protocolos que realizan la señalización (SIP) y el envío de información de voz (RTP). La programación se realizó en una computadora personal con sistema operativo DOS, puesto que permitía más fácilmente la compilación y modificación de los códigos fuentes.

La aplicación se ejecuta en un procesador 486 Embebido, permitiendo que también se ejecute en cualquier ordenador personal que ejecute el sistema operativo DOS.

El diseño en una computadora personal tiene como fin facilitar el proceso de depuración del programa.

Fase 5: Creación de manual

Se escribió el manual técnico del funcionamiento del dispositivo; en este manual se explica con detalle la implementación de software del dispositivo, y los dispositivos de hardware utilizados, además se incluye las principales funciones utilizadas del programa RTP y SIP. Las características y limitaciones de cada uno.

Fase 6: Pruebas finales

El software y hardware se probaron a medida que se construían los diferentes componentes del dispositivo, una prueba final que se realizó bajo una red IP en la que el dispositivo cumplió con todas las funciones programadas de inicialización de sesión, transmisión y recepción de paquetes RTP y finalización de la sesión

Para realizar las pruebas se utilizó un simulador de mensajes SIP realizado con la librería de SIP creada para la programación del teléfono IP. Un computador personal ejecutando el simulador de mensajes se conecto en una red IP, junto con el teléfono IP.

Se observó también la comunicación mediante el programa Ethereal – sirve observar los paquetes en una red Ethernet – instalado en una computadora personal con sistema Operativo Windows XP y conectada a la red IP de prueba.

CAPÍTULO IV. RESULTADOS

En este capítulo se presenta la explicación de cada uno de los resultados de la investigación, como son, selección del codec, del protocolo de señalización y explicación de funcionamiento del teléfono.

IV.1 SELECCIÓN DE CODEC

Se hace un análisis comparativo de los codecs G711 y G729 para medir la complejidad de la implementación de los algoritmos.

IV.1.i G711 PCM

Para esta compresión se utilizó el código de Diciembre 1994 del Comité consultivo Internacional Telegrafico y Telefonico (CCITT), el cual permite la compresión de archivos de muestras de 16 bits a archivos en ley μ o ley A de 8 bits por muestra, y su correspondiente conversión de la compresión a un archivo lineal.

El archivo ejecutable G711.exe fue obtenido utilizando el compilador Dev-C++ versión 4.9.9.2

Procedimiento

Se obtuvo un archivo de voz de duración de 1,15 seg, con un tamaño de, IN.WAV (Formato de Audio de forma de Onda "Waveform Audio Fromat"), mediante el grabador de sonidos del sistema operativo Windows XP, con muestras a 8 Khz. y a 16 bits por muestra (monoaudio).

Este archivo fue modificado y se obtienen solo las muestras de voz en un archivo hexadecimal, IN.HEX de 18400 bytes. Este archivo sirvió como entrada al programa ejecutable de compresión y se obtuvo como salida el archivo comprimido en ley μ, CHANNELA.HEX, y en ley A, CHANNELA.HEX. Luego, a cada archivo

comprimido se le aplicó el proceso inverso y se obtuvieron OUTU.HEX y OUTA.HEX respectivamente.

Resultados

Se obtuvieron archivos comprimidos (CHANNELU.HEX y CHANNELA.HEX) con exactamente la mitad de los bytes que los archivos originales (9200 bytes). Al aplicar la descompresión de los archivos channelμ.hex y channelA.hex, los archivos OUTU.HEX y OUTA.HEX se diferenciaban del archivo de entrada original IN.HEX solamente por algunos bytes (ver Tabla Nro 5:, sólo los valores en el byte de menor orden cambian), pero el sonido grabado se vio poco afectado cuando se realizaron las modificaciones de los archivos de hexadecimal a Formato WAV para ser reproducidos.

	in.hex	outa.hex	outu.hex
Palabra en $0_{(16)}$	$FD5E_{(16)}$ = -674	$FD50_{(16)}$ = -684	$FD54_{(16)}$ = -688
Palabra en $60_{(16)}$	$0211_{(16)}$ = 529	$0210_{(16)}$ = 528	$020C_{(16)}$ = 524
Palabra en $A0_{(16)}$	$0364_{(16)}$ = 868	$0370_{(16)}$ = 880	$036C_{(16)}$ = 876

Tabla Nro 5: Comparación de archivo de entrada a codec G711 ley A y ley μ con los archivos de salida correspondientes
Fuente: Propia

Figura Nro 5: Archivo de entrada, In.hex
Fuente: Propia

0	50	FD	50	FD	30	FD	50	FD	70	FD	B0	FD	70	FD	90	FD
10	D0	FD	B0	FD	D0	FD	08	FE	38	FE	58	FE	78	FE	88	FE
20	D8	FE	18	FF	18	FF	68	FF	58	FF	48	FF	68	FF	68	FF
30	98	FF	88	FF	68	FF	B8	FF	C8	FF	D8	FF	E8	FF	28	00
40	28	00	48	00	88	00	98	00	B8	00	B8	00	E8	00	28	01
50	18	01	38	01	48	01	48	01	58	01	A8	01	E8	01	F8	01
60	10	02	10	02	E8	01	30	02	B0	02	B0	02	D0	02	10	03
70	30	03	70	03	B0	03	20	04	60	04	A0	04	A0	04	E0	04
80	E0	04	60	04	60	04	60	04	60	04	20	04	20	04	60	04
90	60	04	20	04	20	04	20	04	F0	03	F0	03	D0	03	90	03
A0	70	03	30	03	30	03	F0	02	F0	02	10	03	D0	02	F0	02
B0	F0	02	B0	02	B0	02	70	02	30	02	10	02	C8	01	88	01
C0	48	01	28	01	B8	00	A8	00	A8	00	88	00	68	00	28	00
D0	18	00	E8	FF	C8	FF	B8	FF	98	FF	78	FF	48	FF	28	FF
E0	E8	FE	E8	FE	E8	FE	B8	FE	B8	FE	98	FE	88	FE	78	FE
F0	38	FE	28	FE	38	FE	08	FE	F0	FD	D0	FD	B0	FD	90	FD

Figura Nro 6: Archivo de salida con ley A, outa.hex
Fuente: Propia

0	54	FD	34	FD	34	FD	54	FD	74	FD	B4	FD	74	FD	94	FD
10	B4	FD	B4	FD	D4	FD	14	FE	34	FE	54	FE	74	FE	74	FE
20	DC	FE	1C	FF	1C	FF	5C	FF	5C	FF	4C	FF	5C	FF	5C	FF
30	98	FF	7C	FF	6C	FF	B8	FF	C0	FF	D0	FF	E0	FF	28	00
40	30	00	48	00	84	00	94	00	B4	00	B4	00	F4	00	24	01
50	14	01	34	01	44	01	44	01	54	01	AC	01	EC	01	EC	01
60	0C	02	DC	02	EC	01	2C	02	AC	02	AC	02	EC	02	0C	03
70	2C	03	6C	03	DC	03	1C	04	5C	04	9C	04	9C	04	DC	04
80	DC	04	5C	04	5C	04	5C	04	5C	04	1C	04	1C	04	5C	04
90	5C	04	1C	04	1C	04	1C	04	DC	03	DC	03	DC	03	9C	03
A0	6C	03	2C	03	2C	03	EC	02	EC	02	0C	03	CC	02	0C	03
B0	EC	02	AC	02	AC	02	6C	02	2C	02	0C	02	CC	01	8C	01
C0	54	01	24	01	B4	00	B4	00	A4	00	84	00	60	00	20	00
D0	18	00	E8	FF	C0	FF	B8	FF	98	FF	7C	FF	3C	FF	1C	FF
E0	DC	FE	DC	FE	DC	FE	BC	FE	BC	FE	8C	FE	8C	FE	74	FE
F0	34	FE	34	FE	34	FE	14	FE	D4	FD	D4	FD	B4	FD	94	FD

Figura Nro 7: Archivo de salida con ley μ, outu.hex
Fuente: Propia

En las Figuras anteriores (Figura Nro 5:, Figura Nro 6: y Figura Nro 7:) se pueden observar los primeros 256 bytes del archivo de entrada y los dos archivos de salida. Si bien la diferencia entre los valores de las muestras no es muy grande, aún así existen, y sin embargo la calidad del sonido no disminuye notablemente.

Para medir la complejidad del codec, sin entrar en detalles técnicos, se puede observar que la implementación es relativamente sencilla en el código fuente utilizado, pues para realizar conversiones entre un valor de muestra y un valor

comprimido no es necesario más que búsquedas en tablas estáticas de conversión para lo cual basta una función, por ejemplo en el código de la UIT se utilizó la función linear2a y linear2u para las compresiones. Según Spanias en *Speech Coding,* la cantidad de instrucciones por segundo (MIPS) de un codificador PCM, es de 0.01 MIPS, lo cual es una baja cantidad de instrucciones comparado con otros codecs. La desventaja que poseen estas compresiones es que la cantidad de ancho de banda utilizado es de alrededor de 80 Kbps (ver Tabla Nro 6:) con envíos de muestras de voz 20 milisegundos, o 160 muestras por paquete, y es bastante alto en comparación a otros codecs de voz utilizados en telefonía IP.

	Tamaño (kbps)	Tamaño (bytes por paquete)
Muestras	64	160
Cabecera RTP	4,8	12
Cabecera UDP	3,2	8
Cabecera IP	8	20
Total	80	200

Tabla Nro 6: Ancho de banda utilizado usando el codec G711
Fuente: Propia

IV.1.ii G.729 CELP

La implementación fue obtenida a compilar códigos fuentes ANSI C de un simulador del estándar G729 anexo B desarrollado por la Universidad de Sherbrooke, Francia. Se utilizó un compilador Dev-C++ versión 4.9.9.2.

Se obtuvo un codificador CELP, CelpCoder.exe, y su respectivo decodificador, CelpDecod.exe.

El simulador acepta como entrada un archivo de muestras de voz de 16 bits en modo monoaudio. En las pruebas que se dieron con esta compresión se utilizó el archivo IN.HEX grabado a 8 Khz. con muestras de 16 bits cada una, utilizado para las pruebas de G.711.

Procedimiento

El archivo en formato WAV IN.WAV, fue formateado y se obtuvo un nuevo archivo que solo contenía la información de muestras de voz, IN.HEX. Este archivo se aplicó como entrada al programa CelpCoder.exe, y se obtuvo un archivo intermedio, CHANNEL.HEX, que representa las tramas de voz comprimida enviada a través del canal. Luego, el archivo intermedio se utilizó como entrada del decodificador CelpDecod.exe y se obtuvo un archivo final OUT.HEX que simula la descompresión de la voz, y son las muestras de voz que escucharía el receptor. Para que este archivo sea compatible con el sistema operativo y se pudiera escuchar sin nuevas herramientas, la cabecera de un archivo WAV fue colocado al principio del archivo OUT.HEX, creándose un nuevo archivo, OUT.WAV, que pudo ser reproducido fácilmente por el reproductor incorporado del sistema operativo, Reproductor de Windows Media.

Resultados

El archivo de entrada IN.HEX y de salida OUTCELP.HEX presentan diferencias visibles a nivel de valores hexadecimales, sobretodo al principio de las muestras de voz. Estas diferencias afectaron de manera más apreciable el resultado de la grabación, aunque la palabra pronunciada en ambos archivos es claramente audible.

La compresión del archivo de voz mediante el método CELP no produjo un menor tamaño en el archivo intermedio CHANNELCELP.HEX, pero esto no fue producido por el algoritmo de compresión sino por la manera en que fue implementado el programa, que para fines prácticos utilizó 2 bytes por cada 2 bits de código. El archivo de entrada IN.HEX mide 18400 bytes y el archivo CHANNELCELP.HEX mide 18860 bytes. Si tan solo se tomaran los bits de código y se acomodaran de tal manera en el archivo de forma de no desperdiciar espacio, se

obtendría un archivo CHANNELCELP.HEX de aproximadamente 2300 bytes, lo cual es casi 8 veces de menor tamaño que el original.

0	21	6B	50	00	81	00	81	00	81	00	7F	00	81	00	7F	00
10	7F	00	81	00	81	00	7F	00	7F	00	7F	00	81	00	7F	00
20	7F	00	7F	00	7F	00	81	00	81	00	7F	00	81	00	81	00
30	81	00	81	00	7F	00	81	00	7F	00	7F	00	7F	00	7F	00
40	7F	00	7F	00	7F	00	7F	00	7F	00	7F	00	7F	00	7F	00
50	7F	00	7F	00	81	00	81	00	81	00	81	00	81	00	7F	00
60	81	00	7F	00	81	00	81	00	7F	00	7F	00	7F	00	7F	00
70	7F	00	7F	00	7F	00	7F	00	7F	00	81	00	7F	00	81	00
80	81	00	7F	00	7F	00	7F	00	7F	00	7F	00	7F	00	81	00
90	81	00	7F	00	7F	00	7F	00	81	00	7F	00	81	00	81	00
A0	7F	00	81	00	21	6B	50	00	7F	00	81	00	81	00	7F	00
B0	81	00	7F	00	7F	00	81	00	81	00	7F	00	7F	00	7F	00
C0	81	00	7F	00	7F	00	81	00	7F	00	7F	00	81	00	81	00
D0	7F	00	7F	00	81	00	7F	00	7F	00	81	00	7F	00	81	00
E0	7F	00	81	00	81	00	7F	00	81	00	7F	00	81	00	81	00
F0	81	00	81	00	7F	00	81	00	7F	00	81	00	7F	00	7F	00

Figura Nro 8: Archivo codificado en CELP, ChannelCELP.hex
Fuente: Propia

0	02	00	04	00	03	00	03	00	01	00	FF	FF	FF	FF	FF	FF
10	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
20	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	00	00	00	00
30	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
40	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
50	DA	FF	B7	FF	B7	FF	B4	FF	D8	FF	D8	FF	E1	FF	E9	FF
60	F7	FF	FF	FF	07	00	0E	00	12	00	19	00	1E	00	1F	00
70	20	00	48	00	43	00	4B	00	4C	00	49	00	44	00	41	00
80	3A	00	2C	00	28	00	21	00	16	00	10	00	09	00	02	00
90	FD	FF	F4	FF	F2	FF	EF	FF	ED	FF	EC	FF	EC	FF	EC	FF
A0	E9	FF	E9	FF	E8	FF	E9	FF	EA	FF	EA	FF	EB	FF	EC	FF
B0	ED	FF	EF	FF	F0	FF	F1	FF	4E	00	4A	00	50	00	51	00
C0	4B	00	45	00	41	00	37	00	21	00	1E	00	0F	00	05	00
D0	F7	FF	95	FF	8D	FF	82	FF	78	FF	1D	FF	22	FF	25	FF
E0	25	FF	3C	FF	48	FF	5C	FF	17	FF	3D	FF	4A	FF	60	FF
F0	3D	FF	0A	FF	2F	FF	44	FF	6B	FF	86	FF	A3	FF	CA	FF

Figura Nro 9: Archivo de salida, OutCelp.hex
Fuente: Propia

En la Figura Nro 8: se puede apreciar la manera en que se almacena la información de codificación. Los únicos valores de codificación presentes son 0x81 y 0x7F, además de valores de inicio de trama y tamaño.

En la Figura Nro 9: se observa la salida del decodificador a partir de la entrada mostrada en la Figura Nro 8:, y comparándolo con la Figura Nro 5:, en la cual vemos el archivo de entrada, notamos varias diferencias en los archivos (ver Tabla Nro 7:), prácticamente todas las muestras de voz se han modificado, pero como se comprobó cuando se escucho la grabación, la calidad del sonido no desmejoró en gran medida.

	in.hex	outcelp.hex
Palabra en $0_{(16)}$	$FD5E_{(16)}$ = -674	$0002_{(16)}$ = -684
Palabra en $60_{(16)}$	$0211_{(16)}$ = 529	$FFF7_{(16)}$ = 528
Palabra en $A0_{(16)}$	$0364_{(16)}$ = 868	$FFE9_{(16)}$ = 880

Tabla Nro 7: Comparación de archivo de entrada al codec G729 con el archivo de salida correspondientes
Fuente: Propia

La cantidad de MIPS utilizado por un algoritmo CS-CELP es, según Spanias, (1994), de unos 16 MIPS, unas 1600 veces más que lo utilizado por el codec G711, pero utiliza 39,3 Kbps (ver Tabla Nro 8) de ancho de banda en una red Ethernet, lo que resulta en unas 3 veces menos que la recomendación G711.

Desde el punto de vista de implementación del programa este codec es bastante complejo, puesto que se necesitan varios pasos antes de obtener un código, lo que introduce un retardo en el envío de los datos.

	Tamaño (kbps)	Tamaño (bytes por paquete)
Muestras	8	20
Cabecera RTP	4,8	12
Cabecera UDP	3,2	8
Cabecera IP	8	20
Total	24	60

Tabla Nro 8: Ancho de banda utilizado usando el codec G729
Fuente: Propia

El estándar G711 es el más sencillo de realizar, y cumple con los requisitos de reducir la cantidad de información de voz enviada a la mitad, ya sea con la ley de compresión μ o con la ley A. Por otro lado, para redes en donde el ancho de banda no es suficiente para contener la carga que implica el codec G711, es preferible la utilización del codec G.729.

IV.2 SELECCIÓN DEL PROTOCOLO DE SEÑALIZACIÓN

De la investigación realizada y referida en el marco teórico junto con las características mínimas (explicadas en el Anexo B para SIP y en el Anexo A para el conjunto de protocolos H323) que debe tener un teléfono IP utilizando cualquiera de estas tecnologías, se puede decir que en general para realizar una llamada de Voz sobre IP se necesitan las siguientes etapas.

- Registro de usuarios: Los terminales necesitan registrase ya sea en un Gatekeeper si es H.323 o en un servidor de registro para terminales SIP.
- Establecimiento de la conexión: El Terminal inicia la llamada utilizando mensajes de INVITE, TRYING y RINGING si se utiliza el protocolo SIP o CALLPRO, CEEDING y ALERTING cuando se trata del protocolo de señalización H.323.
- Negociación de parámetros: los participantes negocian los parámetros de intercambio de datos en tiempo real si es H323 (es el protocolo H245 el cual se encarga de establecer los parámetros de capacidades para la comunicación). En el caso del protocolo SIP, solo ocurre un intercambio de características del los equipos mediante el protocolo de Descripción de Sesión "Session Descrription Protocol" (SDP).
- Comunicación: En este aspecto no se presenta muchas diferencias porque tanto SIP como H323 utilizan el protocolo de transporte de datos en tiempo real RTP y su protocolo de control RTCP.

- Finalización de la llamada: Uno de los participantes inicia esta fase con un mensaje de BYE, y su respectivo reconocimiento con un mensaje 200 (para el conjunto SIP) o ENDSESSIONCOMMAND, RELEASE COMPLETE (para el conjunto H.323).

Pero existen algunas diferencias en las implementaciones de los protocolos H.323 y SIP como son las siguientes:

- Evaluando la complejidad de ambos protocolos se puede decir que el conjunto de protocolos H323 es más complicado que el protocolo SIP, dado que presenta procesos más complejos de codificación y decodificación de los paquetes debido a que utilizan un formato binario, frente a los paquetes SIP que manejan formato en texto. Además el amplio conjunto de protocolos que incorpora H.323 repercute negativamente en los tiempos de respuesta, así como en el tamaño del código que tienen que incorporar los equipos H.323.

- H323 se apoya en diversos protocolos los cuales hacen que cualquier modificación de H323 requiera de mayor complejidad debido a la dependencia que existe, mientras que en SIP puede interactuar con cualquier protocolo que le brinde calidad de servicio, acceso de directorio etc. sin modificar el protocolo SIP.

- SIP no soporta protocolos de vídeo como T.120 y no tiene ningún protocolo para control de la conferencia. Mientras que H.323 suporta todo tipo de conferencia de vídeo y datos. Los procedimientos permiten control de la conferencia y sincronización de los streams de audio y vídeo.

- H.323 cubre casi todos los servicios como capacidad de intercambio, control de conferencia, señalización básica, calidad de servicio, registro, servicio de descubrimiento y más. Mientras que SIP es modular y cubre la señalización básica, la localización de usuarios y el registro. Otras características se implementan en protocolos separados.

Una vez analizados todos estos aspectos, y tomando en cuenta principalmente la sencillez del protocolo tanto para su implementación como para ser observado en funcionamiento, se tomó la decisión de utilizar el protocolo SIP como el protocolo principal de aplicación del teléfono a implementar.

IV.3 DISEÑO DE LA INTERFAZ DE CONVERSOR

El conversor digital/analógico y analógico/digital se encuentra esquematizado en la siguiente Figura

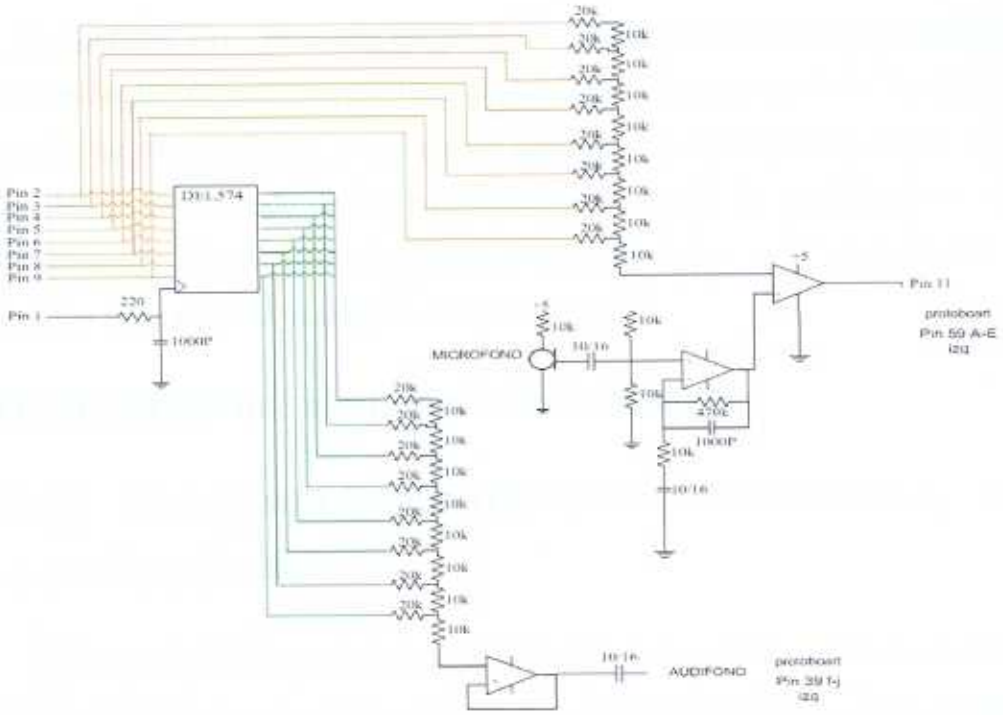

Figura Nro 10: Conversores digital/analógico y analógico/digital
Fuente: Propia.

El circuito de la Figura Nro 10: utiliza dos conversores de R-2R, uno para transformar las muestras de voz digitales a analógicas y luego ser enviadas al audífono por el cual el usuario escucha, el otro conversor se utiliza para poder realizar

audífono por el cual el usuario escucha, el otro conversor se utiliza para poder realizar comparaciones entre una señal previamente definida y la señal analógica que se desea muestrear (voz).

Las señales digitales se envían mediante el puerto paralelo por los pines dos al nueve al circuito de muestreo y retención "sample and hold" 547, luego se envía un borde de bajada por el pin 1 del puerto paralelo hacia el pin de reloj del integrado 547 para que este lea los bits en sus pines de entrada y los mantenga por sus pines de salida, permitiendo al conversor R-2R digital/analógico convertir la señal y mantenerla hasta la siguiente muestra.

Las señales analógicas son recibidas en un comparador analógico, junto con un valor analógico de muestras digitales enviadas por el puerto paralelo, la salida del comparador se conecta al pin 11 del puerto paralelo, y a cada muestra digital que se saca se obtiene el resultado de la comparación por este pin, de manera que se logra una codificación de bit a bit; luego de 8 comparaciones (una por cada bit) la codificación de la muestra analógica estará completa.

IV.4 DESCRIPCIÓN DEL DISPOSITIVO IMPLEMENTADO

IV.4.i PROGRAMAS RESIDENTES PARA MANEJO DE INTERFAZ CON LA TARJETA DE RED

Para el manejo de la interfaz de la tarjeta de red y las funciones de la librería WATTCP, se utilizan programas residentes que manejan la interrupción de hardware de la tarjeta ethernet. Este programa o programas residentes vienen incluidos en los archivos de configuración de las tarjetas de red y poseen el nombre de Interfaces Abiertas de Datos "Open Datalink Interface" (ODI), conocidas como paquete ODI. Una vez ejecutado estos programas, se ejecuta un programa residente propio encargado de manejar una interrupción cada 125 μseg utilizada para muestrear y

codificar la señal analógica y también para enviar las muestras de voz digitalizadas al conversor analógico digital.

Para el caso del 486 embebido se ejecutan los programas LSL.COM, LAN9000.COM y ODIPKT.COM, para tarjetas de redes en equipos de computador personal los programas residentes dependen de la tarjeta de red.

IV.4.ii PROGRAMA RESIDENTE DE CONVERSIÓN ANALÓGICO/DIGITAL Y DIGITAL/ANALÓGICO:

El programa se carga en memoria luego del paquete ODI. Su función principal es la de codificar en 8 bits las muestras analógicas de voz, y de enviar las muestras codificadas en 8 bits por el puerto paralelo para su posterior conversión a analógico por la interfaz del conversor (ver, interfaz del conversor). Cada muestra codificada en digital se guarda en un arreglo (IBuf) destinado para ello, y cada muestra que se vaya a convertir de digital a analógico se obtiene del arreglo OBuf.

La interrupción utilizada para esta tarea es la Petición de Interrupción cero "Interrupt Request 0"(IRQ0) o interrupción $8_{(16)}$. Esta interrupción se ejecuta periódicamente cada 55 mseg, aproximadamente (18,2 veces por segundo). Para que esta se ejecute cada 125 µseg (8000 veces por segundo), se modifica el valor en el puerto $40_{(16)}$ el cual usualmente es de 65536 (identificado con un valor de cero - 0). Los relojes del procesador trabajan a una velocidad de 1,193181 MHz., que dividido entre un valor de 65536 da como resultado 18,2 Hz. Modificando este divisor a través del puerto $40_{(16)}$ se puede obtener la interrupción cada 125 µseg.como se muestra en la Ecuación 7

$$f = \frac{1193181Hz}{65536} = 18{,}2Hz. \rightarrow T = \frac{1}{18{,}2Hz} = 54{,}9mseg.$$
$$f = \frac{1193181Hz}{149} = 8008Hz. \rightarrow T = \frac{1}{8008Hz} = 124{,}8\mu seg \qquad (7)$$

Dentro de la programación de la interrupción y para mantener a la IRQ0 original en funcionamiento, se utiliza un contador que luego de una cuenta de 440 veces, ejecuta a la interrupción $8_{(16)}$ original; y para mantener un contador de un milisegundo, cada 8 interrupciones de 125 μseg, incrementa un contador llamado Timer. Como se muestra en la Ecuación 8.

$$\frac{55mseg}{125\mu seg} = 440$$
$$\frac{1mseg}{125\mu seg} = 8 \qquad (8)$$

Finalmente para permitir la comunicación de datos con programas externos se reprograman tres interrupciones de software, las interrupciones $80_{(16)}$, $81_{(16)}$ y la $82(_{16})$. El puntero en el vector de interrupciones hacia la interrupción $80_{(16)}$ apunta hacia el inicio de datos de la interrupción $8_{(16)}$ de manera que los arreglos IBuf, OBuf y el contador de milisegundos Timer sean direccionables desde afuera del código de la interrupción. Las interrupciones $81_{(16)}$ y $82_{(16)}$, se reprograman con solo la instrucción de retorno de interrupción, pero son llamadas periódicamente por la interrupción $8_{(16)}$.

La interrupción $81_{(16)}$ es ejecutada cuando el arreglo de almacenamiento de muestras analógicas codificadas, IBuf, se llena, indicando que pueden ser utilizadas todas las muestras digitales de voz almacenadas. Por otro lado la interrupción $8_{(16)}$ empezará a utilizar temporalmente otro arreglo y avisará mediante la misma interrupción $81_{(16)}$ cuando este nuevo arreglo este completamente lleno para su uso. Cuando la interrupción $8_{(16)}$ ejecuta la interrupción $81_{(16)}$ se pasan los parámetros del inicio del arreglo que puede ser usado en los registros de maquina AX y DX, conteniendo DX el segmento de memoria y AX el desplazamiento.

La interrupción $82_{(16)}$ se ejecuta cada ves que el arreglo de salida, OBuf, se vacía indicando que puede ser utilizado para guardar los siguientes datos obtenidos por el protocolo RTP que vayan a ser enviados a la interfaz del conversor,

temporalmente la interrupción $8_{(16)}$ utiliza otro arreglo para sacar las muestras digitales de voz al puerto paralelo. Su funcionamiento es muy parecido a la interrupción $82_{(16)}$, el puntero de inicio del arreglo es enviado a través de los registros AX y DX, siendo DX el segmento de memoria y AX el desplazamiento.

IV.4.iii LIBRERÍA RTP

El módulo del programa RTP, brinda la interfaz para formar los paquetes con el formato RTP especificado en el marco teórico (Ver punto II.5.i Paquetes RTP); en esta implementación los parámetros de la cabecera son mantenidos constantes exceptuando por el número de secuencia, puesto que este se utiliza en la recepción para ordenar los paquetes.

El programa tiene dos funciones principales, una para recibir los paquetes RTP, y otra para generarlos; posee también una función que devuelve el estado de RTP en la comunicación, si esta se encuentra activa o inactiva, una función para inicializar al módulo RTP, y una para finalizarlo.

La función de recepción, recibir_pack_rtp, encargada de buscar dentro de la cabecera del paquete RTP el número de secuencia, utilizado para ordenar los paquetes dentro del buffer de la interrupción $8_{(16)}$ OBuf y de copiar los valores de carga del paquete en su respectiva posición dentro de OBuf. En caso de que se reciba el primer paquete RTP de una sesión, cualquier número de secuencia es valido; a partir del segundo paquete RTP recibido, la función mantiene y espera un número de secuencia, si el paquete recibido posee un número de secuencia menor al esperado se descarta, por el contrario si el número de secuencia esperado es mayor se busca la posición en OBuf donde este debe ser copiado, y se actualiza el número de secuencia a esperar con el valor del número de secuencia recibido más uno.

La función de generación de paquete RTP, es la encargada de copiar la cabecera del paquete RTP y obtener las siguientes muestras que deben ser enviadas (armar el paquete RTP). Al copiar la cabecera RTP, se coloca el número de secuencia

correspondiente y se incrementa en una unidad (Esquema del paquete RTP se puede observar en la Figura Nro 2:). La carga de muestras de voz para el paquete RTP se obtiene del arreglo IBuf y se copian al arreglo de RTP, para su posterior envío por la red.

La función rtp_estado() devuelve un valor de cero si no se esperan enviar o recibir paquetes RTP, y un valor de uno si se esperan recibir y enviar paquetes RTP.

Las funciones rtp_init() y rtp_fin() se utilizan para iniciar y finalizar, respectivamente, la recepción y envío de paquetes.

IV.4.iv LIBRERÍA SIP

El módulo de SIP genera las peticiones de REGISTER, INVITE, ACK, y BYE. Además los mensajes informativos 180 repicando, 200 OK, 486 Ocupado. Posee varias funciones principales: una para recepción de paquetes y otra para la generación de cada tipo de paquete, además posee una función de estado del módulo SIP, la cual indica los diferentes estados del dispositivo durante la llamada (Espera, En Llamada, Llamando, Registrando, Procesando, Repicando), una función de inicialización y una función de revisión.

La función de recepción de paquete SIP, se encarga de determinar el paquete SIP recibido, y si este requiere de alguna respuesta, esta es generada automáticamente para luego ser enviada.

Para cada petición SIP implementada (REGISTER, BYE, ACK, INVITE) se creó una función de generación del paquete. Y para los mensajes de respuesta 180, 200, y 486 se crearon funciones de respuesta internas al modulo, puesto que estas respuestas son generalmente automáticas, sólo la respuesta 200 posee una función de interfaz que permite ser generada cuando se atiende la llamada.

La función de revisión, sip_check(), revisa si el timer de SIP se ha agotado y en caso de que haya sucedido reinicia el teléfono. La función de inicialización genera un mensaje de REGISTER y coloca el estado del teléfono en espera luego de 3 segundos.

IV.4.v FUNCIONAMIENTO GENERAL

El teléfono está operativo cuando los programas residentes de manejo de interfaz de la tarjeta de red están en funcionamiento, la interrupción cada 125 μseg ha sido también cargada como programa residente y un programa ejecutable utiliza los módulos de SIP y RTP para señalizar la llamada e intercambiar paquetes de voz, así como reprogramar a las interrupciones $81_{(16)}$ y $82_{(16)}$ para obtener el cambio de los punteros a los arreglos IBuf y OBuf.

La función principal, inicializa las interrupciones $81_{(16)}$ y $82_{(16)}$ para obtener los punteros a los arreglos IBuf y OBuf, inicializa los sockets de wattcp e inicia el socket a ser utilizado por SIP, inicia el módulo SIP y luego entra en un ciclo infinito en el cual se estará constantemente revisando el estado de los sockets con la función de wattcp tcp_tick, y revisando el estado de los módulos RTP y SIP.

Dentro de este ciclo infinito, se revisa constantemente si llega un paquete SIP y, si el estado de RTP es activo, se revisa por la llegada de paquetes RTP y si es necesario enviar un paquete RTP este se genera y se envía; luego dependiendo del estado de SIP, se toman ciertas decisiones de señalización. Si se presiona un botón, leído por el puerto paralelo, dependiendo del estado del teléfono esto generará o una petición INVITE, o una petición BYE, o generará un mensaje informativo "200 ok".

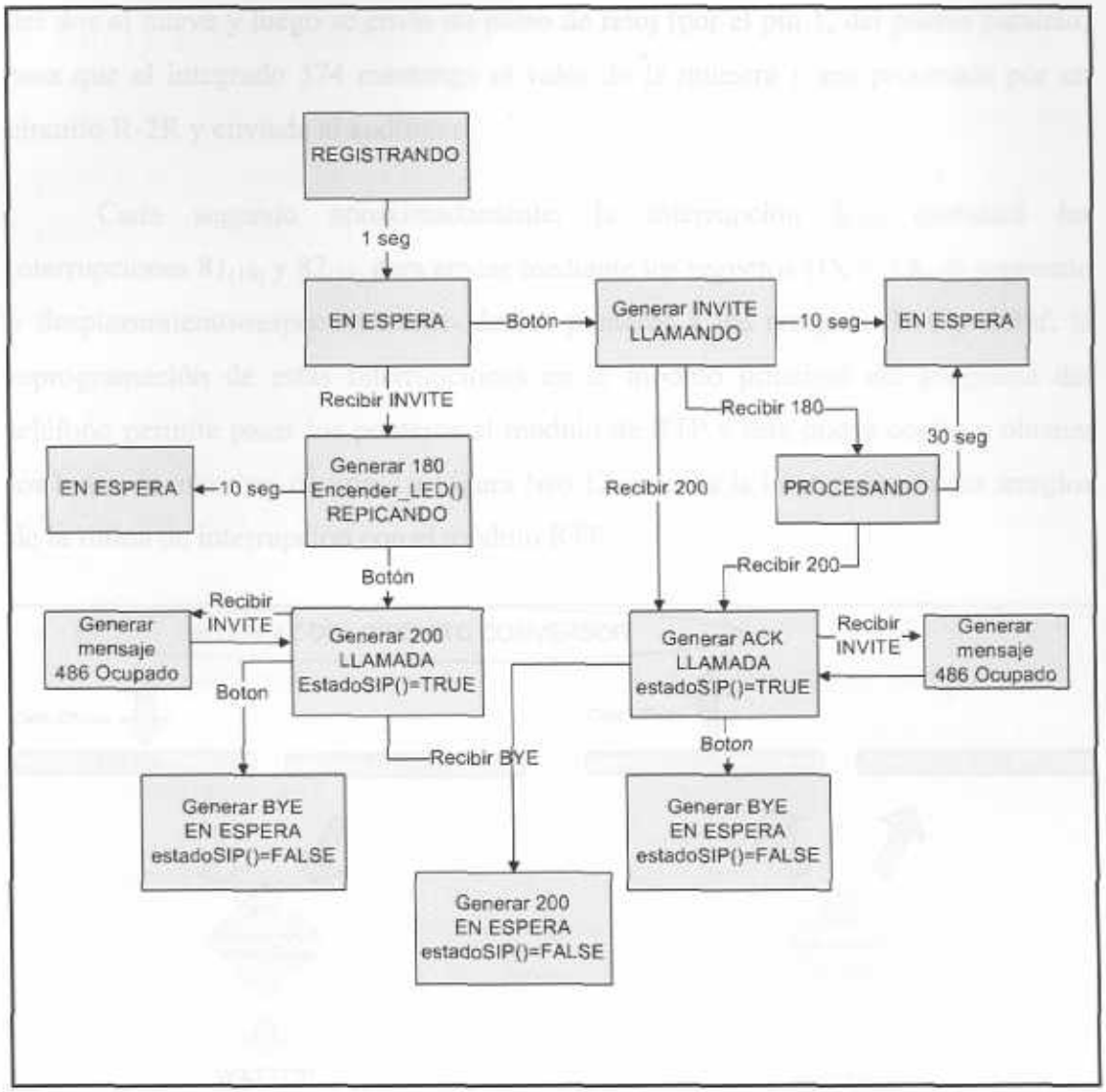

Figura Nro 11: Diagrama de estados *SIP*
Fuente: Propia.

Las muestras analógicas, por su parte, están siendo codificadas cada 125 μseg, utilizando la interrupción $8_{(16)}$ que envía por el puerto paralelo ocho distintos valores digitales, y utilizando un comparador con una entrada, señal analógica (voz) y la otra el valor analógico de la señal enviada por puerto paralelo entre los pines dos al nueve, y cuya salida se conecta con el pin 11(del puerto paralelo) de donde son leídos los resultados de la comparación y se obtiene la codificación de la muestra bit a bit. El valor digital que se obtenga del arreglo OBuf se envía al puerto paralelo por los pines

del dos al nueve y luego se envía un pulso de reloj (por el pin 1, del puerto paralelo) para que el integrado 574 mantenga el valor de la muestra y sea procesada por un circuito R-2R y enviada al audífono.

Cada segundo aproximadamente, la interrupción $8_{(16)}$ ejecutará las interrupciones $81_{(16)}$ y $82_{(16)}$ para enviar mediante los registros DX y AX, el segmento y desplazamiento-respectivamente- de los punteros a los arreglos IBuf y OBuf; la reprogramación de estas interrupciones en el módulo principal del programa del teléfono permite pasar los punteros al módulo de RTP y este puede copiar y obtener los bytes de muestras de voz. La Figura Nro 12 muestra la interacción de los arreglos de la rutina de interrupción con el módulo RTP.

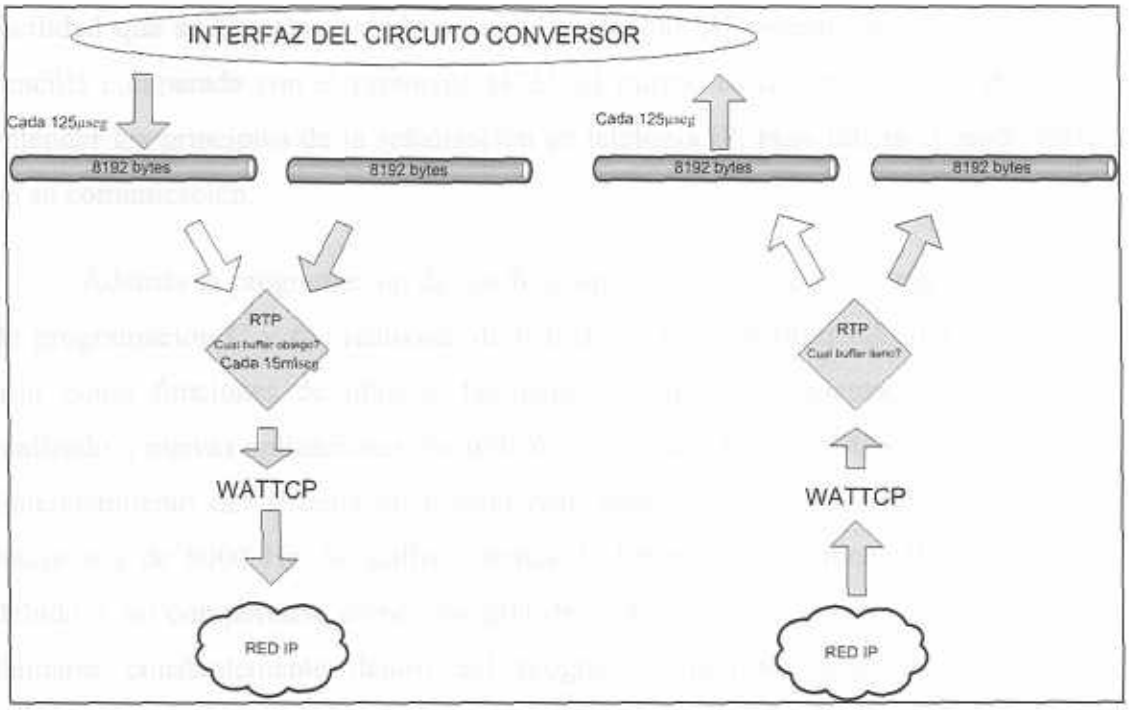

Figura Nro 12: Esquema de arreglos IBuf y OBuf y su interacción con el módulo RTP
Fuente: Propia

CAPÍTULO V. CONCLUSIONES Y RECOMENDACIONES

En este último capítulo se extraen las conclusiones de la realización del proyecto y se proponen líneas abiertas a la investigación y mejoras a los servicios ofrecidos por el teléfono.

V.1 CONCLUSIONES Y RECOMENDACIONES

En este trabajo se ha diseñado e implementado un Prototipo de Teléfono IP con capacidades de crear y recibir llamadas, utilizando los requerimientos mínimos necesarios para que se establezca la comunicación. La utilización de los protocolos aplicados hace posible el desarrollo de este proyecto, ofreciendo la flexibilidad y facilidad que se buscaba, debido a que el protocolo SIP permite una programación sencilla comparada con el protocolo H323, el mismo es fácilmente mostrable para entender los principios de la señalización en telefonía IP, pues utiliza el modo texto en su comunicación.

Además la programación de sus funciones se realizó con el conocido lenguaje de programación C y fue realizada de manera modular permitiendo que se puedan usar como funciones de librería, facilitando el manejo y adaptación del código realizado a nuevas aplicaciones. Se utilizó el sistema operativo DOS que permitió el funcionamiento del sistema en tiempo real, permitiendo realizar muestreos a una frecuencia de 8000 Hz. Se utilizó además, la librería de funciones WATTCP, que debido a no comportarse como una pila de protocolos sino como funciones deben llamarse constantemente dentro del programa principal, consumen tiempo de procesamiento y no permiten una fácil portabilidad a otros sistemas.

Se recomienda la continuidad del proyecto bajo el sistema operativo Linux, puesto que a diferencia de las funciones de la librería WATTCP, Linux permite una pila de protocolos trabajando en conjunto, y agrega la ventaja de capacidades multitarea, no presente en DOS; también brinda una cantidad de protocolos ya

implementados y utilizados en tecnologías VoIP. Se recomienda también el rediseño del conversor A/D y D/A, que permita mayor número de bits de codificación y agregue la compasión y descompansión. Este nuevo sistema de conversor de señal, debe realizarse separado del microprocesador o microcontrolador principal evitando que consuma tiempo de procesamiento para el análisis y envío de paquetes. Un microprocesador con audio integrado es sugerido.

El protocolo RTP es parte importante de la telefonía IP, puesto que los conjuntos de protocolos más significativos de VoIP – H323 y SIP – utilizan a este protocolo como el medio para el envío y recepción de datos en tiempo real; es un protocolo sencillo, con suficiente información en la cabecera para cumplir su función de sincronización y ordenamiento de los datos, además se puede expandir, usando el bit de expansión, y agregando datos en una segunda cabecera.

Los codecs, permiten la compresión de la voz y el ahorro de ancho de banda en la transmisión, pero degradan ligeramente la calidad de voz. Su uso puede atarse a las preferencias de los usuarios y al ancho de banda disponible, sobre todo si el ancho de banda de la red IP es limitado, debido a que junto a las diferentes cabeceras de los protocolos utilizados, se obtiene el ancho de banda necesario para que se de la comunicación. El estudio de los codec de este trabajo está realizado específicamente sobre los mas utilizados G729, G711, aunque por los motivos ya explicados en la limitación, no fue aplicado ningún tipo de codec en este proyecto, pero los programas de los codec que fueron utilizados para las distintas pruebas funcionan perfectamente y pueden ser usados para hacer estudios prácticos como también, futuras mejoras del dispositivo.

El estudio de los protocolos de los sistemas, se llevó a cabo utilizando los documentos RFC de la IETF, recomendaciones de la UIT y otras bibliografías varias. Esta aproximación consumió gran cantidad de tiempo de estudio; se sugiere realizar una aproximación práctica, utilizando aparatos comerciales y analizadores de protocolos como Ethernet para entender el funcionamiento de la comunicación.

Se abre la posibilidad de desarrollar el teléfono IP en un microcontrolador para que éste pueda ser implementado como un teléfono de muestra en asignaturas de la carrera de Ingeniería en Telecomunicaciones. Así como también el uso de los programas resultantes en actividades de laboratorio para demostrar el funcionamiento de un teléfono IP.

Es importante señalar que todos los componentes utilizados en el teléfono, corresponden a las recomendaciones existentes y al resultado del análisis aplicado en el proyecto, igual para la descripción de sus principales funciones.

V.2 BIBLIOGRAFÍA

- Davidson, J. y Peter, J. (s.f.). Fundamentos de Voz Sobre IP. Madrid: Pearson Education.
- Engelke, E. (2003). Wattcp Network Programming Library. (www.wattcp.com)
- Johnston, A (2004). Understanding the Session Initiation Protocol. Artech House
- Khasnabish, B. (2003). Implementing Voice over IP. New Jersey: Jhon Wiley & Sons
- Perkins, C. (2003). RTP – Audio and Video for Internet. Boston: Pearson Education.
- Rosenberg, J., Shulzrine, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., et al (2002). RFC 3261 – SIP Session Initiation Protocol. Extraído en Febrero de 2006 de ftp://ftp.rfc-editor.org/in-notes/rfc3261.txt
- Schulzrine, H., Casrner, S., Frederick, R. y Jacobson, V. (2003). RFC 3550 – RTP: A Transport Protocol for Real-Time Applications. Extraído en Febrero de 2006 de ftp://ftp.rfc-editor.org/in-notes/rfc3550.txt
- Schulzrine, H., Casrner, S.(2003). RFC 3551 – RTP Profile for Audio and Video Conferences with Minimal Control. Extraído en Febrero de 2006 de ftp://ftp.rfc-editor.org/in-notes/rfc3551.txt
- Spanias, A. S. (1994). Speech coding: a tutorial review. Porciones publicadas en Proceedings revista técnica del IEEE, Oct. 1994.

- UIT-T (1990). Modulación por impulsos codificados diferencial adaptativa (MICDA), recomendación UIT-T G729.

- UIT-T (1996). Codificación de la Voz a 8 kbit/s mediante predicción lineal con excitación por código algebraico de estructura conjugada, recomendación UIT-T G729. Ginebra: International Telecommunication Union.

- UIT-T (1999). Modulación por impulsos codificados (MIC), recomendación UIT-T G711.

- UIT- T Users' Group on Software Tools (2005). ITU-T Software Tool Library 2005 User's Manual. Ginebra: International Telecommunication Union.

ANEXO A. ANÁLISIS DE LA IMPLEMENTACIÓN MÍNIMA DEL CONJUNTO DE PROTOCOLOS H323

ANÁLISIS DE LA IMPLEMENTACIÓN MÍNIMA DEL CONJUNTO DE PROTOCOLOS H323

La implementación mínima de H.323 requerida para el proyecto "Diseño de un teléfono IP" definida según la UIT-T se presenta a continuación:

A.1 CAPACIDADES DE MEDIOS

- Capacidad de voz : G.711 (ley A y ley μ);
- Capacidad DTMF, transmisión obligatoria de mensajes de información H.225.0;
- Distribución de medios: soporte de unidifusión

A.2 CAPACIDADES DE CONTROL

Secuencia de conexión rápida de la Rec. UIT-T H.323 (1998), Con el procedimiento de conexión rápida los puntos extremos pueden establecer una comunicación punto a punto básica con apenas un sólo intercambio de mensajes de ida y vuelta, lo que permite una entrega inmediata de trenes de medios al efectuar la conexión de la llamada.

El punto extremo llamante inicia el procedimiento de conexión rápida enviando un mensaje de establecimiento que contiene el elemento arranque rápido (*fastStart*) al punto extremo llamado. El elemento *fastStart* consiste en una secuencia de estructuras OpenLogicalChannel que describen los canales de medios que el punto extremo llamante propone enviar y recibir, incluidos todos los parámetros necesarios para abrir inmediatamente e iniciar la transferencia de medios por los canales.

El rechazo del procedimiento de conexión rápida se lleva a cabo bien mediante la no devolución del elemento *fastStart* o incluyendo el elemento *fastConnectRefuse*. Rechazar el procedimiento de conexión rápida (o no iniciarlo) requiere que se utilicen los procedimientos H.245 para intercambiar capacidades y abrir canales de medios.

- Señalización de registro, admisión y estado (RAS)

Para la señalización RAS del dispositivo SET audio (simple audio endpoint type) se necesitan los procedimientos RAS explicados en la referencia UIT-T H323, Los dispositivos SET deberán soportar como mínimo los siguientes mensajes RAS: transmisión de GRQ, RRQ, URQ, UCF y XRS y recepción de GCF, GRJ, RCF, RRJ, URQ, UCF, URJ y XRS.

Petición de controlador de acceso (GRQ, *gatekeeperRequest*)
UCF Confirmación de desregistro (*unregistration confirm*)
Mensaje no entendido (XRS)
Confirmación de controlador de acceso (GCF, *gatekeeperConfirm*)
Rechazo de controlador de acceso (GRJ, *gatekeeperReject*)
Petición de registro (RRQ, *registrationRequest*)
Confirmación de registro (RCF, *registrationConfirm*)
Rechazo de registro (RRJ, *registrationReject*)
Petición de desregistro (URQ, *unregistrationRequest*)
Confirmación de desregistro (UCF, *unregistrationConfirm*)
Rechazo de desregistro (URJ, *unregistrationReject*)

- Señalización de la llamada (control de la llamada H.225.0)

Maneja los mismos procedimientos del control de llamada, pero una vez abierto el canal de señalización de la llamada, luego de una comunicación, el dispositivo no debe cerrar el canal; además, debe usar el procedimiento de conexión rápida al originar una llamada, manejar los mensajes de información de H.225 en el canal de señalización de llamada y los mensajes de consulta de estado para estimar tiempos de ida y vuelta con su par.

Los dispositivos SET deberán atenerse a los procedimientos de control de la llamada definidos en las Recomendaciones UIT-T H.323 (1998) y H.225.0 (1998). Los dispositivos SET no cerrarán el canal de señalización de la llamada después del establecimiento de la comunicación.

- Señalización de control de sistemas multimedios (H.245)

Se utilizará el procedimiento de conexión rápida para el establecimiento de la conexión y se mantendrá la transmisión repetida del elemento fastStart en los mensajes de señalización de llamada H.225 para reconfigurar o reencaminar trenes de medios. Los dispositivos SET no abrirán una conexión H.245 independiente sino que van a utilizar la estructura de apertura de canal lógico en la secuencia de conexión rápida junto con la determinación implícita principal subordinado. El Principal subordinado se utiliza para evitar conflictos entre dos puntos extremos que están intentando abrir un canal bidireccional, el procedimiento consiste en intercambiar números aleatorios en el mensaje determinación principal-subordinado "masterSlaveDetermination", para determinar cuál es el principal y cuál es el subordinado. Este dispositivo tiene que ser capaz de funcionar como subordinado o como principal.

- Intercambio de capacidad

El procedimiento de intercambio de capacidad para tipos de medios y modos de transmisión se llevará a cabo siguiendo las reglas del procedimiento de conexión rápida, utilizando varias estructuras de apertura de canal lógico como parte de las posibilidades ofrecidas del llamante, la parte llamada elige sus capacidades para enviar y recibir.

En las siguientes líneas se enumeran las capacidades que debe tener la parte receptora para que comprenda y envié información.

A.3 CAPACIDAD DE AUDIO

- G.711 (ley μ, ley A, 56 kbit/s, 64 kbit/s)

A.4 MENSAJES DE SEÑALIZACIÓN DE CANAL LÓGICO

La apertura de canales lógicos se atendrá a las especificaciones de conexión rápida.Los dispositivos SET deberán soportar la transmisión de multifrecuencia bitono (DTMF), como elementos de información en la conexión de señalización de llamada H.225.0 (por ejemplo, utilizando mensajes de información).

Para este dispositivo, las capacidades de audio se indicarán únicamente en la aplicación del procedimiento de conexión rápida y en el intercambio repetido de estructuras de apertura de canal lógico "OpenLogicalChannel" utilizando la conexión rápida. Las capacidades indicadas por el lado remoto que no se comprendan serán ignoradas

Cuando se reciban, las estructuras de apertura de canal lógico se interpretarán de la forma siguiente:

- Si el número de canal lógico coincide con un canal lógico abierto en esos momentos, el canal correspondiente será reconfigurado conforme a los principios del procedimiento de conexión rápida si el componente tipo de datos no es "nulo" ("null").

- Si el componente tipo de datos es "nulo", lo que indica un "canal nulo" "NullChannel", el canal lógico correspondiente se considerará cerrado y cesará la transmisión de medios por el mismo.

- Si el número de canal lógico no coincide con un número de un canal lógico abierto en esos momentos, se abrirá un nuevo canal siguiendo los principios del procedimiento de conexión rápida.

Para el intercambio de medios, los dispositivos SET deberán seguir los procedimientos H.323 y H.225 utilizando RTP/UDP/IP para transportar los trenes de medios. Se utilizarán los formatos de paquetización de medios apropiados.

Las estructuras de **apertura de canal lógico** son utilizadas durante el procedimiento de conexión rápida para indicar sus capacidades y abrir canales de medios simultáneamente en ambos sentidos y reconfigurar trenes de medios durante una conferencia. Por definición, las estructuras de **apertura de canal lógico** contienen solamente parámetros de canal lógico hacia adelante o parámetros de canal lógico hacia atrás.

ANEXO B IMPLEMENTACIÓN MÍNIMA DEL PROTOCOLO SIP

SIP IMPLEMENTACIÓN MÍNIMA

B.1 PROTOCOLOS MÍNIMOS NECESARIOS

Para una implementación SIP mínima, se requieren de los protocolos SIP y Protocolo de descripción de la sesión "Session Description Protocol" (SDP) que acompaña a éste para el intercambio de información media, el protocolo de trasporte en tiempo real "Real Time Transport Protocol" (RTP) y RTP control (RTPC) utilizados para el envío de información, paquetes de voz en nuestro caso, por último son necesarios los protocolos TCP y UDP sobre los cuales se sustentan, para lograr el envío de la información sobre la red.

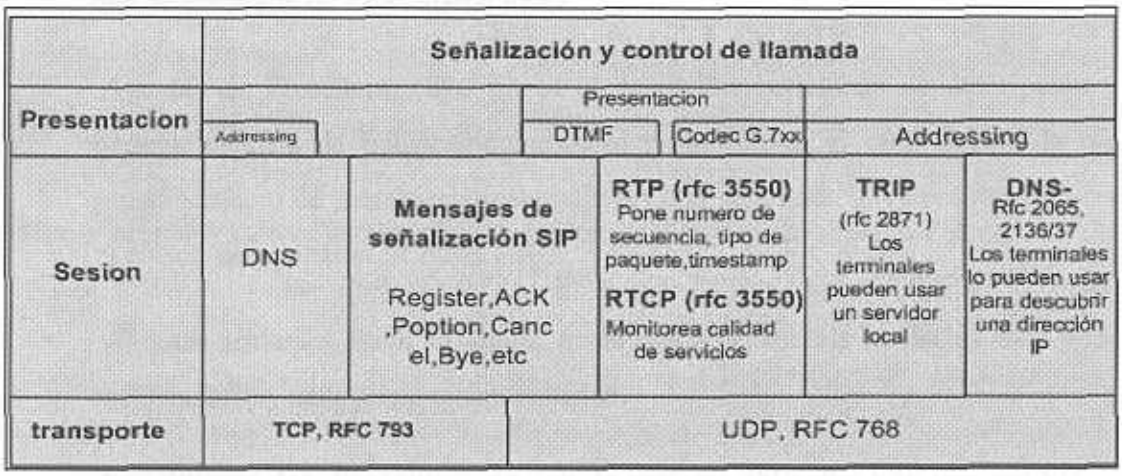

Figura Nro 13: Capas del conjunto protocolo SIP
Fuente: Khasnabish

La arquitectura SIP comprende:

- Agentes usuarios, engloban a los dispositivos SIP de usuarios, ya sean teléfonos SIP, o computadoras que emulen teléfonos SIP.

- Servidores SIP, brindan servicios a los agentes usuarios, son capaces de recibir mensajes SIP y generar respuestas a ellos. Encontramos dos tipos de servidores SIP: el Proxy Server que recibe peticiones SIP de UAs u otro servidor Proxy y

actúa en función del UA, en reenviar o responder a la petición. Por otro lado encontramos los Servidores de Redirección, encargados de responder a peticiones SIP, pero no para generar peticiones; se utilizan sobretodo para indicar locaciones de URIs. Finalmente encontramos los Servidores de Registro, y son aquellos encargados de recibir las peticiones SIP de registros para agregar los URIs a la base de datos.

- SIP Gateway, se utiliza para hacer de interfaz entre dos redes cuyas señalizaciones sean distintas; por ejemplo, si la red SIP se conecta directamente con una red telefónica es necesario que las señalizaciones SIP sean traducidas a señalizaciones SS7, con lo cual se requiere de un Gateway que se encargue de estas operaciones.

- Agentes de Presencia, cuyo fin es el de recibir peticiones de suscripción y generar estados de notificación, esto es requerido en el caso de que la red preste servicio de mensajeria instantánea, en la cual cada UA necesita conocer el estado de conexión de otros URIs.

- Agentes usuarios Back to Back, encargados de recibir mensajes SIP y de reformularlos, para luego retransmitirlos.

Los últimos dos dispositivos no son de carácter obligatorio en una red SIP, y se utilizan para prestar servicios SIP opcionales.

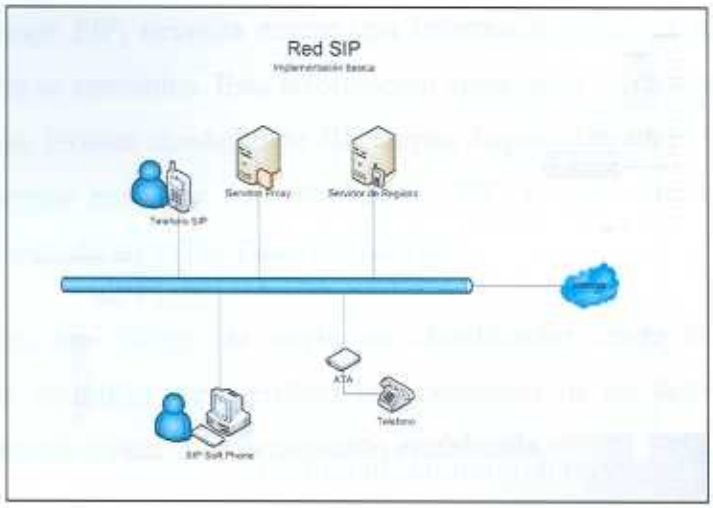

Figura Nro 14: Red SIP
Fuente: Propia

B.2 COMUNICACIÓN SIP

La comunicación SIP se sustenta mediante el envío de mensajes en texto plano entre Agentes Usuarios (UA), de los cuales existen Agentes Usuario Clientes y Agentes Usuarios Servidores; cuando un UA cliente inicia una petición, un UA servidor envía una respuesta a este mensaje.

Los mensajes de petición, llamados métodos, son 13, y los obligatorios son:

- INVITE, para el iniciar el establecimiento de la comunicación.
- REGISTER, para realizar el registro en un Agente de Registro.
- BYE, para terminar una comunicación.
- ACK, para confirmar que la conexión ha sido establecida con éxito.
- CANCEL, para cancelar una petición ya enviada.
- OPTIONS, para listar las capacidades de comunicación que posee el otro Agente Usuario.

Cada mensaje SIP, necesita enviar una información mínima al otro Agente Usuario con el que se comunica. Esta información viene dada en campos o cabeceras dentro del mensaje. Existen alrededor de 50 campos disponibles, entre los cuales 6 de ellos son obligatorios enviarlos en todo método SIP, y 1 de ellos es obligatorio anexarlo en cada método INVITE. Estos campos son:

- Call-ID: en este campo se envía un identificador único de la conexión establecida, se utiliza para verificar la procedencia de los datos, y que estos estén dentro del contexto de la conexión establecida.

- Cseq: Identificador utilizado para llevar una secuencia de los métodos que se envían los AU.

- From: este campo contiene el URI del AU que inicio el método

- To: este campo contiene la identificación o URI del AU a quien va dirigido el método.

- Vía: Este campo es modificado por cada salto por el cual pase el paquete SIP; el AU o servidor Proxy que reciba el paquete revisa el primer identificador en este campo y si concuerda con el suyo significa que el paquete es para él, sino agrega en el tope del campo su identificador y los reenvía. Esto se utiliza para facilitar el enrutamiento de los mensajes de respuesta.

- Contact: en este campo se coloca el contacto al cual debe ser dirigida la respuesta al método del mensaje. Este es el campo adicional que debe ser agregado en todo método INVITE, y una vez el receptor sabe a quien dirigir los paquetes este campo puede ser omitido.

- Max-Forwards: es el número máximo de saltos que puede dar un paquete SIP, y se compara al campo TTL de IP.

ANEXO C ESTABLECIMIENTO DE UNA LLAMADA UTILIZANDO EL PROTOCOLO SIP

ESTABLECIMIENTO DE UNA LLAMADA UTILIZANDO EL PROTOCOLO SIP

C.1 LLAMADA SIN PROXY

Supongamos dos dispositivos SIP conectados a la misma red, cada uno conoce la dirección IP del otro, los URIs de los dos usuarios en cuestión son: Marconi@radio.org con dirección IP 200.201.202.203 y n.tesla@high-voltage.org con dirección IP 100.101.102.103. Ambos trabajan con SIP versión 2.0 y desean establecer una conexión. El intercambio de mensajes entre los dos dispositivos sería el siguiente:

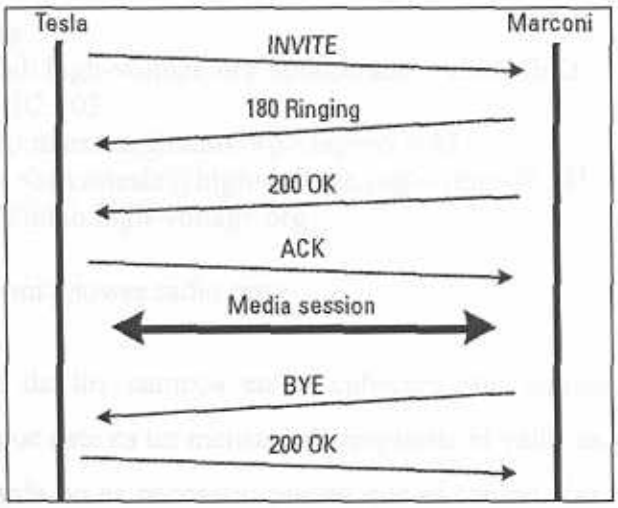

Figura Nro 15: Llamada sin SIP Proxy
Fuente: Johnston

El mensaje de INVITE, al estar en texto plano se vería de la siguiente manera:

INVITE sip:marconi@radio.org SIP/2.0
Via: SIP/2.0/UDP lab.high-voltage.org:5060; branch=z9hG4bKfw19b
Max-Forwards: 70
To: G. Marconi <sip:Marconi@radio.org>
From: Nikola Tesla <sip:n.tesla@high-voltage.org>;tag=76341
Call-ID: 123456789@lab.high-voltage.org
CSeq: 1 INVITE
Contact: <sip:n.tesla@lab.high-voltage.org>
(Contenido de información media del dispositivo de Tesla)

En este mensaje se puede observar el método INVITE en la línea de petición, seguido del URI de origen y la versión del protocolo. Los campos a continuación no poseen un orden en particular, y son los campos de cabecera obligatorios para una petición SIP. Cabe destacar que se puede observar que el campo From lleva asociado un Tag, pero el campo To aún no lo posee, esto debido a que el receptor de esta petición lo creará cuando envíe el mensaje de respuesta.

El mensaje de respuesta que recibirá Tesla, suponiendo que el dispositivo de Marconi este disponible para establecer la conexión sería el siguiente:

SIP/2.0 180 Ringing
Via: SIP/2.0/UDP lab.high-voltage.org:5060;branch=z9hG4bKfw19b;
received=100.101.102.103
To: G. Marconi <sip:marconi@radio.org>;tag=a53e42
From: Nikola Tesla <sip:n.tesla@high-voltage.org>>;tag=76341
Call-ID: 123456789@lab.high-voltage.org
CSeq: 1 INVITE
Contact: <sip:marconi@tower.radio.org>

La mayoría de los campos en la cabecera son copiados del mensaje de petición, y puesto que este es un mensaje de respuesta el valor en Cseq no cambia. El campo Max-Forwards no es necesario puesto que el campo Via indica por donde se enrutará la respuesta. En el campo Contact se coloca el URI del receptor donde puede ser contactado directamente cuando la conexión este establecida. Además, es importante notar que al campo Via se le agregó el parámetro recived (';received=100.101.102.103') este parámetro indica que la petición fue recibida de la dirección IP que contiene, y usualmente se obtiene resolviendo por DNS el URI contenido en el parámetro Via.

Por último, nótese que los parámetros From y To se mantienen igual, esto es debido a que estos parámetros indican el origen y destino del mensaje de petición y no del mensaje de respuesta.

Cuando el usuario Marconi atienda la llamada el dispositivo enviará el mensaje de información 200, junto con sus capacidades, y esperará la confirmación del origen para terminar de establecer la sesión.

SIP/2.0 200 OK
Via: SIP/2.0/UDP lab.high-voltage.org:5060;branch=z9hG4bKfw19b
;received=100.101.102.103
To: G. Marconi <sip:marconi@radio.org>;tag=a53e42
From: Nikola Tesla <sip:n.tesla@high-voltage.org>;tag=76341
Call-ID: 123456789@lab.high-voltage.org
CSeq: 1 INVITE
Contact: <sip:marconi@tower.radio.org>
(Contenido de información media del dispositivo de Marconi)

Este mensaje se crea de la misma manera que se creó el mensaje 180, pero además contiene la información de las capacidades del dispositivo de Marconi. Cuando el dispositivo de Tesla recibe el mensaje 200 OK, envía un ACK para terminar de establecer la sesión.

ACK sip:marconi@tower.radio.org SIP/2.0
Via: SIP/2.0/UDP lab.high-voltage.org:5060;branch=z9hG4bK321g
Max-Forwards: 70
To: G. Marconi <sip:marconi@radio.org>;tag=a53e42
From: Nikola Tesla <sip:n.tesla@high-voltage.org>;tag=76341
Call-ID: 123456789@lab.high-voltage.org
CSeq: 1 ACK

En este mensaje se actualiza el contacto en la línea de petición y el número de secuencia se mantiene igual, pero con el nombre de la petición actualizado.('ACK'). Cuando la sesión termina, Marconi envía una petición BYE como la siguiente:

BYE sip:n.tesla@lab.high-voltage.org SIP/2.0
Via: SIP/2.0/UDP tower.radio.org:5060;branch=z9hG4bK392kf
Max-Forwards: 70
To: Nikola Tesla <sip:n.tesla@high-voltage.org>;tag=76341
From: G. Marconi <sip:marconi@radio.org>;tag=a53e42

Call-ID: 123456789@lab.high-voltage.org
CSeq: 100 BYE

Puesto que la petición ahora la origina Marconi, los campos Via y Cseq son diferentes que cuando la petición las efectuaba Tesla. El campo Via ahora contiene la dirección del host de Marconi, y el número de secuencia de petición fue generado al azar por el mismo host. Si la petición BYE hubiera sido generada por Tesla, el campo Cseq habría sido igual a: 2 BYE.

Tesla responde a la petición con un OK, de la siguiente forma:

SIP/2.0 200 OK
Via: SIP/2.0/UDP tower.radio.org:5060;branch=z9hG4bK392kf
;received=200.201.202.203
To: Nikola Tesla <sip:n.tesla@high-voltage.org>;tag=76341
From: G. Marconi <sip:marconi@radio.org>;tag=a53e42
Call-ID: 123456789@lab.high-voltage.org
CSeq: 1 BYE

C.2 Registro en Servidor Proxy

En este ejemplo se observa el funcionamiento del método REGISTER, el cual utiliza un dispositivo SIP para registrar su contacto en un servidor SIP Proxy y que funcionará para la localización de los contactos en la realización de llamadas.

Supongamos que el URI del dispositivo que desea registrarse es werner.heisenberg@munich.de, y que la dirección IP del dispositivo es 200.201.202.203. El intercambio de mensajes para un registro se realizaría de la siguiente manera:

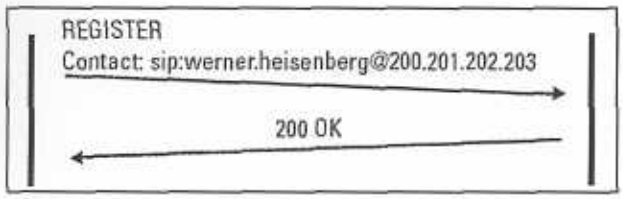

Figura Nro 16: Registro en Servidor SIP Proxy
Fuente: Johnston

La petición hecha por el agente usuario que desea registrarse es parecida a la siguiente:

REGISTER sip:registrar.munich.de SIP/2.0
Via: SIP/2.0/UDP 200.201.202.203:5060;branch=z9hG4bKus19
Max-Forwards: 70
To: sip:werner.heisenberg@munich.de
From: sip:werner.heisenberg@munich.de;tag=3431
Call-ID: 23@200.201.202.203
CSeq: 254 REGISTER
Contact: sip:werner.heisenberg@200.201.202.203

El formato del mensaje es idéntico al ya discutido anteriormente, en este caso el método es REGISTER. En esta petición el campo de mensaje To contiene el URI del contacto que desea registrarse en el servidor, pero se almacena el URI en el campo de Contact. Esto permite registros realizados por terceros, pero no serán discutidos aquí.

El servidor reconoce el registro enviando un mensaje 200 al dispositivo:

SIP/2.0 200 OK
Via: SIP/2.0/UDP 200.201.202.203:5060;branch=z9hG4bKus19
To: sip:werner.heisenberg@munich.de;tag=8771
From: sip:werner.heisenberg@munich.de ;tag=3431
Call-ID: 23@200.201.202.203
CSeq: 254 REGISTER
Contact: <sip:werner.heisenberg@munich.de>;expires=3600

En el que incluye un parámetro de expiración (expires=3600) en el campo Contact, que indica la cantidad de segundos que el URI estará registrado en el servidor.

APÉNDICE 1. MANUAL TÉCNICO DEL DISPOSITIVO

MANUAL TÉCNICO DEL TELÉFONO IP

1.1 HARDWARE

El procesador utilizado es un 486 embebido ejecutando el sistema operativo DOS. Posee 8 Mbytes de memoria Flash utilizada como memoria de acceso aleatorio.

Utiliza el puerto paralelo LPT0 para realizar las conversiones Analógico/Digital y Digital/Analógico y el puerto Eth0 de comunicaciones IP. El puerto serial puede ser conectado mediante un null modem a otro computador ejecutando HyperTerminal u otro programa similar que envíe comandos a través del puerto serial y reciba a través del mismo los caracteres enviados por el 486 embebido y los represente en una pantalla.

El diagrama de bloques de las conexiones del dispositivo se muestra en la figura siguiente

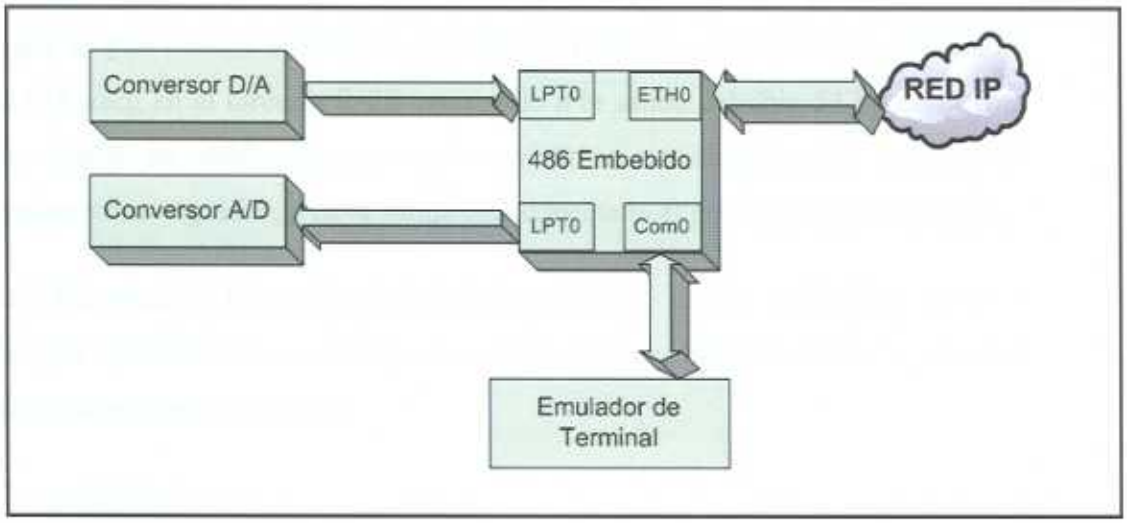

Figura Nro 17: Interconexión del hardware
Fuente: Propia

El diagrama de bloques del conversor digital analógico se muestra a continuación en la Figura.

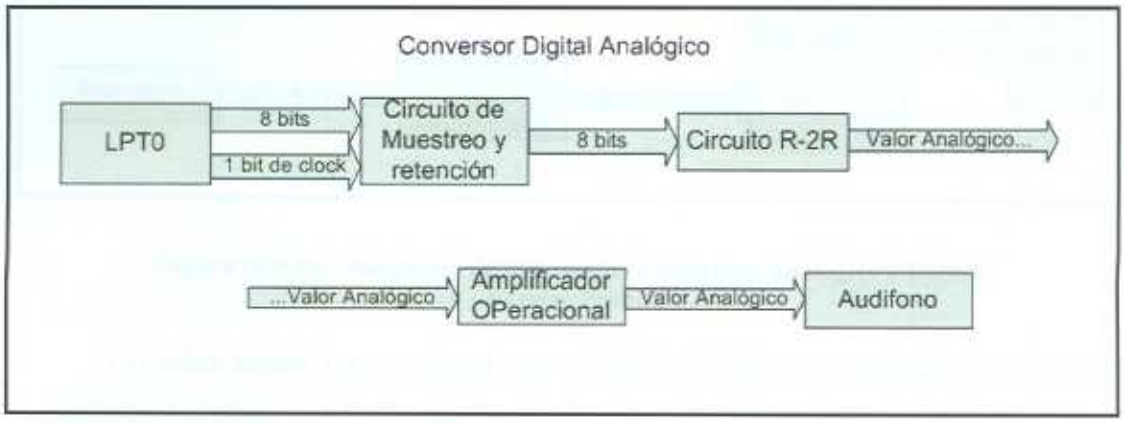

Figura Nro 18: Diagrama de bloques del Conversor Digital a Analógico
Fuente: Propia

Su funcionamiento es el siguiente: El puerto LPT0 saca un valor de 8 bits por los pines del 2 al 9, hacia un circuito 547 de muestreo y retención, para que este circuito mantenga el valor que se saca por el puerto paralelo, se envía un valor de reloj y por el pin 1 hacia el chip de muestreo y retención, este valor se mantendrá durante 125 µseg en el circuito R-2R conectado a la salida del chip 547 y convertirá el valor digital en analógico que será enviado a un amplificador operacional configurado como seguidor, y cuya salida esta conectada al audífono para el usuario

El diagrama de bloques del conversor analógico digital se muestra en la y funciona por aproximaciones sucesivas, entre un valor digital definido y el valor analógico que se desea muestrear.

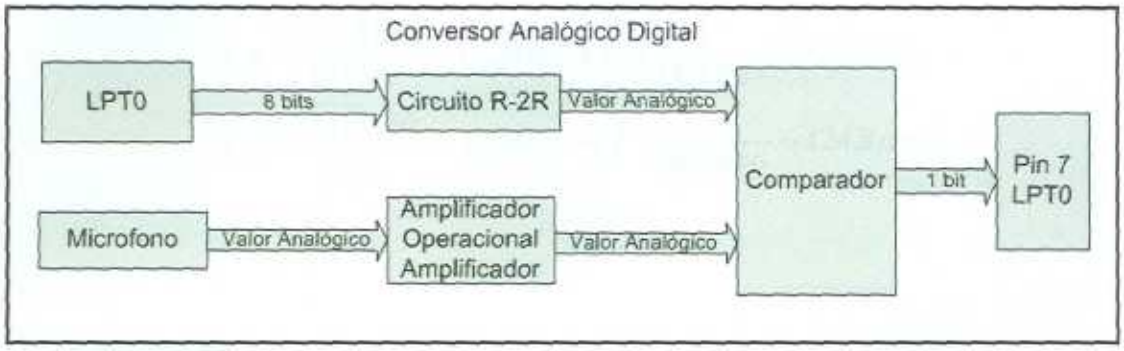

Figura Nro 19: Diagrama de bloques del Conversor Analógico a Digital
Fuente: Propia

Un valor digital (se comienza con un valor de $80_{(16)}$) es enviado por el puerto paralelo en los pines del 2 al 9 hacia un conversor R-2R cuya salida está conectada a un amplificador operacional comparador. La otra entrada del comparador recibe la señal digital analógica que se desea muestrear. Cuando la señal analógica es mayor a la señal de comparación el comparador envía un voltaje en alto, si sucede lo contrario el voltaje que envía es de cero; este voltaje es leído por el pin 11 del puerto paralelo, y si se realizan ocho comparaciones sucesivas una por cada bit de codificación se puede obtener la digitalización de la muestra.

1.1.i INTERRUPCIÓN 8, INTERRUPCIÓN DEL TIMER

Las tarjeta madre de las computadoras personales contienen dos chips temporizadores 8254; este chip contiene tres canales de tiempo, uno de los cuales genera interrupciones a 55 mseg aproximadamente, ó 18,2 veces por segundo.

El control de cada cuanto tiempo se ejecuta la interrupción 8 lo da un contador de 16 bits, cuando el contador se desborda se ejecuta la rutina de interrupción. Este contador puede ser modificado mediante el puerto 40(16) y hacer que la rutina sea llamada en intervalos menores de tiempo. El valor por defecto de este contador es de 65536, y un valor de 149 ejecutaría la rutina 8000 veces por segundo, lo suficientemente rápido para muestrear la voz humana.

$$f = \frac{1193181Hz}{65536} = 18{,}2Hz. \rightarrow T = \frac{1}{18{,}2Hz} = 54{,}9mseg.$$

$$f = \frac{1193181Hz}{149} = 8008Hz. \rightarrow T = \frac{1}{8008Hz} = 124{,}8\mu seg$$

1.2 SOFTWARE

1.2.i RUTINA DE INTERRUPCIÓN

Para poder muestrear la voz y enviar las muestras de voz a velocidad constante por el puerto paralelo, la rutina el timer de la interrupción 8 fue modificado; además la interrupción 8 fue sustituida por una rutina, llamada new_ISR08, encargada de los procesos de conversión A/D y D/A y de llevar un contador de milisegundos en una posición determinada de memoria.

Para lograr este cambio de rutina, se creó un programa VOIP.EXE, que modifica los valores del timer de la interrupción 8 y modifica el vector de interrupciones de las interrupciones 8, $80_{(16)}$, $81_{(16)}$, y $82_{(16)}$; Luego de su ejecución, la rutina new_ISR08 permanecerá residente en memoria ejecutándose a intervalos de 125 μseg, y las interrupciones $81_{(16)}$, y $82_{(16)}$ serán ejecutadas cuando el arreglo IBuf se llene y el arreglo OBuf se vacíe respectivamente. En un principio el programa VOIP.EXE coloca las rutinas de interrupción $81_{(16)}$, y $82_{(16)}$ como funciones vacías que sólo retornan, pero cada ves que son ejecutadas poseen en los registros DX y AX el segmento y desplazamiento del nuevo arreglo que se puede utilizar fuera de la rutina de interrupción.

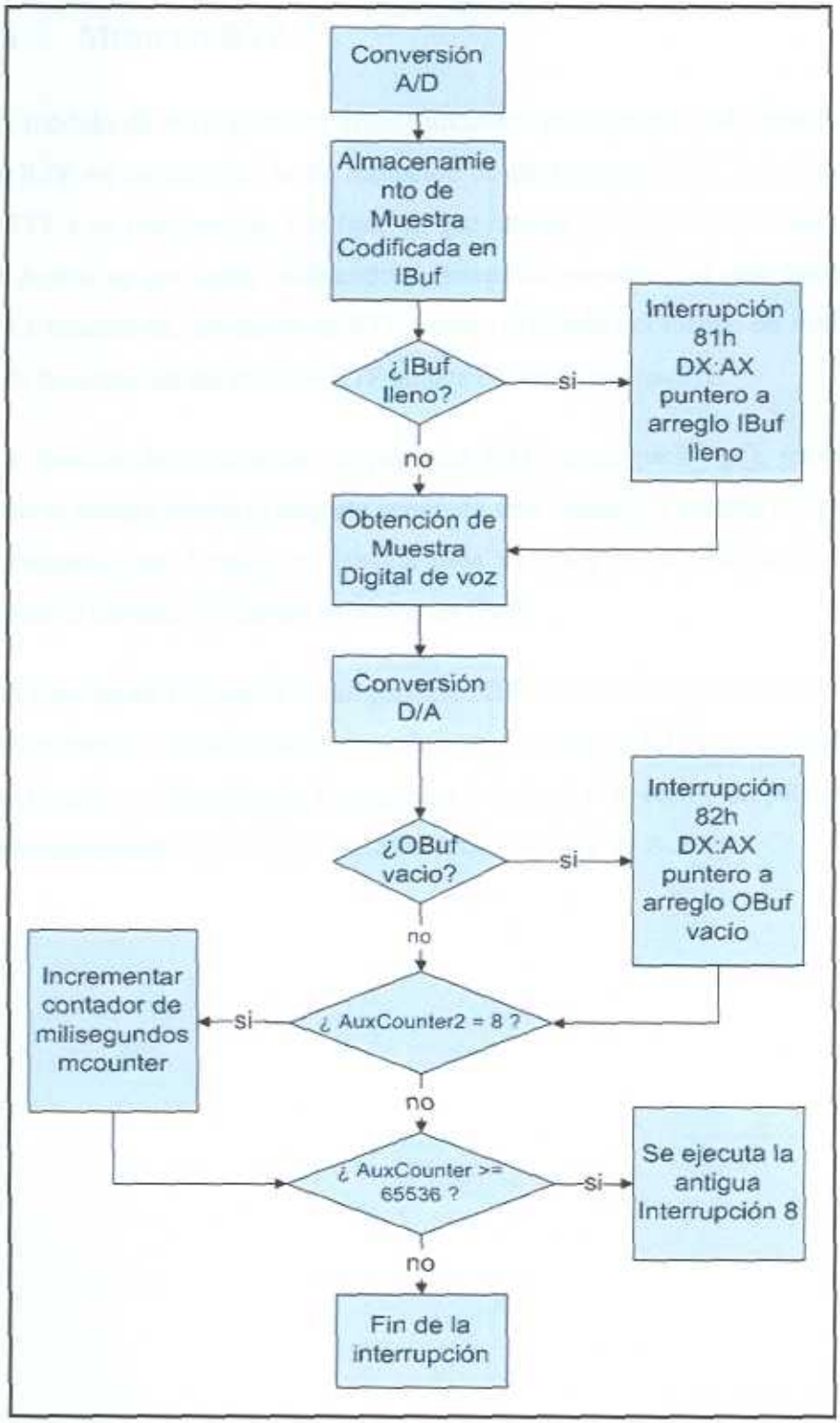

Figura Nro 20: Diagrama de Flujo de la rutina de interrupción
Fuente: Propia

1.2.ii MÓDULO RTP

El módulo de RTP presenta cinco funciones principales, la de generación de paquetes RTP en un arreglo, la de recepción de un paquete RTP, la iniciación del módulo RTP y su finalización, y la función que retorna el estado del módulo RTP, si este esta Activo se encuentra recibiendo y generando paquetes, en caso contrario no lo hace. La función de iniciación de RTP, coloca el estado del mismo en Activo, y la función de finalización del módulo RTP coloca el estado en Inactivo.

La función de generación de paquetes RTP, crear_pack_rtp(), recibe como parámetros el arreglo donde el paquete generado será copiado, y retorna el tamaño del paquete copiado. Esta función es llamada cada 15 mseg por el programa principal, para generar el paquete utilizando los datos en IBuf.

La función de recepción de un paquete RTP recibe como parámetro el arreglo donde se encuentra almacenado temporalmente el paquete RTP, no retorna ningún valor. La función es llamada por el programa principal si se recibe un paquete por el socket correspondiente a RTP y el estado del módulo RTP es Activo.

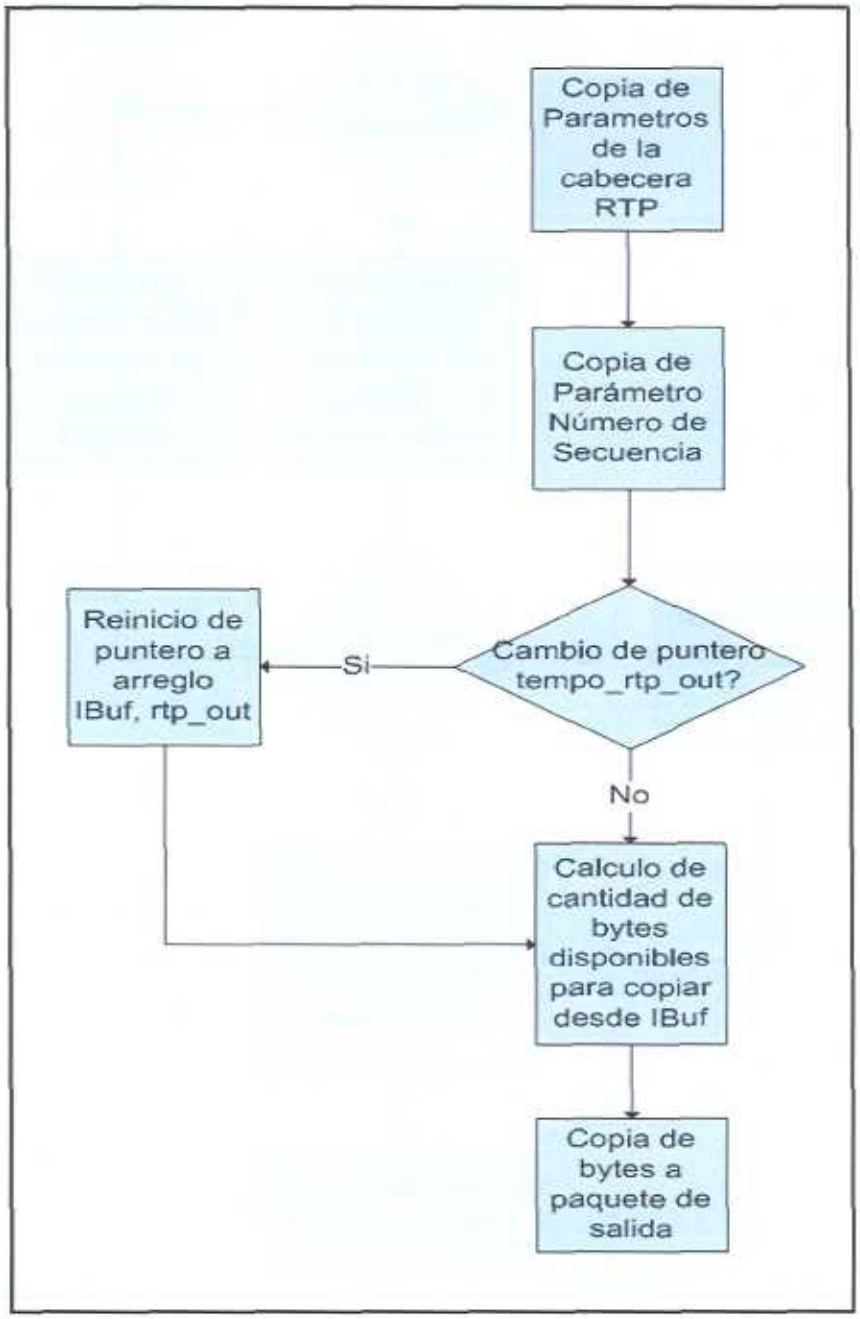

Figura Nro 21: Diagrama de flujon de la función generadora de paquetes RTP
Fuente: Propia

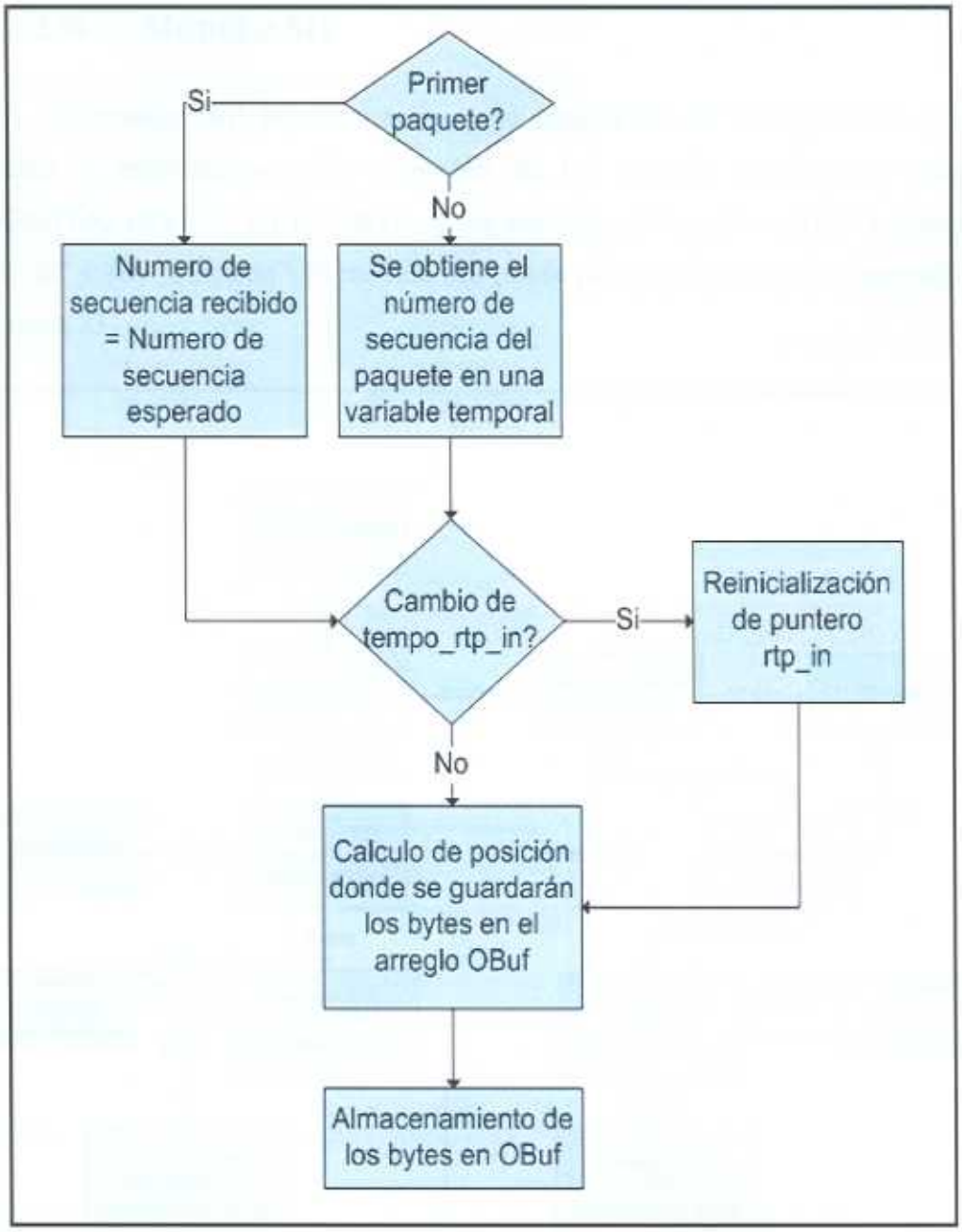

Figura Nro 22: Diagrama de flujo de la función receptora de paquetes RTP
Fuente: Propia

1.2.iii MÓDULO SIP

El módulo SIP brinda funciones de generación de mensajes SIP de tipo petición o informativos y de recepción de los mismos. Utiliza Los métodos REGISTER, INVITE, ACK y BYE, y los mensajes informativos 180 "repicando", 200 "ok" y 486 "ocupado". El módulo SIP puede pasar por los estados observados en la Figura 23

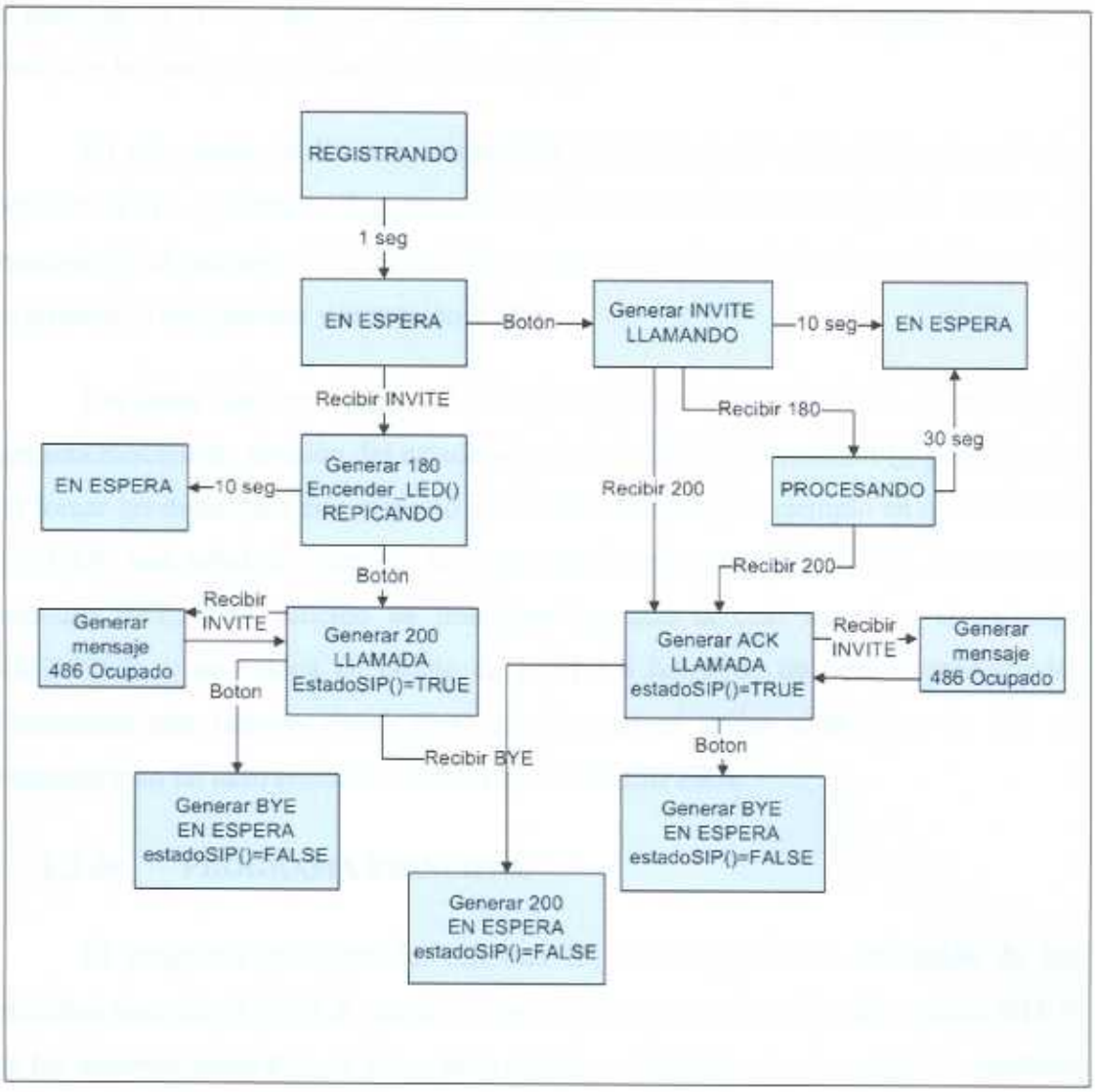

Figura Nro 23: Diagrama de estados del módulo SIP
Fuente: Propia

El módulo siempre inicia enviando un mensaje REGISTER, por el cual espera una respuesta un tiempo predefinido de un segundo, luego del cual pasa a estado EN ESPERA. En este estado es posible enviar una petición INVITE, o se recibe un INVITE. Si se recibe un INVITE se genera una respuesta de 180 Repicando, si la llamda es atendida se genera el mensaje 200 ok, luego del cual la llamada esta establecida por parte del receptor, indicándole a RTP que puede comenzar la transmisión y recepción de paquetes de Voz. Si por el contrarió el dispositivo generó la petición INVITE, luego de recibir el mensaje 200 ok genera el mensaje ACK y establece la conexión, iniciando el módulo RTP.

En un estado de llamada cualquiera de los dispositivos puede generar una petición BYE, y terminar la transmisión y recepción de RTP luego de enviar el mensaje. Si el mensaje BYE es recibido se genera un mensaje informativo 200 ok y se termina la transmisión y recepción RTP.

Las otras funciones del módulo SIP, aparte de las generadoras de mensajes, son una función de revisión del estado actual de la llamada, estadoSIP(), permitiendo así tomar las decisiones con respecto a la acción a tomar, por ejemplo en un estado de ESPERA, una señal de cierre de la comunicación por parte del usuario generará un mensaje BYE. Una función de inicialización, con la cual se envía el mensaje REGISTER, y se coloca en estado de ESPERA luego de un tiempo predefinido. Finalmente una función check_Sip() con la cual se revisa si el timer de SIP ha expirado y en tal caso reinicia el estado SIP a EN ESPERA.

1.2.iv PROGRAMA PRINCIPAL

El programa principal del teléfono IP, se encarga de la iniciación de los módulos tanto de WATTCP, como de los sockets del módulo SIP del módulo RTP y de las interrupciones $81_{(16)}$ y $82_{(16)}$ para recibir constantemente el cambio de punteros a los arreglos IBuf y OBuf, del envío de los paquetes generados por los módulos SIP

y RTP, como de la recepción de paquetes y la llamada a las funciones de recepción de paquetes de SIP y de RTP.

Una ves finalizadas las iniciaciones correspondientes, se entra en un ciclo infinito, el cual esta constantemente llamando a la función tcp_tick() de la librería WATTCP, lee el socket de SIP para recibir los paquetes si hay alguno y en caso el estado de RTP sea activo se lee el socket de RTP en espera de recibir algún paquete. Si el estado de RTP es activo, además, cada 15 milisegundos se llama a la función que genera paquetes RTP y envía los paquetes generados, finalmente, se revisa el estado de SIP y si este ha de tomar alguna acción o generar algún mensaje se llama a la función generadora de mensajes correspondiente y se envía el mensaje.

El diagrama de flujo del programa principal se representa en las siguientes dos Figuras

Figura Nro 24: Diagrama de procedimientos de inicio del programa principal
Fuente: Propia

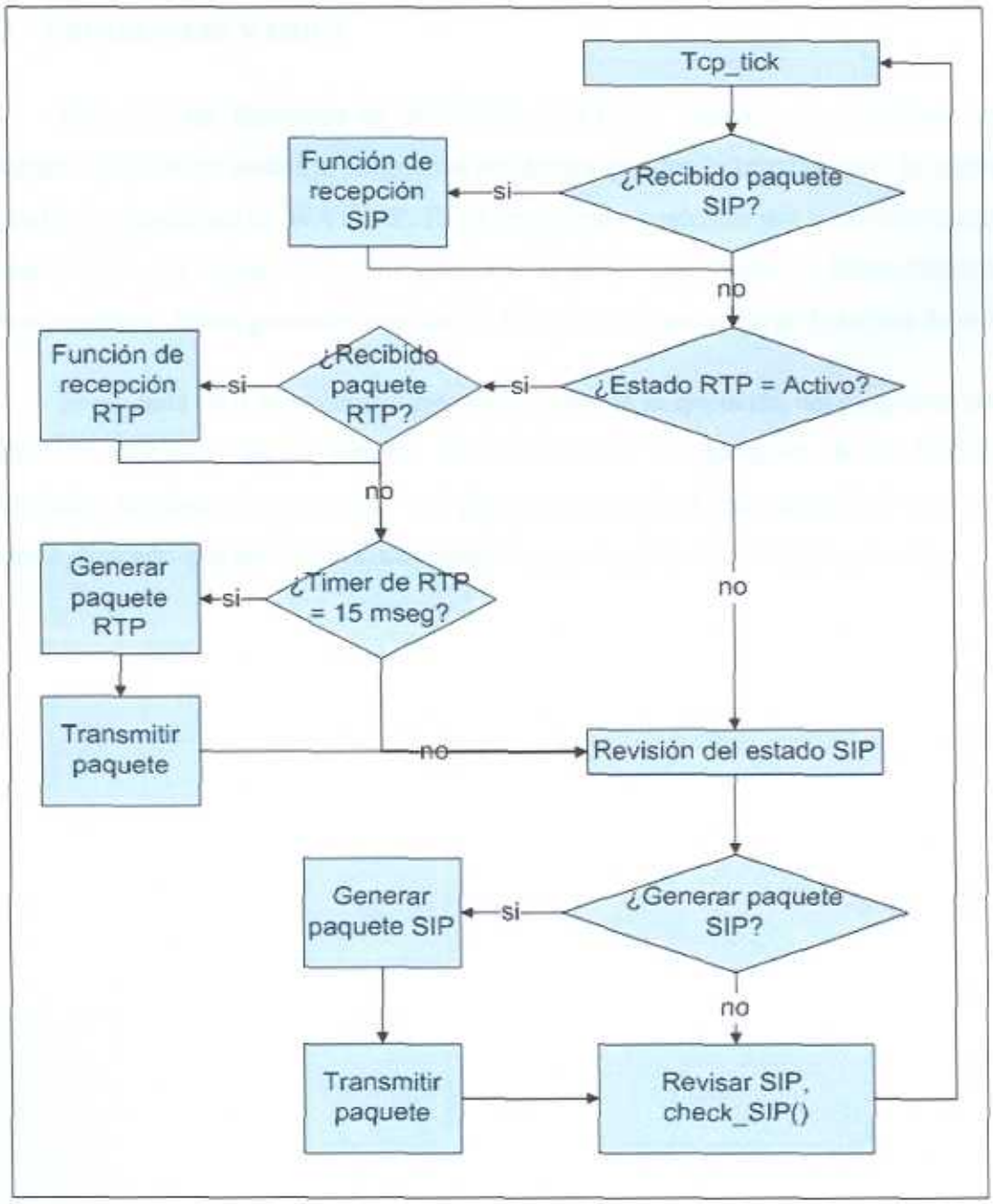

Figura Nro 25: Diagrama de Flujo del ciclo infinito del programa principal
Fuente Propia

1.3 PROGRAMAS VARIOS

Para que las funciones de la librería WATTCP trabajen correctamente, es necesario instalar en memoria programas residentes que den la interfaz entre la tarjeta de red y las funciones de WATTCP. Estos programas residentes son conocidos como paquetes ODI por "Open Data-Link Interface" ó Interfaz de Enlace de Datos Abierta. Estos paquetes vienen generalmente con el disco de configuración de la tarjeta de red.

El paquete ODI es necesario ejecutarlo antes de la ejecución del programa que usará las funciones de la librería WATTCP, sino la ejecución de la función sock_init() terminará la ejecución del programa y enviará un mensaje de error al usuario diciendo que no fueron encontrados los manejadores de la tarjeta de red.