Dispositivo Autónomo para Reproducción de Streaming de Audio Sobre IP

Diego Fernando Bucheli Callejas

José Alejandro Sarria Villa

Universidad del Cauca
Facultad de Ingeniería Electrónica y Telecomunicaciones
Departamento de Telemática
Área de Sistemas Empotrados
2010

Dispositivo Autónomo para Reproducción de Streaming de Audio Sobre IP

Diego Fernando Bucheli Callejas

José Alejandro Sarria Villa

Trabajo de grado como requisito para optar al título de Ingeniero en Electrónica y Telecomunicaciones

Director: Mag. José Armando Ordoñez

Universidad del Cauca
Facultad de Ingeniería Electrónica y Telecomunicaciones
Departamento de Telemática
Área de Sistemas Empotrados
2010

Agradecimiento

Gracias a Dios por permitirme culminar con éxitos mis estudios, reflejados en este trabajo de grado. A mi papa Carlos Fernando Bucheli porque siempre has sido mi modelo a seguir como persona y como profesional, a mi mama Yolanda Callejas por su apoyo incondicional y ejemplo de perseverancia, a mi hermano Carlos Manuel Bucheli por su sincera amistad y sus concejos, finalmente a todas las personas que de forma directa o indirecta contribuyeron con este logro.

Diego Fernando Bucheli

A Dios que es la fuerza que me impulsa para cumplir cada una de mis metas, a mi madre Maria Divina Villa Gomez por su apoyo incondicional y su firmeza, a mis hermanos por su confianza y compañía, a mi familia por creer siempre en mí, a mis profesores por ayudarme a crecer tanto a nivel profesional como humano, a mis amigos y conocidos por brindarme siempre su amistad y solidaridad.

José Alejandro Sarria Villa

Abstract

This paper describe the implementation of a self-contained device for playback of streaming audio over IP specifically for radio broadcast University of Cauca, the document also introduce the hardware, software, illustrate the general device behavior, and present an cost estimation in order to past it from development stage to manufacture.

Resumen

El presente trabajo es un aporte a la investigación aplicada que ofrecerá un primer y gran avance hacia el desarrollo de dispositivos hardware de bajo costo que permita suplir necesidades de la universidad a costos que de otra manera serían prohibitivos. El proyecto busca responder a una necesidad planteada por la Radio Universidad del Cauca, la cual consiste en el desarrollo de un dispositivo autónomo para la recepción y reproducción de Streaming de audio sobre IP de la señal emitida a través de internet por Unicauca Estéreo, partiendo del hecho de que internet es uno de los medios más utilizados y con la infraestructura adecuada en el campus universitario para tal fin y al tiempo una alternativa para llegar a sedes en las que la señal radiada no es una alternativa. Igualmente, dicho dispositivo debe tener un desarrollo de bajo costo y debe permitir acceder a contenidos diferentes al emitido por la emisora con el fin de generar variedad en la información a reproducir y acceder a contenidos de diferentes fuentes en el mundo. Este dispositivo permitirá ampliar la cobertura y difundir el contenido actual de la emisora entre los estudiantes y funcionarios, aumentando el sentido de pertenencia con la institución.

CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

CAPÍTULO 1. INTRODUCCIÓN

1.1 CONTEXTO

En la cotidianidad puede observarse como las personas tienen acceso a la información por diferentes medios de comunicación como: la radio, la prensa, la televisión, etc. Sin embargo, cabe destacar que el internet ha tenido una gran acogida como fuente de información, esto ha "obligado" que los medios de información como la radio migren hacia dicha plataforma tecnológica [1], esto sumado al incremento de la creación de dispositivos finales con acceso a la red genera una gran variedad de oportunidades. Los dispositivos embebidos con características de red, como el que se presenta en este proyecto, tiene como propósito reproducir audio Streaming, específicamente el generado por la radio Universidad del Cauca y difundirlo a las distintas dependencias y oficinas del claustro así como a sedes que están aisladas geográficamente y la señal radiada no es una alternativa, el resultado obtenido es un sistema autónomo basado en un microcontrolador y optimizado con el fin de que su implementación sea de bajo costo comparado con los dispositivos que poseen características similares disponibles en el mercado.

Las ventajas que ofrece este desarrollo van desde un sistema de intercomunicación interinstitucional hasta la creación de espacios que cuentan con un sonido ambiente específico por ejemplo: bibliotecas, laboratorios u otras dependencias, espacio que puede ser aprovechado por la Radio Universidad del Cauca para hacer llegar información relevante a toda la comunidad universitaria [2] gracias a que la gran mayoría de sus instalaciones cuentan con acceso a la red, y así crear un canal de comunicación, transmitiendo información específica a un área común o un despacho en particular [3].

En la actualidad, muchas emisoras trasladan sus contenidos radiados a un medio con mayores posibilidades como la Internet, especialmente aquellas emisoras institucionales. Por lo cual, es importante identificar los mecanismos que van de la mano con este cambio, por ejemplo en la forma de recepción y reproducción de los contenidos ya sean musicales, culturales, institucionales o como es el caso de la Radio Universidad del Cauca una propuesta educativa enmarcada dentro de una variada programación de géneros musicales y temáticas de interés general [4].

A pesar de la facilidad de acceder a los contenidos radiales, al interior de la institución no se promueve de manera clara y eficiente el uso de este medio de comunicación. Lo que se busca con la creación de un dispositivo de reproducción de audio streaming de bajo costo es atender a una necesidad de ampliación de oyentes de la emisora Radio Universidad del Cauca llevando directamente a la comunidad universitaria los contenidos que se generan aprovechando la infraestructura de red de todo el campus incluyendo sedes fuera de la ciudad de Popayán como por ejemplo el municipio de Santander de Quilichao.

1.2 DEFINICIÓN DEL PROBLEMA

Es posible observar como Internet es, ahora, un medio de comunicación importante comparado con los definidos como tradicionales: prensa y radio; transformando en una necesidad, para el "hombre actual", mantenerse informado a través de ella, lo cual ha generado la necesidad de crear dispositivos que permitan el acceso hacia los diferentes medios de comunicación soportados por estas nuevas tecnologías. Uno de ellos es conocido como *Audio Streaming* (flujos de audio) sobre IP, tecnología que permite la reproducción de "sonidos" por conexión a un servidor de audio remoto a través de una red de datos o Internet [5]. Esta tecnología de a*udio streaming*sobre IP permitiría aprovechar la infraestructura con la que cuenta hoy en día la Universidad del Cauca [3], esto significa que implementando un dispositivo que cumpla con ciertas características ayudaría a difundir, de manera eficaz, información de interés general a toda la comunidad universitaria, donde los contenidos estarían a cargo de su División de Comunicaciones.

Una de las barreras al momento de implementar un dispositivo de audio streaming es el elevado costo que puede llegar a tener un dispositivo autónomo para la reproducción de Streaming de audio, atendiendo a que las soluciones disponibles en el mercado son productos con características adicionales que quizás no son de mayor utilidad en el escenario de estudio como el propuesto (Universidad del Cauca), partiendo de esto, es poco viable la adecuación de un dispositivo para la reproducción de streaming de audio como los analizados más adelante en este documento, en vista de que son soluciones tecnológicas propietarias que cuentan con sus propias herramientas de programación y así mismo con arquitecturas propias ya definidas. [6], [6.1], [6.2], [6.3].

1.3 OBJETIVOS

1.3.1 OBJETIVO GENERAL

Diseñar e implementar un dispositivo de bajo costo para la reproducción de flujos de audio sobre IP de manera autónoma.

1.3.2 OBJETIVO ESPECÍFICOS

- ✓ Determinar la mejor combinación de dispositivos, microcontroladores, sistema operativos e interfaces para la solución requerida.
- ✓ Desarrollar un sistema para la gestión y control de la información que será enviada a las diferentes dependencias donde esté ubicado el dispositivo.

- ✓ Facilitar el acceso a un medio de comunicación masivo difundido a través de Internet a las diferentes dependencias de la Universidad del Cauca por medio de un dispositivo autónomo de bajo costo.

1.4 SOLUCIÓN PROPUESTA

La solución propuesta consiste en diseñar e implementar un dispositivo de uso masivo y bajo costo capaz de recibir y reproducir flujos de audio digital transportados mediante el protocolo IP, los cuales sean transmitidos por medio del servidor de audio de la Radio Universidad del Cauca. Y desarrollar un módulo de selección de contenido que nos permita elegir los contenidos que se decidan difundir los cuales pueden ser específicamente los propuestos por la Radio Universidad del Cauca o también contenidos afines y ofrecidos por algún servidor de audio por internet.

1.5 CONTRIBUCIONES Y PRINCIPALES APORTES

El proyecto busca diseñar e implementar un dispositivo autónomo de bajo costo, entendiéndose autónomo como un equipo dedicado que no requiere hardware o software externo, el cual teniendo acceso a una conexión a Internet y a la red eléctrica permita la recepción y reproducción de audio utilizando la infraestructura de datos de la Universidad del Cauca, la información recibida a través del dispositivo va a ser la misma que se radia, el dispositivo cuenta además con la funcionalidad de selección de contenidos la cual permite tener un contenido más selectivo (diferente al radiado) entre los usuarios llevando determinada información a un área específica.

Los principales aportes de este trabajo de grado son:

- El diseño y construcción de un dispositivo autónomo de bajo costo, entendiéndose autónomo como un equipo dedicado que no requiere hardware o software externo, y con una simple conexión a Internet y a la red eléctrica permite la recepción y reproducción de audio utilizando la infraestructura de datos de la Universidad del Cauca.

- La implementación software basada en algún tipo de firmware o código base en el cual se agregue una función de gestión que consiste básicamente en controlar el acceso a una emisora predeterminada como lo es Radio Universidad del Cauca o a un determinado número de emisoras preseleccionadas, esta función es posible llevarla a cabo de manera remota o de manera local.

- Aportar un estudio sobre cuáles son los fundamentos teóricos a tener en cuenta para el desarrollo de este tipo de dispositivos y aplicaciones. Identificando cuáles son los

protocolos necesarios y específicos para poder capturar, reproducir y almacenar los flujos de audio Streaming sobre IP.

- Desarrollo de un prototipo funcional que contenga una serie de elementos producto de un estudio comparativo que pretende determinar la mejor combinación de dispositivos hardware e interfaces que existen en la actualidad para la solución requerida. Basados en la relación costo/beneficio, herramientas libres y además aprovechar elementos que pueda contar la universidad como programadores, software, etc.

1.6 CONTENIDO DEL DOCUMENTO

El capítulo II presenta los fundamentos teóricos del trabajo. Inicia con la definición de streaming de audio, planteando una idea general sobre este tipo de tecnología. Seguidamente estudia la arquitectura TCP/IP desde el punto de vista del Streaming analizando los diferentes protocolos y planteando la configuración más adecuada para nuestro problema de investigación aplicada. Posteriormente los conceptos de Streaming y arquitectura TCP/IP en el concepto de *Sistema Embebido* son conjugadospara lo cual es necesario dar claridad sobre la composición de los mismos, e introduce al lector en el concepto de *Sistema Mínimo* y plantea la utilización de Microcontroladores como elementos centrales de estos sistemas.

El capítulo III corresponde al desarrollo hardware de este proyecto, inicia con el estudio de cada una de las fases del sistema analizando los elementos hardware más adecuados para la implementación de nuestro dispositivo y los elementos teóricos necesarios para que este funcione correctamente, una vez determinado esto, realiza un estudio detallado de los componentes seleccionados así como un diagrama de bloques en donde identifica las principales fases de desarrollo que componen el prototipo. A continuación basándose en un diseño compuesto por módulos y su correspondiente interacción propone un diseño circuital (Anexo B) teniendo en cuenta las tablas de especificación de cada uno de los elementos, este diseño desarrollado mediante el uso de alguna herramienta adecuada para esta labor, obteniendo como producto final de este capítulo los diagramas finales de una PCB (Anexo C) y los correspondientes archivos Gerber necesarios para fabricar el prototipo IRD *(Internet Radio Device)*.

Seguidamente el capítulo IV analiza diferentes plataformas software para la implementación de la función de selección de contenidos del dispositivo, esta función consiste en una aplicación cliente-servidor que requiere de un proceso de autenticación por parte del usuario para poder seleccionar el contenido que se va a reproducir, el usuario puede acceder a la Radio Universidad del Cauca o a otras emisoras preseleccionadas. A lo largo de este capítulo existe una explicación breve sobre el código implementado y las funciones requeridas para poder realizar el proceso de reproducción

de Streaming de audio así como de las rutinas a generar para poder integrar todos los elementos seleccionados en el Capítulo III correspondiente al Desarrollo Hardware.

El capítulo V corresponde a una serie de pruebas para verificar el correcto funcionamiento del prototipo que integra el diseño Hardware y Software, las pruebas consisten básicamente en una serie de instrucciones para la conexión y manipulación del dispositivo a través de la red.

Finalmente, el capítulo VI Conclusiones y Líneas Futuras presenta las conclusiones y algunas perspectivas de futuros trabajos.

Los anexos generados son los siguientes:

Anexo A. Diagrama circuital
Anexo B. Diseño PCB (PrintedCircuitBoard).
Anexo C. Tablas de especificaciones de elementos del dispositivo.
Anexo D. Código fuente.
Anexo E. Análisis de Costos
Anexo F. Manual de Usuario
Anexo G. Lista de elementos (*Bill of Materials*)

CAPÍTULO 2. MARCO TEORICO

2.1 INTRODUCCIÓN

Este capítulo está dividido en dos partes, inicialmente profundiza en la teoría de Streaming describiendo la manera en que este tipo de tecnología puede clasificarse según su modo de distribución, identificando dentro de esta clasificación el tipo de Streaming que va a reproducir el prototipo final desarrollado, en adelante llamado IRD (*Internet Radio Device*, Dispositivo de Radio por Internet). Seguidamente expone los fundamentos teóricos sobre los cuales está basado el desarrollo del Sistema propuesto IRD, para ello ofrece una introducción sobre la Arquitectura TCP/IP, específicamente en los niveles o capas de Aplicación y de Red, teniendo en cuenta la manera en que estas son integradas con el proceso de transmisión y recepción de audio *streaming*, esta introducción finaliza con una tabla comparativa sobre los protocolos involucrados en el proceso de *Streaming* buscando resaltar los más adecuados para el desarrollo de nuestro proyecto.

La segunda parte de este capítulo realiza una breve comparación entre los dispositivos existentes, mencionando algunos proveedores y el costo de las soluciones que proponen, con ello se busca que el lector identifique la diferencia y las ventajas de desarrollar un dispositivo con las características que a lo largo de este proyecto está planteado. Por último y buscando relacionar la teoría expuesta este primer capítulo expone de manera clara los Sistemas Mínimos como soporte hardware para el desarrollo del Sistema, además de los sistemas embebidos, su composición e implementación así como las herramientas software proporcionadas para tal propósito.

2.2 STREAMING DE AUDIO

2.2.1 TECNOLOGÍA DE STREAMING DE AUDIO

El Streaming de Audio consiste en la emisión de archivos de audio desde un servidor y a través del protocolo Internet (IP, *Internet Protocol*) permitiendo conexiones remotas desde computadoras o dispositivos autónomos. En la práctica, los dispositivos que reproducen Streaming de audio tienen un papel similar al que cumplen receptores de radio tradicionales al sintonizar la programación de una emisora de radiodifusión. Antes de la existencia de este tipo de tecnología, para poder reproducir sonidos desde la red, era imprescindible hacer una descarga del archivo de audio para oírlo posteriormente. La tecnología de Streaming de Audio permite la reproducción de sonido en tiempo real, es decir, en el mismo momento en que el dispositivo del usuario final establece una conexión con el servidor, sin necesidad de descargar previamente ningún archivo. Cabe aclarar que bajo esta premisa puede realizarse Streaming de audio en vivo o en diferido [7].

Una sesión de Streaming está definida por los siguientes pasos; primero el cliente se conecta con el servidor e inicia el envío de contenido que puede ser almacenado o codificado en tiempo real, luego el cliente ubica el contenido recibido en un buffer de almacenamiento, cuando en el buffer tenga cierta cantidad del contenido procede a reproducirlo al tiempo que la transmisión continua por parte del servidor. Suponiendo un decremento en la velocidad de la descarga y una interrupción considerable en el tiempo, el buffer queda vacío y la ejecución del contenido queda pospuesta hasta que la conexión sea restaurada.

2.2.2 TIPOS DE STREAMING

El proceso de Streaming puede dividirse en dos categorías, en función de la manera de obtener la información a difundir: Streaming en directo y Streaming bajo demanda [8].

El Streaming en directo es aquel que transmite eventos que están sucediendo justo en el momento de la difusión. Por ejemplo, la transmisión de conferencias, presentaciones culturales, clases magistrales o similares, un evento deportivo, normalmente pueden difundirse usando este tipo de Streaming. La transmisión de radio y televisión por Internet también tiene estas características, aunque en ocasiones parte de la información que se difunde no es necesariamente un evento en directo (por ejemplo, un programa que ha sido grabado previamente, pero que va a difundirse en un momento determinado). En este tipo de transmisión empleamos el término difusión (Broadcast) porque realmente está transmitiendo “en vivo” a todos los clientes la misma información, que no es más que el evento que existe en ese momento. Así, independientemente de cuándo un cliente se conecta con el servidor, todos ven exactamente el mismo punto del Stream en un instante determinado (excepto las variaciones de los retardos en la red que hacen que unos clientes reciban antes los datos que otros). Además, para dar un servicio realmente eficiente de este tipo de Streaming es conveniente que la difusión sea realizada con técnicas de Multicast.

En un Streaming multimedia bajo demanda, la transmisión del medio empieza desde el inicio del evento a ser reproducido para cada uno de los clientes. Los pasos para llevar a cabo una sesión Streaming de este tipo son iguales a los descritos con anterioridad pero adicionalmente se define una interacción desde el cliente hacia el servidor, para ello requiere de un canal bidireccional que permita el control de los flujos multimedia. Adicional a esto, es necesario para este tipo de Streaming (bajo demanda) tener en cuenta determinados protocolos en la capa de aplicación y de transporte que brindan una sensación de interactividad por parte del usuario.

2.2.3 DIFUSIÓN DE STREAMING

En Multicast un único Stream es compartido entre diferentes clientes de tal forma que el servidor envía la información una única vez y ésta llega a todos los clientes que han demandado el servicio. Como ya hemos comentado, esta técnica es ideal para la difusión de medios en vivo (por ejemplo, radio en Internet como el que provee la Radio - Universidad del Cauca) ya que todos los clientes están dispuestos a recibir el mismo flujo de datos.

Esta técnica reduce claramente el tráfico en la red. Sin embargo, para llevarla a cabo es necesario tener acceso a una troncal con soporte Multicast, o que el servidor y los clientes estén conectados a una red o redes IP bajo un mismo dominio de administración en las que el Multicast esté habilitado y existan enrutadores (routers) dispuestos a encaminar información Multicast (routers Multicast).

En la transmisión típica Unicast cada cliente inicia su propio Stream independientemente de que todos los clientes estén interesados en el mismo Stream de datos (esto es, aunque sea una difusión “en vivo”), de forma que muchas conexiones uno-a-uno son iniciadas (una entre el servidor y el cliente por cada uno de los clientes). Esta técnica causa que el servidor requiera de un ancho de banda mucho mayor que los clientes y aumenta el tráfico en la red.

2.3 ARQUITECTURA DESDE EL PUNTO DE VISTA STREAMING

Buscando establecer una base teórica sobre la arquitectura necesaria para establecer un proceso de reproducción de Streaming, a continuación son mencionados los elementos presentes dentro de dicha arquitectura (Figura 1).

Figura 1. Arquitectura para el Streaming de Audio

La Arquitectura Streaming está compuesta por:

- Sistema de producción
- Formatos de almacenamiento
- Servidor (Base de datos)
- Proxy
- Red/Protocolos
- Cliente

Dado que nuestro objetivo es desarrollar un Dispositivo para la reproducción de Streaming sobre IP a continuación son detallados los elementos: Red/Protocolos y cliente, los elementos restantes son establecidos por la infraestructura hardware y software provisto por la Universidad del Cauca

2.3.1 RED/PROTOCOLOS

Al referirse al Streaming existen unas limitaciones en cuanto al tiempo de transmisión de los archivos, por lo tanto, requiere una mayor coordinación entre cliente-servidor ya que el ritmo de transmisión está marcado por los tiempos de reproducción del contenido. En este tipo de tecnología la pérdida de paquetes puede manejarse gracias a la utilización de protocolos conocidos como *ligeros*, entre ellos están: Protocolo de Transporte de Tiempo Real (RTP, *Real-time TransportProtocol*) [9], Protocolo de Flujo de Datos en Tiempo Real

(RTSP, *Real Time Streaming Protocol*) [10] y el Protocolo de Datagramas de Usuario (UDP, *UserDatagramProtocol*) [11].

La utilización de un protocolo u otro depende en gran parte del tipo de tráfico a cursar, específicamente para nuestro caso de estudio, Radio Universidad del Cauca, el objetivo es centrarnos en la captura y reproducción del Streaming generado (*Streaming live*), las características adicionales de interactividad provista por los protocolos ligeros, como puede verse más adelante, no son objeto de estudio.

2.3.2 PROTOCOLO DE TIEMPO REAL (RTP)

Con el auge de la información en tiempo real a través de internet fue necesario desarrollar por parte de los entes reguladores (IETF, *Internet EngineeringTaskForce*) un protocolo que proporcionara las condiciones necesarias para llevar a cabo la transmisión de audio y video en tiempo real, teniendo en cuenta que TCP como protocolo de transporte no fue diseñado para este fin y no cumple, en sus versiones estándar, con las expectativas y necesidades de las nuevas aplicaciones.

El protocolo RTP fue creado específicamente para la transmisión de audio y video, gracias a que incluye en su cabecera información que permite sincronizar imagen y sonido, al tiempo determina si se han perdido paquetes y si estos han llegado en el orden correcto. Cuando una conexión RTP es realizada, dos direcciones distintas son definidas, ellas son controladas desde dos puertos distintos, de forma que audio y video viajan por separado controlados por el RTCP (*Real Time Control Protocol*, Protocolo de Control de Tiempo Real) adicional a esto el protocolo RSVP (*ResourceReServationProtocol*, Protocolo de Reserva de Recurso) tiene como objetivo añadir información de retorno desde el cliente hacia el servidor para garantizar una calidad de servicio. Cabe aclarar que RTP no asegura la entrega continua de información, como tampoco la de todos los paquetes o la entrega desordenada de los mismos, aunque si los controla. El protocolo RTP está dentro del modelo OSI entre la capa de transporte y la capa de aplicación por lo cual es implementada sobre los protocolos subyacentes TCP o UDP pero generalmente es soportado sobre el protocolo UDP ya que posee menor retardo que TCP.

2.3.3 PROTOCOLO DE FLUJO DE DATOS EN TIEMPO REAL (RTSP)

El protocolo de Flujo de Datos en Tiempo Real (*RTSP*, *Real Time Streaming Protocol*) está a nivel de aplicación de presentación multimedia cliente/servidor, permite controlar el flujo de audio y video sobre la red IP, existen algunos otros protocolos que trabajan de manera similar, entre ellos:MMS de *Microsoft (MMS, Microsoft Media Server)* o *RTMP (Real-Time MessagingProtocol*) y *RTMFP (Real Time Media FlowProtocol*) de *Adobe*, por ser propietarios no son de nuestro interés.

Como fue mencionado, este protocolo introduce un nuevo concepto en el tratamiento de contenido multimedia a través de la red, introduciendo una característica adicional como es la interactividad de esta manera se tienen dos canales de comunicación entre el cliente y el servidor: i) Un canal para el control de la sesión (RTSP), ii) Un canal para la transmisión de la información (RTP/UDP/TCP) (Ver Figura 2).

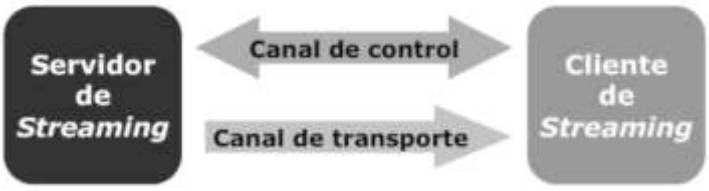

Figura 2. Comunicación entre Servido-Cliente por RTSP

Este protocolo permite la recepción de información multimedia desde distintos servidores, permitiendo que el cliente tenga la capacidad de establecer varias sesiones de control concurrente con diferentes servidores de medios, sincronizándolos en el nivel de transporte de aquí que su implementación esta principalmente en servicios dedicados de Streaming.

RTSP puede caracterizarse por su estructura lexicográfica similar a la de HTTP y a MIME[1] permitiendo que todos los mecanismos de autentificación de HTTP tales como autenticación básica o de resumen sean aplicables. Además reutiliza mecanismos de seguridad de la Web en el nivel de transporte o en el protocolo en sí. RTSP funciona en forma independiente del protocolo de transporte pudiendo hacer uso de un protocolo no fiable de datagramas (UDP) o un protocolo fiable de datagramas (RTP) y un protocolo confiable (TCP). Mediante este protocolo puede controlarse la grabación, así como el retroceder o avanzar en la reproducción del Streaming, además reutiliza conceptos de HTTP y la infraestructura del mismo protocolo.

Ventajas de RTSP

- Compresión del vídeo digital, descartando cuadros y detalles redundantes.
- Transmisión de los datos en paquetes que son leídos por el cliente mientras llegan.
- Utilización de buffers o memorias de reserva para aminorar los retardos y demoras inherentes a la red.

[1] MIME es acrónimo de "Multipurpose Internet Mail ExtensionsEncoding", un estándar utilizado en Internet con dos finalidades: de un lado, normalizar el intercambio de todo tipo de archivos (texto, audio, vídeo, etc) en la Red; de otra, acabar con el problema de las transferencias de texto internacional por e-mail.

- No requiere almacenamiento en el cliente.
- No hay desperdicio de ancho de banda.
- Difusiones en Multicast.
- Pueden comprimirse pistas individuales en un film a partir de un servidor, no importa donde este geográficamente.
- Escalable.

Desventajas de RTSP

- Un video se detiene si la tasa de datos excede la velocidad de conexión.
- Puede ser detenido por firewalls o el NAT.
- Requiere un servidor Streaming o un Broadcast. [12]

2.3.4 HTTP EN APLICACIONES STREAMING

Hypertext Transfer Protocol o HTTP es un protocolo orientado a transacciones y sigue el esquema petición-respuesta entre un cliente y un servidor. El cliente que efectúa la peticiónes es conocido como "useragent" (agente del usuario). La información transmitida es llamada recurso y es identificada mediante un localizador uniforme de recursos (URL). Los recursos pueden ser archivos, el resultado de la ejecución de un programa, una consulta a una base de datos, la traducción automática de un documento, etc. HTTP es un protocolo sin estado, es decir, que no guarda ninguna información sobre conexiones anteriores [13].

La transacción está formada por un encabezado seguido, opcionalmente, por una línea en blanco y algún dato. El encabezado especificará cosas como la acción requerida del servidor, o el tipo de dato retornado, o el código de estado. El uso de campos de encabezados enviados le da gran flexibilidad al protocolo. Estos campos permiten él envió de la información descriptiva en la transacción, permitiendo así la autenticación, cifrado e identificación de usuario. Un encabezado es un bloque de datos que precede a la información propiamente dicha, por lo que muchas veces es referenciado como metadato, término que hace referencia a que son datos sobre los datos. Si son recibidas líneas de encabezado del cliente, el servidor las coloca en las variables de ambiente de CGI con el prefijo HTTP_ seguido del nombre del encabezado. Cualquier carácter guión (-) del nombre del encabezado es convertido a caracteres "_".

El servidor puede excluir cualquier encabezado que ya esté procesado, como *Authorization*, *Content-type* y *Content-length*. El servidor puede elegir excluir alguno o todos los encabezados si incluirlos exceden algún límite del ambiente de sistema. Ejemplos de esto son las variables HTP_ACPT y HTP_R_AEN. El servidor envía al cliente:

- un código de estado que indica si la petición fue correcta o no. Los códigos de error típicos indican que el archivo solicitado no fue encontrado, que la petición no fue realizada de forma correcta o que requiere autenticación para acceder al archivo.
- La información propiamente dicha. Como HTTP permite enviar documentos de todo tipo y formato, es ideal para transmitir multimedia, como gráficos, audio y video. Esta libertad es una de las mayores ventajas de HTTP.

HTTP define 8 métodos (algunas veces referido como "verbos") que indican la acción que desea que sea efectuada sobre el recurso identificado. Lo que este recurso representa, si los datos pre-existentes o datos que son generados de forma dinámica, depende de la aplicación del servidor. A menudo, el recurso corresponde a un archivo o la salida de un ejecutable que residen en el servidor.

A continuación son mostrados los tres que tiene más relevancia para este trabajo de grado

HEAD: Pide que la respuesta idéntica a la que correspondería a una petición GET, pero sin el cuerpo de la respuesta. Esto es útil para la recuperación de meta-información escrita en los encabezados de respuesta, sin tener que transportar todo el contenido.

GET: Pide una representación del recurso especificado. Por seguridad no debería ser usado por aplicaciones que causen efectos ya que transmite información a través de la *(URI,UniformResourceIdentifier)* agregando parámetros a la URL.

Ejemplo: GET /images/logo.png HTTP/1.1 obtiene un recurso llamado logo.png
Ejemplo con parámetros: /index.php?page=main&lang=es

POST: Somete los datos a que sean procesados para el recurso identificado. Los datos quedan incluidos en el cuerpo de la petición. Esto puede resultar en la creación de un nuevo recurso o de las actualizaciones de los recursos existentes o ambas cosas.

En la actualidad existe una propuesta por parte de Apple para desarrollar un protocolo para *streaming* específicamente basado en http denominado http *livestreaming*, esta propuesta nace debido a los inconvenientes que presenta protocolos como RTSP que exigen ciertos recursos para poder ser utilizados. Indicando que para sistemas embebidos o celulares inteligentes el protocolo HTTP es una de las soluciones a futuro más adecuada debido a su simplicidad y adaptabilidad [14].

2.3.5 COMPARACIÓN ENTRE RTSP Y HTTP (NIVEL DE APLICACIÓN)

El modelo TCP/IP combina todos los aspectos relacionados con las aplicaciones en una sola capa y asegura que estos datos estén correctamente empaquetados antes de que pasen a la siguiente capa, específicamente en lo referente al Streaming es posible encontrar que usualmente los protocolos RTSP y HTTP son usados con mayor frecuencia, cada uno aporta características y ventajas para un proceso de Streaming Multicast o Unicast.

RTSP busca proporcionar, para prestaciones multimedia, los mismos servicios que HTTP proporciona para textos y gráficos, de hecho fue desarrollado con una sintaxis similar de tal modo que la mayoría de mecanismos de extensión de HTTP pueden aplicarse a RTSP. Por otra parte RTSP difiere de HTTP en varios aspectos (Tabla 1).

Tabla 1. Diferencias entre HTTP y RTSP

Protocolo	Difusión	Servidor Dedicado	VoD	Difusión	Estados	Simétrico	ID	Amigable con Firewalls	Sintaxis	Control de Contenido
RTSP	Multicast *Unicast	X	X		X	X	X		Similar	X
HTTP	Unicast *Multicast			X				X	Similar	

La razón por la cual RTSP mantiene los estados de la sesión, cuando esta es establecida, es para enlazar las diferentes peticiones con los Stream relacionados de esta manera tener control sobre el contenido multimedia. Por otra parte la razón de la simetría del protocolo HTTP es por su sencillez, a diferencia de RTSP donde la interactividad que proporciona al cliente basada en nuevos métodos como Pause, FF, REW, REC exige un flujo constante de peticiones desde el servidor hacia el cliente y viceversa.

Teniendo en cuenta las características que ofrece RTSP, su utilización puede justificarse cuando la sesión Streaming es establecida y tiene como fin el flujo de contenido multimedia (Video y Audio) y en algunos casos la información complementaria, debido a que normalmente el Streaming y los datos de control viajan por canales separados razón por la cual es necesario una sincronización de los mismos. Sin embargo cuando el contenido Streaming es solamente Audio, como nuestro caso, es válida la utilización de HTTP.

2.3.6 PAUTAS Y SELECCIÓN DE PROTOCOLOS

Por lo descrito anteriormente vemos la posibilidad de acceder al Streaming de audio de manera directa (en vivo), o por demanda (contenido almacenado). Pero es necesario aclarar que el tipo de acceso depende en gran medida de los contenidos dispuestos por parte de la emisora. También cabe resaltar que la difusión que la emisora Radio Universidad del Cauca proporciona es del tipo Multicast por ello inicialmente es planteado que el tipo de Streaming al cual acceder desde el IRD es directo para lo cual el funcionamiento sería el siguiente:

1. Se recibe la transmisión de la emisora.
2. Comienza la emisión.
3. Espera petición de los clientes (IRD).
4. Cuando recibe la petición decide si la acepta (control de admisión).
5. Establece conexión con cliente (IRD).
6. La única interacción del usuario está definida por la función de selección de contenido que se implementó (transmite determinado flujo de información, no transmite).
7. Finaliza la conexión cuando el usuario lo solicite
8. Finaliza la conexión cuando deja de recibir señal desde la emisora

Después de explorar las ventajas de los protocolos mencionados fue decidido que en el nivel de aplicación HTTP sería la solución más conveniente, porque es considerado que aunque no sea uno de los métodos más utilizados para el Streaming de Audio, es un protocolo que tiene implícito una gran sencillez haciendo que el establecimiento de conexión con el servidor sea bastante sencillo, proceso que puede resumirse en una serie de peticiones como es lo usual en HTTP. Por otra parte los protocolos RSTP y RTP tienen un número mayor de cabeceras haciendo que la interpretación de las mismas por parte del microcontrolador requiera de mayor nivel de procesamiento situación que es evitada con el uso del protocolo HTTP ya que este disminuye la complejidad en el manejo cabeceras.

El protocolo UDP ofrece una forma de conexión práctica y comprensible para implementar como protocolo en la capa de transporte para el prototipo IRD, por otra parte al utilizar el protocolo TCP nos podría brindar las ventajas de establecimiento de la conexión propias del protocolo pero es claro que debe implementarse junto con él, un manejo de tiempos que permitan hacerlo más eficiente para el tráfico de Streaming de Audio.

2.4 Dispositivos de radio por internet (IRD)

Un Dispositivo de Radio por Internet (IRD, Internet Radio Device) es un hardware que autónomamente recibe y reproduce audio desde una estación de radio a través de Internet. Los dispositivos existentes en el mercado soportan los formatos de Streaming más comunes como son: MP3 (MPEG Audio Layer 3) y WMA (Windows Media Audio) algunas otras unidades soportan RealAudio y formatos menos comunes como Ogg (Formato libre), AAC (AC-2). La importancia de estos formatos de Streaming o códigos de comprensión radica en permitir fácilmente la interpretación y comprensión del Streaming de datos que es transferido en un flujo de paquetes y su eficiencia está determinada por la simplicidad que ofrece para que el dispositivo de usuario que pueda ejecutarse en tiempo real y la velocidad de conexión sea lo suficientemente rápida para que el proceso de Streaming de Audio sea fluido [15].

Los dispositivos de radio por Internet generalmente soportan listas de reproducción en formatos como: M3U, PLS y por otro lado soportan protocolos de Streaming de Audio como: SHOUTcast, Icecast, y MMS/MMSH (*Microsoft Media Streaming Protocol/Over HTTP*).

Algunos Dispositivos de Radio por Internet tienen conexión para banda ancha a través de interfaces de redes como Ethernet o IEEE 802.11 (ISO/IEC 8802-11), con TCP/IP como soporte de acceso a Internet. Algunas características adicionales que están en los IRD actualmente refieren a distintas interfaces y mecanismos de interacción con el dispositivo, encontrando que la gran mayoría cuenta con un despliegue en una Pantalla de Cristal Líquido (LCD, LiquidCrystalDisplay), puerto de bus universal en serie (USB, Universal Serial Bus) que puede ser usado como conector externo para reproductores MP3 o dispositivos de almacenamiento para archivos MP3, Memorias Flash (SDHC, Secure Digital High Capacity) .

Al realizar la investigación acerca de los dispositivos para la reproducción de flujos de audio encontramos que un factor común es su costo el cual es considerablemente alto, debido a que cuentan con funciones adicionales, con distintas interfaces las cuales incrementan su precio. A continuación están listadas algunas de las empresas más representativas que producen estos dispositivos y una breve descripción de los mismos.

La empresa AWOX ofrece una amplia gama de dispositivos para la reproducción de audio a través de Internet, una de sus principales soluciones la han denominado Radio por Internet [16], este dispositivo puede ser conectado a cualquier red doméstica con el fin de dar a las personas un acceso rápido a la música que desea escuchar. Soporta conexiones Ethernet por cable y Wi-Fi, así como USB y es compatible con DLNA (Digital Living

Network Alliance), ello significa que no presenta problemas de conexión con otros dispositivos compatibles con DLNA. El precio de la solución es USD$145.

Philips ha desarrollado un reproductor que permite escuchar radio por Internet, además permite el intercambio de música entre dispositivos que tengan conexión Wi-Fi [17]. El precio de este dispositivo es USD$ 170.

La empresa Rokuha diseñado un sistema de música completo con conexión Wi-Fi denominado Sound Bridge radio, permite escuchar cualquier emisora en el mundo. Este dispositivo presenta una avanzada funcionalidad de flujos de música digital, altavoces estéreo, subwoofer, radio AM / FM y reloj despertador. Es una solución caracterizada por el potencial de funcionar en Internet y la facilidad de usarlo [18]. El precio de la solución es USD$299.

De esta forma es evidente que los sistemas embebidos con las características propuestas son en su mayoría sistemas propietarios de elevados costos, a pesar de la existencia de herramientas de carácter libre como Sistemas Operativos y microcontroladores de bajo costo. Es de destacar que el precio de un dispositivo como los de los proveedores Roku, Philips, Awox, entre muchos más, están alrededor de los USD$140 alcanzando en algunos casos los USD$300. Es claro que al contar con funcionalidades extras, además de la reproducción de los flujos de audio, hace que su valor sea superior y frente al prototipo propuesto para el desarrollo en este proyecto los costos disminuirían considerablemente.

2.5 SISTEMA MÍNIMO DE PROPÓSITO GENERAL

Es un sistema que hace uso de un mínimo de componentes como son: memoria RAM, memoria ROM, periféricos, actuadores para realizar funciones o rutinas. Sus aplicaciones pueden ubicarse en campos diferentes como por ejemplo la automatización de procesos y dentro de estos desempeñar tareas específicas como: instrumentación, control, monitoreo, señalización, comunicaciones y procesamiento de señales como es nuestro caso. Para tener una idea de la composición de un sistema mínimo a continuación se tiene un diagrama en bloques (Ver Figura 3), donde están representados sus elementos así como la forma en que interactúan entre ellos.

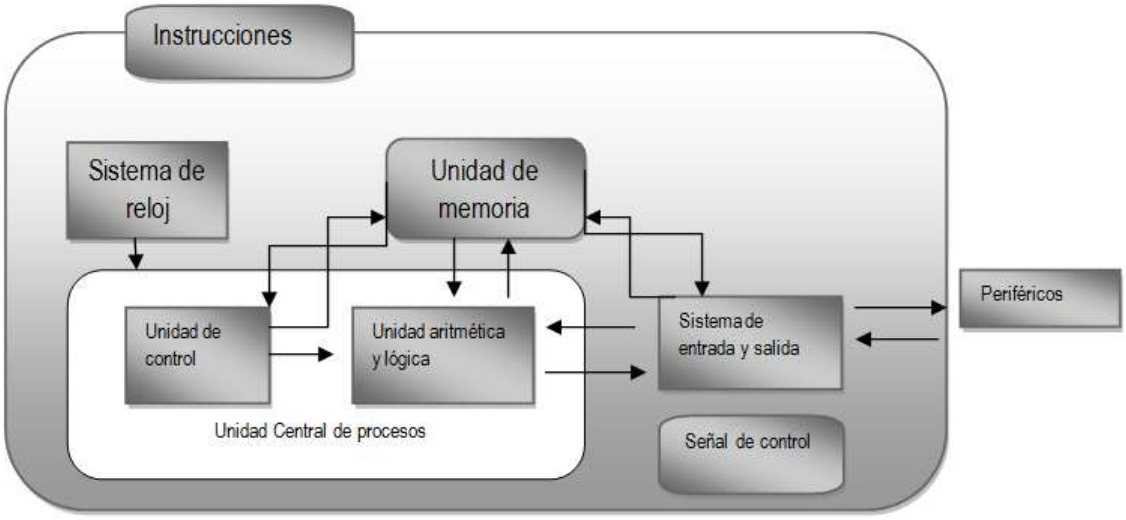

Figura 3. Diagrama de Bloques para un Sistema Mínimo Básico.

Los Microcontroladores son utilizados para el desarrollo de Sistemas Mínimos, estos dispositivos, en adelante llamados MCU, contienen: una CPU, puertos paralelos E/S (Entrada y Salida), temporizadores, memorias, conversores ADC (de señales analógicas a señales digitales o viceversa), para ellosexiste una sección dentro de este capítulo dedicada a su estudio y sus funcionalidades orientadas a las necesidades de este proyecto.

2.5.1 MICROCONTROLADORES

Un microcontrolador es un circuito integrado o chip que incluye en su interior las tres unidades funcionales de un computador: Unidad Central de Procesamiento (CPU), Memoria y Unidades de Entrada/Salida. Son diseñados para disminuir tanto el tamaño de los dispositivos como el costo y el consumo de energía de un sistema en particular. El propósito fundamental de los microcontroladores es el de leer, interpretar y ejecutar los programas que el usuario escribe. El carácter programable de los microcontroladores simplifica el diseño de circuitos electrónicos, permitiendo modularidad y flexibilidad, ya que un mismo circuito pueden ser utilizados para diferentes funciones con solo cambiar el programa en el microcontrolador.

Hoy en día los microcontroladores ofrecen nuevas características, nuevos periféricos y nuevas estructuras. La tecnología Flash en la memoria de programa logra una mayor eficiencia permitiendo programar y borrar la memoria en la propia placa del sistema (*In SystemProgramming, ISP*). También permite la reprogramación de aplicaciones en la

misma placa (*In-ApplicationProgramming, IAP*). La incorporación de circuitos (*PhaseLoockedLoop, PLL*) en el oscilador posibilita la utilización de cristales de baja frecuencia, los supervisores a nivel hardware y software representa una mejora importante reduciendo el número de componentes externos, por último las nuevas tecnologías del silicio permiten tener cada vez encapsulados más pequeños reduciendo el tamaño y el costo del producto final [19].

Las aplicaciones de los microcontroladores son diversas, estando limitadas por la imaginación del usuario. Es común encontrar microcontroladores en campos como la robótica y la automática, en la industria del entretenimiento, en las telecomunicaciones, en la instrumentación, en el hogar, en la industria automotriz. Es estimado que por cada procesador de propósito general vendido más de 100 microcontroladores son vendidos, que la tasa de mercado asociada a estos dispositivos es más del 50% del total de procesadores, y por si fuera poco, la tasa de ventas es estimada que aumente en los próximos años [20].

2.5.2 COMPONENTES DE LOS MICROCONTROLADORES

Los microcontroladores son la evolución natural de la tecnología de la microelectrónica de los microprocesadores. Un microprocesador esta cimentado sobre una CPU donde el bus de datos, el bus de direcciones y el bus de control salen al exterior, en ellos son conectados los periféricos necesarios para realizar un sistema [19] (Ver Figura 4).

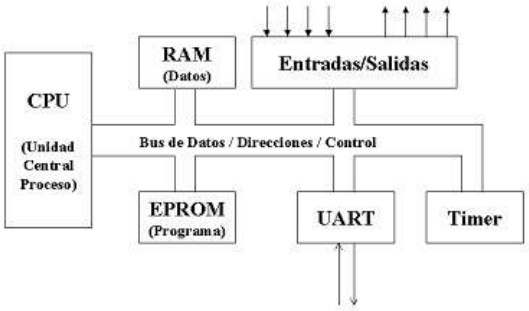

Figura 4. Microprocesador

Por su parte un microcontrolador integra una cantidad de periféricos, así como el bus de datos (Direcciones y Control) y permite tener un dispositivo para cada solución. La siguiente figura muestra la composición básica de cualquier microcontrolador (Ver Figura 5)

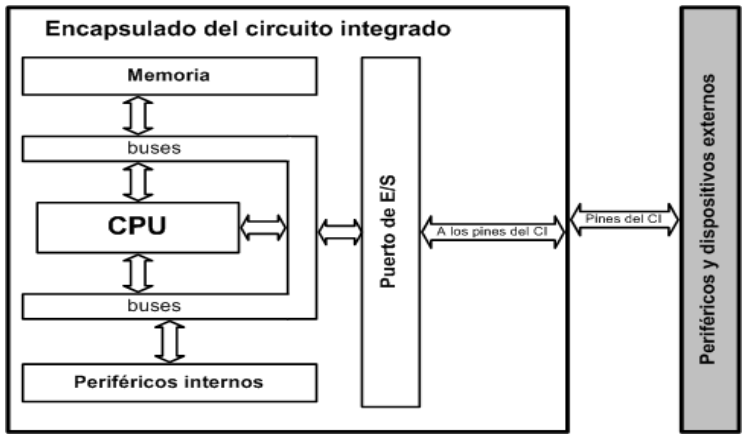

Figura 5. Estructura básica de un microcontrolador

Los microcontroladores requieren de un programa para que realice una función específica. Este es almacenado normalmente en la memoria de solo lectura (ROM, *Read-OnlyMemory*). Un microcontrolador típico tendrá un generador de reloj integrado y una pequeña memoria de acceso aleatorio (RAM, *Random Access Memory*), además cuenta opcionalmente con ROM programable borrable solo de lectura (EPROM, *EraseableProgrammableRead-OnlyMemory*), una ROM programable y borrable eléctricamente (EEPROM, *Electrically-EraseableProgrammableRead-OnlyMemory*) ó una memoria FLASH. Los microcontroladores disponen generalmente también de una gran variedad de dispositivos de entrada/salida, como conversor analógico a digital, temporizadores, Transmisor-Receptor Asíncrono Universal (UART, *Universal AsynchronousReceiver-Transmitter*) y buses de interfaz serie especializados.

2.5.3 Sistemas embebidos

Un sistema embebido es una combinación de hardware y software de computadora, sumado tal vez a unas piezas mecánicas o de otro tipo, diseñado para tener una función específica [19]; una de sus características más relevantes es la flexibilidad, ofreciendo la posibilidad de realizar modificaciones sencillas, en unas líneas de código del software del sistema embebido, reduciendo los costos que implican remplazar un circuito integrado.

Un uso bastante común para los sistemas embebidos es su implementación en sistemas de tiempo real, donde el control del tiempo es vital para obtener un correcto funcionamiento. Los sistemas en tiempo real necesitan realizar cálculos en un límite de tiempo; aspecto que debe ser considerado en el momento de escoger un microcontrolador capaz de proveer al sistema de tal nivel de procesamiento, particularmente este proyecto tiene como objetivo diseñar e implementar un dispositivo para la reproducción de Streaming de audio sobre IP, ello implica capturar la señal de Streaming de manera continua, aproximándose a un sistema en tiempo real.

La principal diferencia entre un sistema embebido y un PC tradicional consiste en la falta de adaptabilidad relativa del sistema embebido. Es decir, el sistema embebido (aunque cada vez ofrezca mayor versatilidad y potencia) está destinado a una actividad concreta y poco cambiante [21]. Un sistema embebido ofrecerá una base hardware básico, medianamente fijo y alejado de la típica modularidad de los sistemas PC. Por otro lado la utilización de un sistema orientado a una aplicación concreta permite ahorrar en costos y evitar excesos hardware (es decir, la utilización de procesadores, memorias sobredimensionadas). Según Jon Catsoulis [21] es posible dividir los diferentes niveles de un sistema computacional así:

Hardware: Correspondiente al nivel más bajo, sobre el cual son controlados físicamente los dispositivos, directamente junto al procesador. Los elementos están relacionados físicamente mediante señales eléctricas, la programación y control es de forma directa por el diseñador.

Firmware: Capa software, contiene datos de programa que permanecen almacenados permanentemente en el sistema. Estos permiten inicializar el hardware y sus subsistemas relacionados. Principalmente, esta capa contiene el *Bootloader*: un programa específico de arranque ejecutado sobre el procesador y que permite la inicialización del sistema operativo y los elementos necesarios para poder pasar el testigo de control del sistema a elementos de control superiores como un sistema operativo u otro software de control.

En sistemas embebidos de menor complejidad, el *Bootloader* está acompañado (o incluso substituido) por el conocido *firmware*, una capa normalmente destinada a evitar la utilización del sistema operativo. El *firmware* carga el programa de funcionamiento básico del sistema, normalmente orientado a sistemas destinados a ejecutar una única función. Así, según la complejidad del firmware podemos encontrarnos desde un sencillo *Bootloader* que prepare el paso hacia el sistema operativo hasta un programa completo de control del sistema orientado a una tarea específica.

Sistema Operativo: capa situada exclusivamente en sistemas embebidos más avanzados. El sistema operativo controla el conjunto de procesos desde un punto de vista elevado y global. Controla la utilización de la memoria y todos los dispositivos de interfaz tales como el teclado, los discos y todos los sistemas externos a él. El sistema operativo ofrece un conjunto de herramientas software para controlar los distintos elementos por parte del usuario. No todos los sistemas embebidos utilizan esta capa, como comentamos anteriormente, sino que solo sistemas avanzados u orientados a multitarea precisarían de esta capa. Aun así, la mayoría de sistemas consistiría en una simple aplicación *firmware* que controle el único proceso del sistema.

Aplicación: capa final y más elevada del sistema (aunque a veces comparta funcionamiento con la capa de *firmware*) consiste en los programas que añaden

funcionalidad a todas las capas anteriores. Resulta la más cercana al usuario, permitiendo la interacción más sencilla del programador con el sistema apoyándose en el buen funcionamiento de las capas inferiores.

Aun así, los niveles anteriormente comentados, podrían definir cualquier sistema computacional, como un sistema de control industrial, un PC o un sistema embebido (Ver Figura. 6).

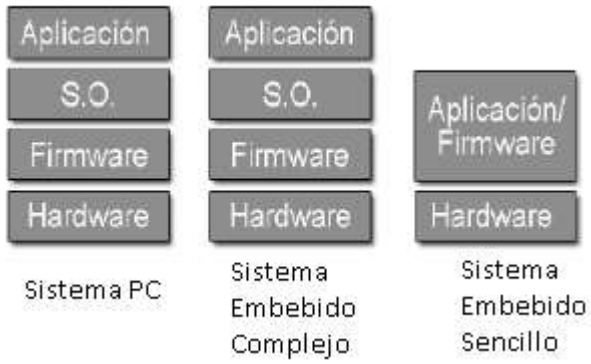

Figura 6. Estratificación de diseño software diferenciada según el sistema a utilizar

2.6 HERRAMIENTAS PARA EL DESARROLLO DE SISTEMAS CON MICROCONTROLADORES

Las herramientas para el desarrollo de aplicaciones soportadas sobre microcontroladores están formadas por programas, aplicaciones e interfaces que permiten implementar de manera más eficiente un proyecto como el que es mostrado en este documento. Algunas de las herramientas para el desarrollo de sistemas basados en microcontroladores son las siguientes [22].

- Ensamblador: la programación bajo este lenguaje puede resultar compleja, pero es considerada una de las más eficientes porque aprovecha al máximo los recursos del microcontrolador al otorgar al programador el dominio absoluto del sistema.

- Compilador: la programación en lenguajes de alto nivel como C o BASIC permite disminuir el tiempo requerido de desarrollo. En el momento de llevarlo a su equivalente en lenguaje ensamblador, las líneas de código en lenguaje de alto nivel generan muchas más líneas de código en lenguaje ensamblador, normalmente en una relación de uno a tres. Para lo cual es necesario contar con un microcontrolador con una capacidad de memoria considerable.

- Simulador: es un software capaz de realizar en un PC las rutinas y programas realizados para un microcontrolador. Permiten tener el control total de la ejecución de

programa, además de realizar la depuración de errores. Existen algunos inconvenientes con la simulación de entrada y salida de datos.

- Placas de evaluación: Son tarjetas con un microcontrolador dispuesto con la mayoría de periféricos conectados, como LED'S, pantalla LCD, teclados, memorias, sensores. También es habitual encontrar una interfaz para conectarla a un PC y descargar los programas correspondientes.

- Emuladores en circuito: es un intermediario que se debe conectarse entre el PC y el zócalo de la tarjeta del circuito impreso donde está el sitio para el microcontrolador. De esta forma el programa o código es ejecutado desde el PC pero es posible obtener los parámetros de respuesta como si fuera ejecutado desde el mismo microcontrolador.

- Programador: es un dispositivo que conectado a un PC permite grabar en el microcontrolador el programa desarrollado. Otra posibilidad es utilizar un cargador de arranque, muy útil en la etapa de desarrollo de un programa, es un pequeño programa en el microcontrolador que está montado en la placa del circuito bajo desarrollo y que puede comunicarse con las herramientas de desarrollo a través de un enlace serie, como puede ser RS232, USB, I2C o un bus CAN**.**

- Paquetes IDE: actualmente existen paquetes de software denominados Entornos de Desarrollo Integrado, IDE, que suelen funcionar bajo plataformas Windows aunque existen versiones para sistemas operativos como Linux, estas últimas resultan bastante eficientes; incluyen editores de texto para el ensamblador o el compilador, permiten la simulación del programa y también pueden integrar el control de emuladores y programadores de dispositivos. Ejemplos de estos entornos de desarrollo son MPLAB de Microchip que permite programar en lenguaje ensamblado y PCWH de la casa CCS que incluye un compilador C para los microcontroladores PIC de Microchip [23].

CAPITULO III. DESARROLLO HARDWARE

INTRODUCCIÓN

El presente capítulo describe las etapas en las que está divido el desarrollo del sistema, inicialmente muestra el diseño Hardware, bosqueja su funcionamiento, dando a conocer las especificaciones, es decir, lo que el sistema debe realizar, a partir de esto presenta un diseño modular en donde esta identificadas cada una de las secciones en las cuales puede dividirse el sistema general esto permite abordar de una manera más específica las diferentes fases que puede requerir el diseño hardware. Después de tener claridad sobre las fases necesarias para el desarrollo del Sistema, muestra un estudio comparativo de los elementos requeridos. Específicamente para la selección del microcontrolador es necesario tener claridad sobre los recursos que van a ser usados (número de entradas/salidas, convertidores A/D o D/A, comparadores, puertos de comunicación serie, memoria de datos RAM y EEPROM, etc.). Para finalizar en la etapa de desarrollo hardware realiza la integración de los elementos seleccionados a través de su conexión lógica y física por medio de un diagrama de circuito total, el cual puede verse en el Anexo 1 y el Anexo 2.

El siguiente diagrama (Ver Figura 7) se ofrece mayor claridad frente al proceso de diseño hardware, en él pueden identificarse las 3 etapas planteadas anteriormente así como las etapas complementarias para el desarrollo total del diseño hardware.

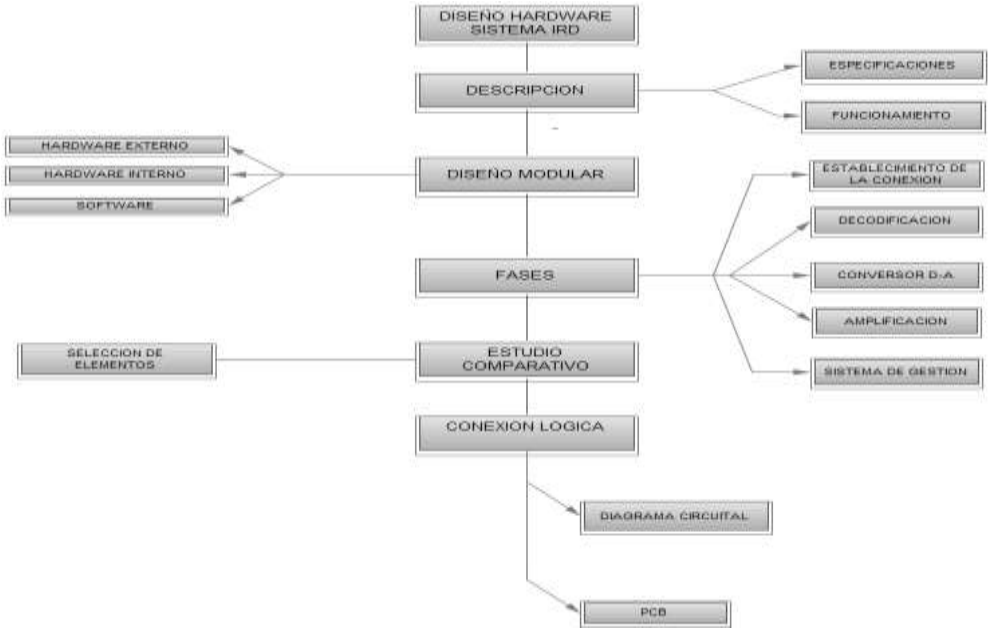

Figura 7. Diagrama de Etapas para el Diseño Hardware

3.1 DISEÑO HARDWARE

Aunque existen sistemas embebidos con procesadores muy avanzados, gran cantidad de estos sistemas basan su núcleo hardware, no en un procesador (de gran velocidad y prestaciones de cálculo) sino en microcontroladores. Ellos, capaces de gestionar procesos aunque no de forma tan dinámica como los procesadores, ofrecen la capacidad integrada de interfaz con elementos básicos; ahorrando espacio y ofreciendo mejores prestaciones de conexión y comunicación en un mismo dispositivo. Permitiendo, así, adaptarse fácilmente de forma dedicada a una tarea específica sin necesidad de elementos externos de gestión (Ver Figura. 8).

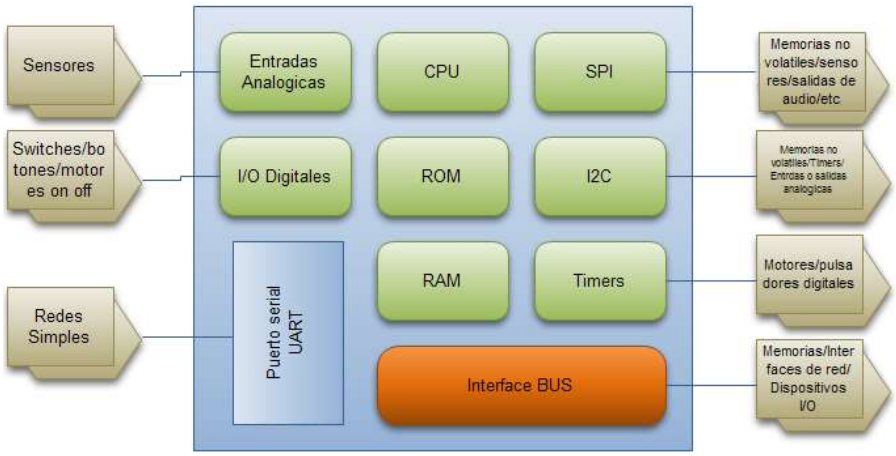

Figura 8. Esquema de un diseño basado en un sistema Embebido

De esta manera, un sistema basado en un microcontrolador como el propuesto, permitirá "on chip" la gestión y manejo de gran cantidad de recursos sin la necesidad de añadir demasiados elementos externos al núcleo. También ofrecerá la capacidad de gestión de diferentes recursos externos, como buses de comunicación (SPI, I2C) e incluso salidas programables (ICSP), que facilitan en gran medida el diseño y programación del sistema. Todo ello teniendo en cuenta que es logrado a cambio de cierta limitación y lentitud computacional.

Al retomar el objetivo general el cual consiste en diseñar e implementar un dispositivo de bajo costo (Sistema Embebido), el sistema basado en un microcontrolador, a expensas de la velocidad, ofrece ahorro de espacio con una capacidad de integración grande, incluyendo en el mismo sistema la mayoría de los elementos básicos requeridos, disminuyendo los costos lo cual es uno de los objetivos y por último ofrece la capacidad de expansión (añadiendo memorias FLASH externas, ROM, RAM, u otros periféricos). Por las razones enunciadas anteriormente el elemento central del sistema es un microcontrolador, sus grandes prestaciones además de ser un elemento de bajo costo y fácil implementación lo ubican como el mejor candidato como elemento principal del

desarrollo, cuenta con la mayoría de recursos, interfaces y periféricos necesarios para implementar un sistema con la descripción dada a continuación.

3.2 DESCRIPCIÓN DEL SISTEMA

El sistema IRD (Internet Radio Device, Dispositivo de Radio Internet), a nivel general puede describirse como una implementación Hardware y Software que capturar y reproduce los flujos de Streaming de Audio generados por un servidor, específicamente este desarrollo trabaja con el Servidor de Radio Universidad del Cauca.

Para establecer una conexión con este servidor el sistema debe contar con una interfaz física que permita conectarlo con Internet y a nivel software debe existir un módulo capaz de manejar esta interfaz mediante diferentes protocolos y establecer la conexión.

Una de las consideraciones más importantes para el manejo de memoria del sistema es a la capacidad de almacenamiento con la que debe contar, esta debe ser lo suficientemente amplia para alojar el modulo software que maneja la interfaz de conexión a Internet, el buffer necesario para la trama de Streaming de Audio y la aplicación cliente del sistema.

Una vez la información está en los buffer de memoria es necesario analizar e interpretar estos datos; teniendo en cuenta que la información en el origen ha sido codificada y comprimida de tal manera que pueda ser transportada por la red, es necesario que el Sistema implemente un módulo hardware y software que realice el proceso inverso que consiste en la decodificación de formatos para así disponer de los datos en su forma básica digital. Posteriormente los datos deben ser convertidos a su forma analógica, para ello debe contar con un módulo hardware de conversión digital-analógica, la calidad obtenida al final de este módulo depende del nivel de resolución para realizar la conversión, finalmente es posible que el nivel de la señal analógica obtenida sea demasiado bajo lo cual hace necesario la implementación de un módulo hardware de amplificación el cual está encargado de llevar la señal a un nivel adecuado.

Finalmente el sistema debe contar con una interfaz mediante la cual el usuario interacciona, está compuesta por interruptores como elemento modificador del comportamiento del sistema y una pantalla. Para ello debe existir un módulo software encargado de manejar estas dos interfaces de usuario y que está en constante interacción con el modulo encargado del almacenamiento y memoria del sistema.

3.3 DISEÑO MODULAR HARDWARE

La división del sistema general en módulos nos permite identificar por separado las características hardware y software bajo las cuales pueden estar descrito básicamente es un diagrama modular secuencial en el cual el resultado al final de un módulo es la condición inicial del siguiente (Ver Figura 9).

Figura 9. Módulos del Sistema IRD

3.3.1 MODULO DE ESTABLECIMIENTO DE CONEXIÓN

El objetivo principal de esta etapa consiste en que el prototipo aquí desarrollado (IRD) debe ser capaz, en primera instancia de conectarse a la red que provee internet a la Universidad del Cauca, posteriormente establecer la conexión con el servidor de audio de la Radio Universidad del Cauca. Por consiguiente dado este escenario surgen interrogantes como:

- ¿Qué elementos hardware debe tener el IRD para hacer la conexión física?
- ¿Qué software es necesario para manejar los elementos hardware y establecer la conexión lógica con la red y con el servidor de audio de la radio Universidad del Cauca?
- ¿Cuáles son los protocolos principales que intervienen en esta etapa?
- ¿Cuál es la interfaz adecuada que debe tener el microcontrolador?

Para poder responder a estos interrogantes es necesario entender cómo funciona la red que provee internet a la Universidad del Cauca y así identificar cuáles son los elementos hardware y software más adecuados para esta etapa. También es planteada una posible interpretación del Módulo para el establecimiento de la conexión (Ver Figura 10).

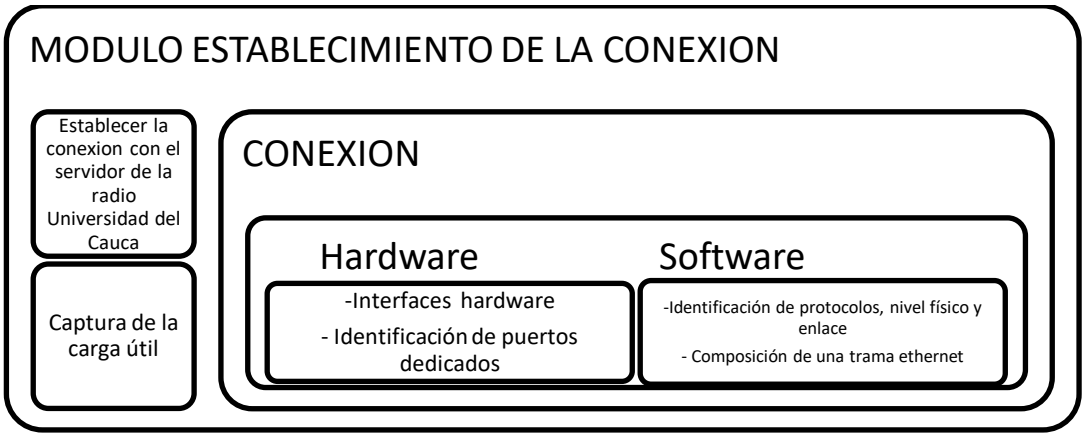

Figura 10. Módulo de Establecimiento de Conexión

La Universidad del Cauca cuenta con una red física que dispone de un backbone en fibra óptica con más de 4 kilómetros de longitud y un sistema de cableado estructurado en cada uno de los edificios, que cubre a la Universidad con 1450 puntos activos y una red corporativa que permite interconexión entre las distintas sedes de la Universidad y el acceso a Internet a través de dos enlaces, uno con Telecom y otro con Orbitel-Emtel.

La tecnología que usa para dar acceso a internet a los diferentes terminales y equipos es conocida con el nombre de Ethernet, el cual es probablemente el estándar más popular para las redes de área local (LANs) [24]. En una configuración Ethernet los equipos están conectados mediante cable coaxial o par trenzado (*Twisted-pair*) y compiten por acceso a la red utilizando un modelo denominado CSMA/CD (*CarrierSenseMultiple Access withCollisionDetection*).

Los estándares Ethernet no necesitan especificar todos los aspectos y funciones necesarios en un Sistema Operativo de Red NOS (*Network OperatingSystem*) como ocurre con otros estándares de red, la especificación Ethernet referencia solamente a las dos primeras capas del modelo OSI (*Open SystemsInterconnectionH12.2*). Estas son la capa física (el cableado y las interfaces físicas), y la de enlace, que proporciona direccionamiento local; detección de errores y controla el acceso a la capa física. Una vez conocidas estas especificaciones nuestro dispositivo está en condiciones de integrarse en una red sin problemas. A continuación los elementos más importantes correspondientes a estas dos capas son mencionados.

En primera instancia debe tenerse muy en cuenta la recomendación IEEE 802.3 [25] para 10 Base T, la cual es la configuración con la que cuenta hoy en día la Universidad del Cauca para dar acceso a los equipos a internet, esta recomendación nos lleva a utilizar un conector RJ45 como elemento pasivo, basado en la recomendación podemos obtener la siguiente información:

- Tecnología = 10BaseT
- Velocidad de transmisión =10 Mbps
- Tipo de cable = Par Trenzado
- Distancia máxima = 100 m (La norma indica que no debe existir más de 100mts entre el primer PIC y el próximo sin usar antes un HUB, repetidor, switch.
- Elementos como: Hub o Switch.

Con base en estos protocolos y basados en la anterior información, el microcontrolador debe contar con uno o varios módulos que le permita realizar las siguientes tareas:

Debe contar con un módulo encargado de codificar y decodificar la información presente en el par trenzado (Canal de comunicación) durante un enlace Ethernet. Este módulo toma los cambios en el voltaje del canal RX y los convierte a paquetes de datos digitales para procesarlos en el microcontrolador y también toma los datos que quiere transmitir TX el microcontrolador y los codifica a sus respectivos niveles de voltaje para poder ser enviados a través del par trenzado (canal de comunicación).

Debe contar con un módulo que permita implementar el estándar IEEE 802.3, basado en una dirección MAC, el cual toma los datos a transmitir y los empaqueta ("arma" la trama de datos) de forma que cumpla con esa norma. Caso inverso con los datos que llegan donde debe "desarmar" la trama Ethernet y obtener la carga útil que en este caso es la información de audio a reproducir. Generalmente este tipo de dispositivos utilizan lo que es conocido como interfaz de gestión independiente del medio (*MIIM, Media Independend Interface Management*) y posee comunicación con el modulo anterior.

Es importante contar con una memoria única y exclusivamente como Buffer para almacenar temporalmente los paquetes de datos que llegan o que serán transmitidos. Debido a que la capacidad de transporte de la carga útil de una trama Ethernet es de (0 – 1500 bytes) y para Streaming es necesario contar con un almacenamiento de por lo menos 8000 bytes esta memoria debe tener una capacidad de 8 Kbyte equivalente a 8 tramas Ethernet aproximadamente para así evitar saltos a la hora de reproducir el audio. Además debe contar con un módulo que permita controlar el acceso a la memoria RAM descrita anteriormente y atender a los llamados del Procesador del PIC, del módulo DMA para transmitir y del canal RX del MAC. Para terminar contar con un módulo que indique el estado de la comunicación y de los demás procesos.

3.3.2 MODULO DE PROCESAMIENTO

Como fue mencionado en la descripción del sistema el elemento principal del desarrollo debe ser un Microcontrolador, este ejecuta directamente el código principal de la aplicación, proporciona el control y la comunicación entre los diferentes elementos que componen el sistema y las capacidades para el establecimiento de la conexión a la red.

Específicamente el elemento está dentro del Módulo de Procesamiento, las funciones que cumple van desde la interpretación del Stack TCP/IP, el manejo de periféricos como por ejemplo el decodificador, conversores D/A, puertos de entrada y salida, por último el manejo de los recursos de memoria (Ver Figura 11).

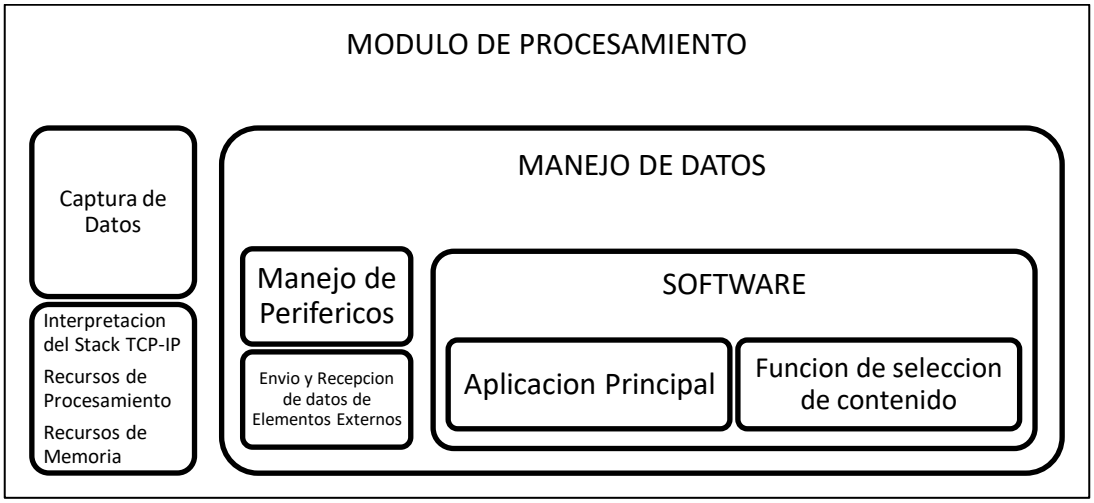

Figura 11. Módulo de Procesamiento

Los datos de audio recibidos desde el servidor de Radio Universidad del Cauca en el Sistema desarrollado son en forma de Streaming a través de una conexión vía internet con el servidor que inicia en el módulo de Establecimiento de Conexión y por medio de un protocolo conocido para ambas partes, Servidor y Cliente inician él envió de datos al Sistema desarrollado. Estos datos serán almacenados en un buffer establecido previamente, la fluidez de la reproducción del Streaming de Audio depende en gran medida de la capacidad del buffer con la que cuenta el Sistema y del ancho de banda disponible en la red, como fue mencionado anteriormente los sistemas embebidos cuyo elemento central es un microcontrolador cuentan con recursos de memoria reducidos porque sus aplicaciones así lo definen, pero específicamente para una aplicación de Streaming es necesario contar con un espacio en memoria mayor, esto es muy importante al momento de definir el diseño inicial del Sistema puede ser necesario recurrir a elementos de almacenamiento externos como por ejemplo SRAM o memoria FLASH.

La estructura básica de funcionamiento del Módulo de Procesamiento consiste en el manejo de los datos de Streaming como producto final del Módulo de Conexión, esta conexión como ya sabemos es realizada mediante el intercambio de determinada información según el protocolo usado, posteriormente el servidor volcará de forma continua los datos sobre el socket del cliente, que deberá obtener esta información a tiempo, sin capacidad de retransmisión por perdida de paquete o en caso de errores, seguidamente el modulo realiza dos tareas de forma simultánea, ofrece de manera visual la información correspondiente a los datos obtenidos mediante el Modulo de Interfaz de

Usuario y a su vez entrega un resultado al Módulo de Decodificación para su posterior reproducción.

Con el fin de obtener una reproducción fluida y de calidad es necesario tener en cuenta dos condiciones como son:

- Protocolos ligeros: entendiendo ligeros como protocolos rápidos, que permitan intercambio sin demoras, teniendo en cuenta que esta mejora en la velocidad es obtenida a cambio de la deficiencia en la corrección de errores. Por tanto la utilización de protocolos de transporte como UDP o protocolo de nivel de aplicación HTTP resultan muy recomendables por su agilidad en la transmisión.

- Rapidez en la precarga: Aunque para la reproducción de contenido Streaming no es necesario la totalidad de la descarga en el destino, la fragilidad en el entorno de comunicación bien sea por errores en el servidor, errores en el cliente o simples retrasos en la transmisión, puede provocar una reproducción excesivamente cortada. Para solventar esta situación el sistema debe contar con un buffer de entrada que permita aliviar estos fallos durante la comunicación.

Para cumplir con estas dos condiciones es necesario realizar una selección entre los protocolos que intervienen en el proceso de establecimiento, transporte y envió de datos Streaming, es por ello que bajo el Modulo de Procesamiento encontramos quizás el elemento Software más importante del Sistema conocido como Stack TCP-IP, en él están definidos todos aquellos protocolos que intervienen en el proceso de captura de Streaming de Audio.

La elección de un formato de codificación de audio para la transmisión como podrá verse en la descripción del siguiente modulo, cumple un papel importante en la obtención de una precarga rápida, acompañado esto del protocolo UDP permite una transmisión rápida y sencilla de los datos codificados. Por encima de ellos esta establecida la última capa de comunicación basada en una variedad de capas software, tan diferentes como el amplio conjunto de programas destinados a los sistemas de Streaming desde programas Propietarios como *last.fm*[26]o programas genéricos de Streaming de Audio como *Icecast*[27] o como en el caso del Sistema en desarrollo una conexión concreta a un servidor para la obtención de los datos (generalmente servidores FTP o HTTP).

Teniendo en cuenta la aplicación donde será utilizado el microcontrolador necesita de un procesamiento rápido y continuo de datos, por ser una aplicación en tiempo real, es necesario en este módulo tener en cuenta conceptos como nivel de procesamiento del microcontrolador. Existen diferentes arquitecturas como por ejemplo de 4, 8, 16, 32 número de registros de bits también conocido como el tamaño de palabra que pueden manejar para las operaciones, por ejemplo, un microcontrolador de 8 bits maneja

variables de 8 bits o más explícitamente valores de 0 a 255 (00h a FFh), si se desea realizase cálculos matemáticos u operaciones más complejas es necesario realizarlas en dos partes. Por otra parte al usar un microcontrolador de 16 bits, el cual maneja palabras del doble de capacidad, valores numéricos más grandes en una sola operación (de 0 a 65535 o FFFFh), haciéndolo más rápido en ese tipo de operaciones, uno de 32 bits maneja valores hasta de 4, 294, 967,295 (0 a FFFFFFFFh) en una sola operación. Es entonces necesario considerar la utilización de MCU's con una capacidad de procesamiento adecuada, por ejemplo de 16 bits, aunque existen microcontroladores como los de la familia PIC18F de Microchip que cuentan con un procesamiento de 8 bits mejorados que podrían ser una buena opción ya que son elementos de bajo costo.

Para realizar el manejo de dispositivos externos el microcontrolador cuenta con interfaces como SPI (*Serial Periferical Interface*) la cual permite realizar la conexión y manipulación de elementos como el Decodificador, memorias externas, etc.

SPI (*Serial Periferical Interface*)

Esta interfaz esta compuesta de un Bus de datos de tres líneas, sobre el cual son trnsmitidos paquetes de 8 bits, dos de estas líneas transfieren los datos (una en cada dirección) y la tercera línea el reloj. Cada una de estas tres líneas porta la información entre los diferentes dispositivos conectados por medio de esta interfaz al sistema desarrollado, esta comunicación es de tipo full dúplex por lo que los dispositivos conectados por lo general actúan como receptores y transmisores como por ejemplo el Modulo de Decodificación y de ser necesario la utilización de memorias externas son conectados por medio de esta interfaz.

Es necesario definir en el momento de la selección de elementos que componen el Modulo de Procesamiento como en el caso del microcontrolador (Hardware Interno) y Decodificador y Memorias externas (Hardware Externo), la manera en que deben comportarse con respecto a la interfaz SPI, ya que su conexión puede realizarse de dos formas: Los dispositivos conectados al Bus pueden ser definidos como Maestros cuando son ellos quienes inician la transferencia de información sobre el Bus y generan la señal de reloj y control, o pueden ser definidos como esclavos cuando están controlados por el Maestro a través de una línea selectora denominada *Chip Selector o Select Slave*, por lo tanto un esclavo es activado solo cuando esta línea está activa

La señal sobre la línea del reloj se denomina (SCLK), es generada por el Maestro y sincroniza la transferencia de datos. La línea MOSI (Master Out Slave In) o comúnmente llamada SI, transporta los datos desde el maestro hacia el esclavo. La línea MISO (Master In Slave Out) también llamada SO, transporta los datos desde el esclavo hacia el maestro (Ver Figura 12).

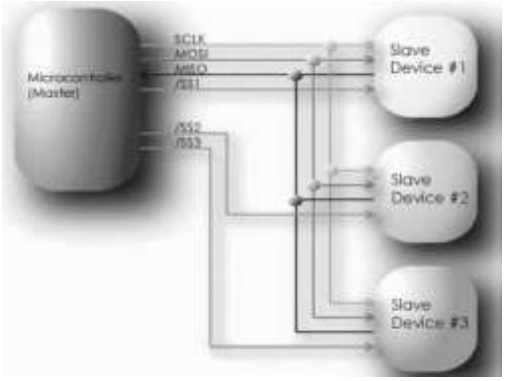

Figura 12. Conexión SPI (Mater / Slave)

3.3.3 MODULO DE INTERFAZ DE USUARIO

La implementación de este módulo está directamente relacionada con la función de selección de contenido a implementar sobre el prototipo Hardware, ya que como tal la interfaz de usuario puede de igual forma clasificarse entre la interfaz local y la interfaz remota (Ver Figura 13).

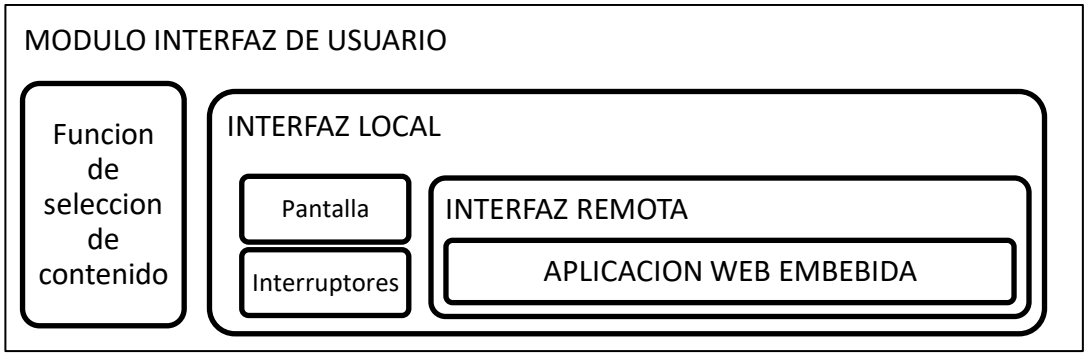

Figura 13. Módulo Interfaz de Usuario

Para la interfaz de usuario local es necesario contar con un elemento que permita visualizar la información capturada por el dispositivo, como por ejemplo la información adicional de los datos de Streaming de Audio (nombre del servidor, nombre de la emisora, etc.) a través de un pantalla (LCD) para el despliegue de información además es necesario añadir elementos con los cuales pueda interactuar se con el dispositivo, para ello son necesarios una serie de interruptores que cumplen con varias funciones, entre ellas modificar los niveles de volumen y específicamente, permiten al usuario identificarse ante la función de selección de contenidos y navegar entre diferentes emisoras. La interfaz de usuario remota será planteada en la sección referida al desarrollo Software del Sistema IRD, pero es necesario aclarar que esta incluye dentro de este módulo por tener relación con el Modulo de Procesamiento.

3.3.4 MODULO DE DECODIFICACIÓN

Este módulo una vez tenga la carga útil almacenada parcialmente en los buffer de memoria, con la información de audio a reproducir, es esencial determinar en qué formato esta información y decodificarla con su respectivo códec. Debido a que los servidores de audio generalmente codifican la información en cualquiera de los siguientes formatos: MP3 (*MPEG Layer 3*), WMA (*Windows Media Audio*), AAC (*Advanced Audio Coding*),etc, es necesario analizar en qué formato esta la información en el servidor de la Radio Universidad del Cauca y escoger el dispositivo hardware y software que haga esta función con el mayor rendimiento, desempeño y que cumpla con el criterio de bajo costo que busca este proyecto (Ver Figura. 14).

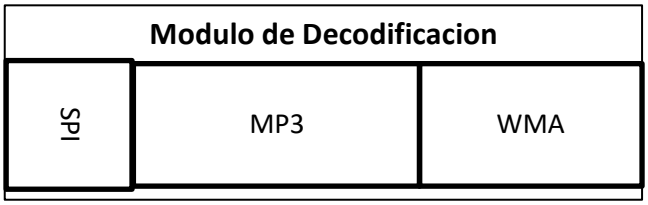

Figura14. Modulo de Decodificación

Al describir este módulo es necesario hacer referencia a la combinación de Hardware como el elemento decodificador y Software como la aplicación encargada de identificar el tipo de codificación y el direccionamiento que debe darle a la información desde donde toma los datos es decir del Módulo de Procesamiento hasta el siguiente modulo. También es necesario hablar sobre los Códec, estos implementan un algoritmo que permite codificar y decodificar los datos auditivos buscando reducir la cantidad de bits que ocupa un archivo de audio, con el fin de comprimir señales de audio en un flujo de datos (Audio Streaming), para que estas ocupen un menor espacio. En el momento de su decodificación se traducen desde el formato en el que están codificados por ejemplo MPEG (I, II, III), WMA, ACC; para luego ser manipulados en un formato más apropiado.

La comprensión de archivos de sonido realiza por medio de algoritmos. Para lograr la reducción del archivo utiliza una técnica conocida como PNS (Norma de Percepción de Ruido). Es considerado de percepción porque la mayoría de formatos de audio aprovechan características del oído humano para diseñar la comprensión de los algoritmos que dan forma a un archivo de sonido.

Un códec de audio es un tipo de códec diseñado específicamente para la compresión y descompresión de señales de sonido audibles para el ser humano. Los códec de audio cumplen la función de reducir el número de datos digitales necesarios para reproducir una señal auditiva, lo que es denominado comprensión de datos aplicado a un fin concreto por

ejemplo la reproducción de sonido almacenado o el proveniente de una transmisión a través de IP como en el presente caso.

Para poder reproducir un archivo de audio es necesario el códec con el que fue comprimido, debido a que cada códec tiene su algoritmo particular para realizar el proceso de compresión y descomprensión, este varia particularmente para cada tipo de códec. Debe considerarse entonces como un parámetro de selección para nuestro elemento códec, teniendo en cuenta que los formatos que utiliza el Stream de la emisora Radio Universidad de Cauca son MPEG3 y WMA[28].

Los códec pueden caracterizarse por los siguientes parámetros:

- Número de canales: un flujo de datos codificado puede contener una o más señales de audio simultáneamente. De manera que puede tratarse de audiciones "mono" (un canal), "estéreo" (dos canales, lo más habitual) o multicanal. Los códec de audio multicanal suelen utilizarse en sistemas de entretenimiento "cine en casa" ofreciendo seis (5.1) u ocho (7.1) canales.

- Frecuencia de muestreo: de acuerdo con el teorema de Nyquist, determina la calidad percibida a través de la máxima frecuencia que es capaz de codificar, que es precisamente la mitad de la frecuencia de muestreo. Por tanto, cuanto mayor sea la frecuencia de muestreo, mayor será la fidelidad del sonido obtenido respecto a la señal de audio original. Por ejemplo, para codificar sonido con calidad CD nunca se usan frecuencias de muestreo superiores a 44,1 kHz, ya que el oído humano no es capaz de escuchar frecuencias superiores a 22 kHz.

- Número de bits por muestra. Determina la precisión con la que puede reproducirse la señal original y el rango dinámico de la misma. Suelen utilizarse 8 (para un rango dinámico de hasta 45 dB), 16 (para un rango dinámico de hasta 90 dB como el formato CD) o 24 bits por muestra (para 109 a 120 dB de rango dinámico). El más común es 16 bits.

- Pérdida. Algunos códec pueden eliminar frecuencias de la señal original que, teóricamente, son inaudibles para el ser humano. De esta manera puede reducirse la frecuencia de muestreo. En este caso se dice que es un códec con pérdida o lossy códec (en inglés). En caso contrario es un códec sin pérdida o lossless códec (en inglés).

A continuación esta una comparación entre las principales características de los formatos manejados por la Radio Universidad del Cauca como son MPEG 3 y WMA. En la Tabla 2 están resumidas a manera de comparación ambos formatos.

MPEG-1 Audio Layer 3

Más conocido como MP3, es un formato de audio digital comprimido con pérdida desarrollado por el Moving Picture ExpertsGroup (MPEG) para formar parte de la versión 1 (y posteriormente ampliado en la versión 2) del formato de vídeo MPEG. El mp3 estándar es de 44 kHz y un bitrate de 128 Kbps por la relación de calidad/tamaño. Aun así, con el paso de los años y el aumento de velocidad de transferencia de archivos por red, cada vez es más común encontrar archivos de MP3 con un bitrate superior a 320 Kbps En dicha calidad, la diferencia entre la canción original y el archivo podría decirse que es ínfima [29].

WMA

Es la abreviación de Windows Media Audio. Códec privativo desarrollado por *Microsoft,* de naturaleza más avanzado que el MP3 permite la utilización de multicanales así como audio de alta definición. Establece su comprensión a través de teoría psicoacusticas, basada en la supresión de información situada en frecuencias no audibles para el oído humano. Además permite calidades de comprensión semejantes a las de MP3 pero con un nivel de comprensión mayor

Tabla 2. Comparación entre los códec MPEG III y WMA

Codec	MP3	WMA
• Tipo de comprension	• Con perdidas	• Con perdida, sinperdida
• Frecuencia de muestreo	• 8, 11.025, 12, 16, 22.05, 24, 32, 44.1, 48kHz	• 8, 11.025, 12, 16, 22.05, 32, 44.1, 48, 96kHz
• Tasa de bits	• 8, 16, 24, 32, 40, 48, 56, 64, 80, 96, 112, 128, 160, 192, 224, 256, 320 Kbps	• 4-768kbps / variable (sin pérdidas)
• Bits por muestra	• Cualquiera	• 16, 24 (modo sin pérdidas) / Cualquiera (modo con pérdidas)
• CBR	• Si	• Si
• VBR	• Si	• Si
• Multicanal	• No	• hasta 8 canales (WMA Professional) / hasta 6 canales (WMA sin pérdidas)

Después de seleccionar el protocolo de transporte y la forma de obtención de datos en el Modulo de Procesamiento, resta definir el formato de comprensión audio sobre el cual va a trabajar en este módulo, teniendo en cuenta que el Streaming de Audio generado desde la Radio Universidad del Cauca, están en ambos formatos WMA y MP3, es válido considerar una solución Hardware que nos ofrezca la facilidad de manejar ambos formatos, por lo tanto hasta el momento la solución válida ofrecida para el diseño del

Sistema final IRD sería una combinación de las mejores alternativas expuestas hasta el momento en lo referente a protocolos, interfaz para la conexión de elementos externos y formatos de comprensión: (Protocolo de transporte UDP además del Stack necesario TCP-IP, con obtención de datos por http y manejo de formatos MP3 y WMA, así como manejo de interfaz SPI para la conexión de elementos externos).

3.3.5 MODULO DE REPRODUCCIÓN

Este módulo está compuesto a su vez por dos módulos los cuales pueden describirse como los responsables del proceso de reproducción de Audio ya que son encargados de generar una señal analógica y amplificarla.

Módulo de Conversión D/A (procesamiento de señal)

Debido a que hasta el momento la información está en forma digital, debe realizarse el proceso de conversión, el cual consiste en pasar de una señal digital a una señal analógica, para esto existe múltiples técnicas y dispositivos los cuales deberán evaluarse y así determinar cuál es la solución hardware y software más adecuada para esta etapa. Básicamente la conversión digital a analógica (D/A) es el proceso de tomar un valor representado en código digital y convertirlo en un voltaje o corriente que es proporcional al valor digital [30]. Un DAC contiene normalmente una red resistiva divisora de tensión que tiene una tensión de referencia estable y una fija como entrada. Para un convertidor Digital - Analógico de 4 bits cada entrada digital puede ser sólo un "0" o un "1" el voltaje de salida analógica tendrá uno de 16 posibles valores dados por una de las 16 combinaciones de la entrada digital.

La resolución en convertidores D/A puede definirse de dos maneras: primero por el número máximo de bits de salida (la salida digital), este dato permite determinar el número máximo de combinaciones en la salida digital, el número máximo está dado por: 2n donde n es el número de bits. También la resolución es entendida como el voltaje necesario (señal analógica) para lograr que en la salida (señal digital) haya un cambio del bit menos significativo (LSB). Para hallar la resolución utilizada la siguiente fórmula: Resolución = VoFS / [2n - 1] donde:

- n = número de bits del ADC
- VoFS = es el voltaje que hay que poner a la entrada del convertidor para obtener una conversión máxima (todas las salidas son "1")

Los conversores D/A más comunes de este tipo son de 8 y de 12 bits; un conversor de 8 bits permite una resolución de 256, es decir, para un intervalo de conversión 0-10 V a

cada unidad le corresponden aproximadamente 40 mV; la resolución de un conversor de 12 bits es de 4096 pasos, 2.5 mV.

Módulo de Amplificación

La etapa de amplificador de potencia o etapa de ganancia son los nombres utilizados para referirse a un amplificador de audio en un diseño como el que aquí es planteado. Su función es aumentar el nivel de una señal, incrementando, para ello, la amplitud de la señal de entrada mediante corriente de polarización (voltaje negativo, voltaje positivo) en el transistor de salida.

En la actualidad existen integrados dedicados, donde pueden encontrarse las etapas de decodificación, conversión y amplificación en un solo elemento, lo cual puede resultar bastante conveniente debido a que simplifica el diseño así como la implementación hardware al no tener que implementar los tres módulos por separado.

3.4 ESTUDIO COMPARATIVO DE MICROCONTROLADORES

El microcontrolador es parte esencial en el diseño e implementación del dispositivo, debido a que es el encargado de integrar el componente software con el componente hardware del sistema, además, su selección será basada en costo Vs funcionalidad, disponibilidad en el mercado y soporte por parte del proveedor, todo esto sumado al objetivo propuesto de desarrollar un prototipo de bajo costo y buen desempeño; Para ello fue realizado un estudio comparativo de los diferentes MCU existentes en el mercado teniendo en cuenta las siguientes características: Controlador Ethernet integrado, Modulo ADC, Procesador 8/16/32 bit, Memoria 256kb en adelante, Herramientas Software libres.

El estudio fue realizado basándose en la información ofrecida por los siguientes proveedores: Microchip, Texas Instrument, Renesas, Atmel, Philips (NXP), Freescale Se espera que el costo promedio del microcontrolador sea de $ 2 US a $ 5 US y que cuente en lo posible con el manejo de datos y de multimedia (Audio), por esta razón la tabla comparativa (Tabla 3), reúne la información general de los MCU citados a continuación, ella hace referencia a estas características.

Texas Instruments

Las características encontradas sobre estos microcontroladores: LM3S6100 y LM3S6420, no especifican módulos para conversor D/A lo que supone la utilización de un componente externo; teniendo en cuenta que ambos dispositivos son de implementación superficial debe contarse con algún medio para la adaptación como una tarjeta (PCB). Para la programación del dispositivo es ofrecida documentación, código fuente, librerías y algunos paquetes por parte del proveedor [31].

NXP (Philips)

Reúnen varias de las características propuestas para suplir las necesidades del proyecto entre ellas manejo de datos Ethernet y salida analógica para parlantes con el correspondiente conversores D/A. Por parte de ambas empresas son manejados *ARM7TDMI® core y Cortex™-M3 core*. El precio está fuera del rango de lo propuesto entre 12 USD y 17 USD. Existe un amplio soporte como documentación y paquetes de implementación software aunque mucha de la información requiere de licencias que deben ser compradas. Existen distribuidores de estos dispositivos para Colombia como ArrowElectronics Inc. VaultInformationServices LLC. [32]

RenesasTecnology

La familia súper H, serie SH7780 cuenta con una serie de dispositivos de la familia SH7763 de 32 bits con CPU SH-4A core. Su precio es considerablemente alto así que es posible que aunque su desempeño y características son las más idóneas debido a que puede ser utilizado en aplicaciones como HAC, SSI, Tarjetas de interfaz multimedia, interfaz de Streaming; no es objeto de consideración.

La serie SH-Ether pertenece a la familia Súper H conservando sus características esenciales como su CPU, cuenta con una serie de 6 dispositivos (grupos SH7516 SH7616, SH7618, SH7619, SH7670, SH7710) con uno o dos canales controladores de Ethernet que cumplen con el estándar IEEE 802.3u. Su principal diferencia radica en la memoria RAM ya que va desde 4k - 8k - 16k - 32k. En general estos dispositivos podrían considerarse una opción correcta considerando que es conseguido un ahorro en la implementación de módulos externos como memorias o un módulo controlador de Ethernet así como conversores D/A, en adelante y considerando las altas prestaciones de los MCU como lo son la serie Súper H de Renesas y la familia LCP 2300 de NXP. Es posible considerar un MCU de mayor costo con una combinación de módulos externos de bajo costo. El microcontrolador R5F212E2NFP se ajusta al precio establecido además cuenta con modulo para conversión D/A pero no cuenta con un controlador Ethernet [33]

Dallas Semiconductor

Los MCU de Maxis Dallas Semiconductor MAXQ7666 son conocidos por su buen desempeño, esta solución cuenta con el módulo de conversión D/A además de una amplia memoria SRAM, a pesar de esto no posee un controlador de Ethernet y su costo es elevado frente a algunas otras soluciones que en general ofrecen características similares [34].

FreeScale

La familia MCF5225X consiste de una serie de dispositivos altamente integrados on-chip Ethernet, USB, CAN ideal para aplicaciones en red, además cuenta con funciones de cifrado. Cuenta con un software como complemento que no tiene costo (Freescale MQX RTOS software). ExisteunaampliadocumentaciónMCF5225x Family Fact Sheet*MCF52259*

ColdFire® Microcontroller DataSheetMCF52259 ColdFire® Integrated Microcontroller Reference Manual [35].

Microchip

Los microcontroladores desarrollados por Microchip son unos de los MCU más conocidos y usados en el área de la Electrónica, cuentan con una amplia documentación y aplicación en desarrollo, además que pueden ser adquiridos por un precio razonablemente económico con relación a su buen desempeño. La referencia incluida en este estudio comparativo de microcontroladores es el PIC18F67J60 perteneciente a la familia de los PIC18, esta referencia introduce una nueva línea de dispositivos de bajo voltaje con todas las características adicionales que poseen los microcontroladores PIC18. Dentro de sus características principales es importante mencionar que este microcontrolador tiene integrado el controlador Ethernet lo cual provee de una ventaja evitando la utilización de dispositivos externos para llevar a cabo esta tarea, podría decirse que su desarrollo por parte de Microchip se debió la incursión de los MCU en dispositivos finales de una red. Debido a que trabajar con un flujo de datos continuo (Streaming) es necesario que el buffer de memoria con el que cuenta en este caso el controlador Ethernet del MCU tenga la suficiente capacidad de alojar los datos mientras que el procesamiento de los mismos es realizado, haciendo que se considere incluir un módulo externo para memoria lo cual es definido en el siguiente capítulo. Por último podemos mencionar que este microcontrolador no cuenta con el módulo de conversión D/A, es decir, que es necesario un módulo externo que cumpla esta tarea para obtener una señal analógica del proceso de captura de Streaming de Audio a realizar [36].

Tabla 3. Tabla de Comparación De MCU

PROVEEDOR	REFERENCIA	PRECIO ($)	CPU	RAM	TIPO DE ROM	F (MHz)	ETHERNET	PERIFERICOS /SERIAL	ADC
Texas Instruments	LM3S6100	4.65	32 bits RISC ARM CORTEX	16 kb SRAM	64 KB flash	25	10/100 Ethernet	X /I2C/2SSI/ 2SSPI/3UART	A/D 10 bits 8 Ch.
Texas Instruments	LM3S6420	4.95	32 bits RISC ARM CORTEX	32 kb SRAM	96 kb flash	25	10/100 Ethernet	X /I2C/2SSI/ 2SSPI/3UART	A/D 10 bits 8 Ch.
NXP Philips	LCP2364/ 66/68	8.15	16/32 bit ARM7	64 kb SRAM	Hasta 512 kb Flash	72	10/100 Ethernet	X /4UART/2 SSP/1SPI/ USB/2CAN/3I2C/1I2	A/D y D/A 10 bit

								S	
Renesas	SH7780	-	32 bits H-4A Core	4-8-16 32 kb SRAM	-	Hasta 400	10/100 Ethernet	X /USB/2I2C /LCD	A/D
Renesas	R5F212E2NFP	2.92	16/32 bits	512kb	8 kb Flash	20	-	X /UART	A/D 10 bits 12 Ch.
Maxis Dallas SMT	MAXQ7666	6.13	16 bits RISC	512 byte SRAM	16 kb Flash	8	-	X /CAN 2.0/UART	A/D y D/A
FreeScale	MCF5225X	7.43	32 bits ColdFire2 RISC	64 kb SRAM	512 kb Flash	80	10/100 Ethernet	X /QSPI/UART/I2C/USB/CAN	A/D 12 bit 8 Ch.
Microchip	PIC18F67J60	4.55	8 bits RISC	3 kb	128 kb	25	1O BASE T ETHERNET	X /2USART/ 1 MSSP/SPI /I2C	A/D 10 bit 11 Ch.

3.4.1 MÉTRICAS PARA LA SELECCIÓN DEL MICROCONTROLADOR

Las métricas consideradas para la selección del microcontrolador como componente central del diseño son:

- **Precio**: un factor muy importante para obtener un dispositivo final de bajo costo, para ello es establecido un tope máximo de USD$6 sobre el costo de este dispositivo.
- **Disponibilidad en el mercado**: en nuestro medio, ¿Qué tan difícil es encontrar este componente?, se toma en cuenta la disponibilidad del componente en el país.
- **Facilidad de programación y soporte**: es necesario saber qué nivel de complejidad requiere el manejo del firmware al momento de programarlo, de igual forma debe considerarse si es necesario adquirir herramientas adicionales software o hardware.
- **Funcionalidad**: es de esperar que el microcontrolador cuente con los módulos principales en lo posible el módulo de conexión Ethernet evitando así incremento en la complejidad del diseño.

Aplicando el Método Ponderado que consiste en dar un valor cuantitativo a las métricas consideradas, en este caso, valores entre 1 y 10 es obtenida la siguiente tabla (Tabla 4).

Tabla 4. Asignación cuantitativa para las métricas de selección del MCU

Métrica (Factores)	Valor para ponderar	Ponderado (Vp/T)
Precio	10	0,32
Disponibilidad en el mercado	6	0,19
Facilidad de programación y soporte	8	0,26
Funcionalidad	7	0,23
TOTAL	31	1,00

De acuerdo a la información técnica presentada en la Tabla 4 se determinó una calificación o costo (Tabla 5), del 1 al 5 correspondiente a Insuficiente, Regular, Bueno, Muy bueno y Sobresaliente respectivamente. Esta información fue obtenida de las páginas oficiales de los fabricantes.

Tabla 5. Calificaciones y Costos de los MCU

Factor / (Alternativa)	Precio USD	Disponibilidad en el mercado	Facilidad de programación y soporte	Funcionalidad
LM3S6100	4.65	3	3	2
LM3S6420	4.95	3	3	2
LCP2364/66/68	9.15	3	3	4
R5F212E2NFP	2.92	2	3	2
MAXQ7666	6.13	2	2	3
MCF5225X	7.43	3	3	4
PIC18F67J60	4.55	3	5	5

Para obtener la calificación ponderada se multiplica el valor ya ponderado por la calificación asignada en la tabla anterior. Para los valores en dólares se selección el valor menor, a ese valor le es asignado la mayor calificación, es decir10. A continuación los excedentes de los demás valores son obtenidos y aplica una regla de tres. Los procesos mencionados son resumidos en la tabla (Tabla 6)

Tabla 6. Matriz de Puntos de los MCU

Factores	**Ponderacion**	**LM3S6100**		**LM3S6420**		**LCP2364/66/68**		**R5F212E2NFP**		**MAXQ7666**		**MCF5225X**		**PIC18F67J60**	
		Calif	Calif P	Calif	Calif P	Calif	Calif P	Calif	Calif P	Calif	Calif P	Calif	Calif P	Calif	Calif P
Precio	0,32	5	1,6	5	1,6	2	0,64	10	3,2	4	1,28	3	0,96	9	2,88
Disponibilidad en el mercado	0,19	3	0,57	3	0,57	3	0,57	2	0,38	2	0,38	3	0,57	3	0,57
Facilidad de programacion	0,26	3	0,78	3	0,78	3	0,78	3	0,26	2	0,52	3	0,78	5	1,3
Funcionalidad	0,23	2	0,46	2	0,46	4	0,92	2	0,46	3	0,69	4	0,92	5	1,15
TOTAL			**3,41**		**3,41**		**2,91**		**4,3**		**2,87**		**3,23**		**5,59**

Como puede apreciarse en la Tabla 6, los puntajes obtenidos se comportan de manera similar en la gran mayoría de casos siendo el precio un criterio determinante en el momento de la selección. El mayor puntaje lo obtuvo el microcontrolador de la familia PIC18 de Microchip debido a sus grandes ventajas en cada uno de los criterios planteados.

3.5 ESTUDIO COMPARATIVO (DECODIFICACIÓN, CONVERSIÓN Y AMPLIFICACIÓN)

La etapa de decodificación del formato sobre el cual viene codificado el Streaming de Audio bien sea MPEG 3, WMA o AAC entre otros, podría considerarse una etapa bastante importante y definitiva por dos razones, primero es aquí junto con la etapa de Conversión D/A donde en definitiva es obtenida una mayor o menor calidad de la señal analógica a reproducir y segundo, porque en el estudio comparativo realizado anteriormente es claro que la mayoría de los microcontroladores cuentan con un módulo de conversión A/D, pero aquellos microcontroladores de menor costo no cuentan con este periférico, además el Codificador, etapa que fue mencionado por su importancia en la calidad que imprime a la señal analógica a obtener, no forma parte de los elementos comunes con los que cuentan estos microcontroladores, haciendo que necesariamente esta sea una etapa externa al sistema.

A continuación son mencionados los dispositivos considerados como elementos a tener en cuenta para el diseño del prototipo, la elección está basada en las características ofrecidas como por ejemplo manejo de formatos de compresión, por efectos prácticos debe buscarse que por lo menos maneje MPEG3 y WMA. Para el módulo de conversión Digital-Análoga es de esperarse que la resolución manejada sea de mínimo 8 bit.

PT8406 (MP3/WMA Decoder IC)
Este circuito integrado desarrollado Princeton TechnologyCorp es capaz de decodificar todos los formatos de audio MPEG y todas las frecuencias de muestreo, como por ejemplo: MPEG1 (32/44.1/48 kHz), MPEG 2 (16/22.05/24 kHz) y MPEG 2.5. Otro de los formatos capaz de decodificar este integrado es el conocido como WMA (V2/V7/V8/V9). Maneja una resolución de 16 a 24 bit y sus aplicaciones van desde dispositivos reproductores MP3 y WMA, PDA, diccionarios electrónicos y todas las aplicaciones MP3/WMA. [37]

STA013 (MP3 Decoder IC)
Este integrado altamente flexible desarrollado por STMicroelectronics es un decodificador de audio MPEG nivel 3, específicamente MPEG1 y MPEG2 ISO, MPEG 2.5. Utiliza una interfaz serie para el ingreso de datos, después de realizar el proceso de decodificación la señal obtenida está disponible tanto en un canal o dos canales para ser enviada

directamente a un convertidor digital-análogo por la interfaz de salida PCM con la que cuenta. Es utilizado en el desarrollo de tarjetas de sonido para computadores y dispositivos multimedia en general.

Existen diferentes empaquetados o presentaciones de este circuito integrado desde SO de 28 pines como los empaquetados superficiales TQPF de 44 pines [38].

VS1053b (Decodec IC)

Este controlador es una solución bastante robusta como Decodec, desarrollado por VLSI Solution, es capaz de decodificar AAC, ADPCM, FLAC, MIDI, MP1-MP2-MP3, OGG, PCM, WAV y MWA, además codifica OGG y ADPCM. Como puede verse mediante este integrado es posible manejar gran cantidad de formatos incluidos los que por requerimiento debe manejar el prototipo. Posee una memoria RAM dedicada tanto para instrucciones como para datos. Cuenta con un control serial de datos de entrada mediante una interfaz de 8 pines de entrada y salida considerados como de propósito general. Tiene incluido un módulo para conversión tanto D-A como A-D, en ambos casos estéreo y por último implementa una etapa de amplificación para auriculares con salida estéreo de 30 ohm.

Este tipo de dispositivos son una solución bastante conveniente para diseños como el planteado en este proyecto debido a sus grandes prestaciones como decodificador además de las funcionalidades que puede aportar al prototipo y que a futuro pueden considerarse así como el ahorro en implementación Hardware ya posee las 3 etapas definidas como son Decodificación, Conversión y Amplificación en un solo modulo [39].

ROHM (16 bit DAC)

Este circuito integrado desarrollado por ROHM Semiconductor and Electronic, proporciona una conversión a 16 bits estéreo es decir que la salida analógica cuenta con 2 canales. Este producto se usa en equipos electrónicos típicos donde es necesario realizar la etapa de conversión digital – analógica (Equipos audio visuales, equipos de oficina, dispositivos de comunicaciones, aplicaciones electrónicas) [40].

MCP4725/28 (12 BIT DAC)

Es un integrado de baja potencia y alta precisión desarrollado por Microchip, cuenta con un solo canal y con una conversión a 12 bit. Los datos de entrada y de configuración de este DAC pueden ser programados por la memoria no volátil EEPROM, tiene como función permitir que el DAC mantenga su código de entrada mientras no está en funcionamiento y recuperarlo después de una inactividad. Esta característica es muy útil cuando el dispositivo es utilizado como soporte para otros dispositivos en la Red [41].

3.5.1 MÉTRICAS PARA LA SELECCIÓN DEL HARDWARE (DECODIFICACIÓN – CONVERSIÓN - AMPLIFICACIÓN)

Al igual que en la selección del microcontrolador y dada la importancia de los componentes a seleccionar en lo referente a las etapas de decodificación, conversión y amplificación, fue seguido el mismo Método de Ponderación donde las métricas fueron (Tabla 7):

- Funcionalidad: esta métrica es considerada importante porque busca reducir la complejidad del circuito final, en este caso debe esperarse que el elemento reúna las tres funciones (decodificación, conversión y amplificación).
- Formato y Resolución: como fue expuesto en secciones anteriores el Streaming de audio se encuentra codificado bajo diferentes formatos y debe esperarse que como mínimo el elemento decodificador soporte formatos como MP3, WMA.

El precio sigue siendo un objetivo al realizar un dispositivo de bajo costo, aunque esto puede variar dependiendo de las características del elemento y porque la mayoría de estos elementos son comprados por cantidad lo cual hace que el precio varié.

Tabla 7. Asignación cuantitativa para las métricas de selección del Módulo de Reproducción

Metrica (Factores)	**Valor para ponderar**	**Ponderado (Vp/T)**
Funcionalidad	10	0,59
Formato y Resolucion	7	0,41
TOTAL	17	1,00

Según la información técnica sobre los componentes evaluados para esta etapa fue determinada una calificación o costo (Tabla 8), del 1 al 5 correspondiente a Insuficiente, Regular, Bueno, Muy bueno y Sobresaliente respectivamente. Esta información fue obtenida de las páginas oficiales de los fabricantes.

Tabla 8. Tabla de evaluación cuantitativa y cualitativa

Alternativa/Factor	**Funcionalidad**	**Formato y Resolucion**
PT8406	3	3
STA013	2	3
VS1053b	5	4
ROHM	2	1
MCP4725/28	2	2

Para obtener la calificación ponderada se multiplica el valor ya ponderado por la calificación asignada en la tabla anterior (Tabla 9).

Tabla 9. Matriz de puntos del Módulo de Reproducción

Factores	Ponderacion	PT8406		STA013		VS1053b		ROHM		MCP4725/28	
		Calif	Calif P	Calif	Calif P	Calif	Calif P	Calif	Calif P	Calif	Calif P
Funcionalidad	0,59	3	1,77	2	1,18	5	2,95	2	1,18	2	1,18
Formato y Resolucion	0,41	3	1,23	3	1,23	4	1,64	1	0,41	2	0,82
TOTAL			3		2,41		4,59		1,59		2

Después de revisar los distintos elementos que podrían hacer parte del módulo de procesamiento del Streaming de Audio encontramos 3 opciones para una posible solución cada una con sus ventajas y desventajas propias:

- Solución 1.Decodec STA013 – DAC MCP4725 – Amplificador LM4810Q
- Solución 2. Decodec STA013 – DAC/Amplificador LM4903
- Solución 3. Decodec/DAC/Amplificador VS1053b

3.6 HARDWARE SELECCIONADO

3.6.1 MICROCONTROLADOR

Después de estudiar las características de los MCU propuestos como objeto de estudio en la sección anterior puede realizarse las siguientes aclaraciones:

- La mayoría de los MCU cumplen en gran medida con las condiciones técnicas necesarias como manejo de datos a través de un controlador Ethernet por lo cual fue necesario incluir más de una característica a la vez como por ejemplo el manejo de memoria.
- Al considerar un MCU de precio relativamente económico fue realizado una preselección, en la que fue tenido en cuenta las herramientas disponibles y la documentación.

PIC18F67J60

Finalmente encontramos que el microcontrolador desarrollado por Microchip cumple con las expectativas para el desarrollo del proyecto; como fue mencionado en el estudio realizado, el PIC de la familia 18F97J60 esta optimizado para aplicaciones embebidas e integra Controlador de Acceso al Medio (*Medium Access Controller, MAC*) y Dispositivo de Capa Física (*PhysicalLayer Device, PHY*). Esta integración con hasta 128 Kbytes de memoria de programa Flash, proporcionando una solución sencilla y económica además de los elementos necesarios para comunicación remota mediante un único chip, (Tabla 3).

Las siguientes son las características con las que cuenta el MCU PIC18F67J60 [36]:

- La sencilla migración realizada por Microchip en sus dispositivos PIC18 añadiendo Ethernet permite un amplio desarrollo en aplicaciones para redes de área local (LAN) con un mínimo coste y tiempo de desarrollo.

- Al poseer un buffer dedicado de 8 Kbyte permite un eficiente almacenamiento de paquetes, su consulta y modificación reduciendo la demanda en el microcontrolador integrado.
- A pesar que necesita memorias externas para albergar el Stack de TCP IP y la aplicación para la función de selección de contenido, la interfaz SPI permite una integración rápida con el microcontrolador, cabe aclarar que el costo las memorias oscila entre USD$ 1 - USD$ 2

Esta última característica es bastante conveniente teniendo en cuenta que las herramientas Software requeridas por algunos otros MCU mencionados en el estudio tienen limitaciones por ser versiones Demo y además que son incluidos dentro de paquetes que rebasan las expectativas de costo para la realización del proyecto. La pila TCP\IP de Microchip, incluye las siguientes características principales (Tabla 10):

- Optimizado para todas las familias PIC18, PIC24 Y PIC32.
- Protocolos: ARP, IP, ICMP, UDP, TCP, DHCP, SNMP, HTTP, FTP, TFTP.
- Enchufe de apoyo para TCP y UDP.
- Servicio de nombre NetBIOS, Secure sockets layer (SSL).
- DNS- Domain Name System, Dispositivos de Ethernet.
- Soportado por compiladores MPLAB C18, C30 y C32.

Tabla 10. Características De Los Dispositivos Pic18f97j60 (64 Pines)

CARACTERISTICAS	PIC18F67J60
Frecuencia de operación	DC – 41.667 MHz
Memoria de programa (bytes)	128 k
Memoria de programa (instrucciones)	65532
Memoria de datos (bytes)	3808
Fuentes de Interrupciones	26
Puertos I/O	Puertos A,B,C,D,E,F,G
Pines I/O	39
Timers	5
Módulos PWM (capturar, comparar)	2
Modulo mejorado PWM	3
Comunicación Serial	MSSP (1), USART mejorado (1)
Comunicación Ethernet (10Base-T)	Si
Puerto paralelo de comunicaciones	No
Bus de memoria externa	No
Modulo A/D	11 canales de entrada
Resets	POR, BOR, RESET Instruction, Stack Full, Stack Underflow, MCLR , WDT (PWRT, OST)
Conjunto de instrucciones	75
Empaquetado	64 pines TQFP
Interfaz SPI	Si

MODULO ETHERNET DEL MICROCONTROLADOR PIC18

La figura identifica los bloques internos que componen el modulo Ethernet del microcontrolador PIC18F67J60 (Ver Figura 15).

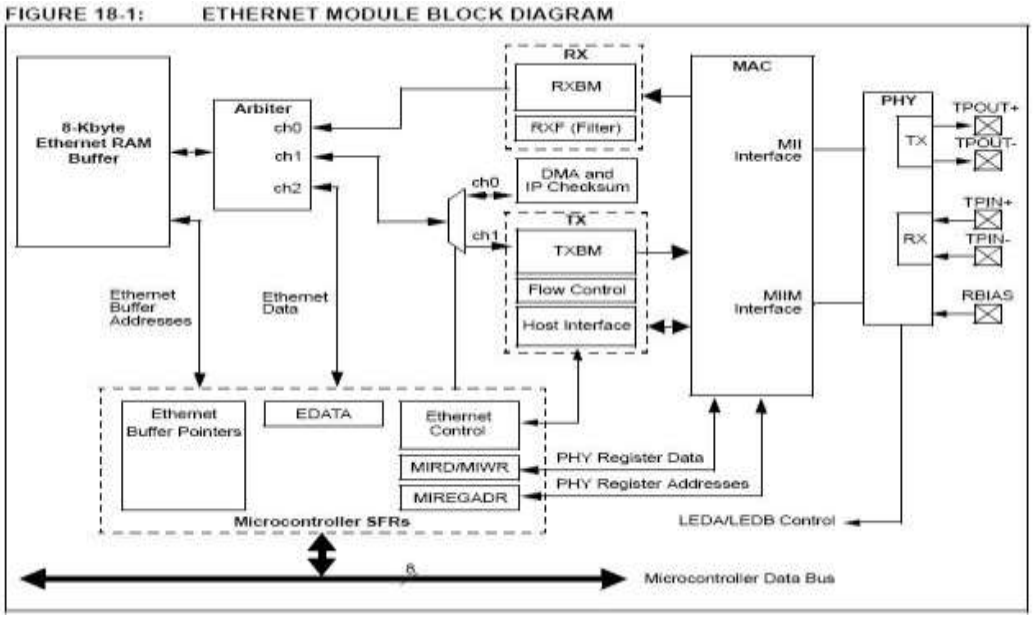

Figura 15. Diagrama de bloque del Módulo Ethernet PIC18F67j60

Este módulo cuenta con las siguientes características:

Módulo transceiver PHY

Esta encargado de codificar y decodificar la información presente en el par trenzado (Canal de comunicación) en un enlace Ethernet. Este módulo toma los cambios en el voltaje del canal RX y los convierte a paquetes de datos digitales para procesarlos en el PIC. Tiene la capacidad de tomar los datos que quiere transmitir el PIC y los codifica a sus respectivos niveles de voltaje.

Modulo MAC

Implementa el estándar IEEE 802.3, toma los datos a transmitir y los empaqueta (arma la trama de datos) basada en los parámetros que indica el estándar. Caso inverso con los datos que llegan a la interfaz. Cuenta con una interfaz conocida como MIIM (Media Independend Interface Management) la cual es la encargada de comunicarse con el módulo PHY.

Memoria RAM 8K Byte

Usada única y exclusivamente como Buffer para almacenar temporalmente los paquetes de datos que llegan o que serán transmitidos.

Arbiter

Controla el acceso a la memoria RAM descrita anteriormente y atiende a los llamados del Procesador del PIC, del módulo DMA para transmitir y del canal RX del MAC.

Registros de propósito específico (SFRs)
Encargados de controlar todo el Modulo Ethernet y de indicar el estado de la comunicación y demás.

3.6.2 MODULO DE PROCESAMIENTO DE STREAMING AUDIO

Después de plantear tres posibles soluciones hardware para implementar dentro de este módulo, en la sección *3.5.1 Métricas para la* Selección del Módulo de Decodificación *e*s evidente que la Solución 1, frente a las otras dos soluciones es menos costosa con un precio total de USD$15.69 podría ser una solución acertada, pero hay que tener en cuenta algunas consideraciones como la calidad que ofrece por ejemplo la solución 3 o quizás la 2 aunque esta última resulta más costosa con un valor total de USD$21.44. La circuitería externa que requiere la solución 1 es quizás una razón de peso que puede influir en su elección como solución para el prototipo ya que al manejar varias etapas hardware por separado los elementos en lo referente a la polarización y adaptación del elemento al circuito final son incrementados.

La facilidad de adaptación de la solución 3 y el manejo de datos que puede realizar este circuito integrado por sí mismo al contar con una memoria RAM dedicada además del control de volumen y algunas otras prestaciones hacen de esta solución una de las más idóneas, teniendo en cuenta también que para adquirir varios de los elementos que hacen parte de la solución 1 y 2 debe hacerse en grandes cantidades en las tiendas que venden estos elementos.

Vs1053b
VS1053b está basado en un procesador digital de señales propietario, VS_DSP. Este contiene todos los códigos y memoria de datos para decodificar los siguientes formatos de audio OggVorbis, MP3, AAC, WMA and WMA PCM +ADPCM. Cuenta también con interfaces seriales, un convertidor digital – análogo estéreo, una salida analógica amplificada con sus correspondientes filtros. También permite realizar codificación de audio ADPCM compatible con una línea de entrada para micrófono con un amplificador y un convertidor analógico – digital. Por último cuenta con una interfaz UART (Universal Asynchronous Receiver-Transmitter, Transmisor-Receptor Asíncrono Universal), (Tabla 11) [39].

Tabla 11. Características del Decodec VS1053b

Referencia	Codificador	Decodificador	DAC	ADC	Amplificador	Memoria	Control de volumen	Procesador
VS1053b	ADPCM	OggVorbis, MP3, AAC, WMA and WMA PCM +ADPCM	*X*	*X*	*X* *Drive de Auricular Estéreo 30 ohm*	*16 KiB RAM(instrucciones) 0.5 KiB RAM(datos)*	*X* *Manejos de Bajos y Brillos*	*VS_DSP 16/32 bit*

3.6.3 MEMORIA SRAM EXTERNA

Memoria Estática de Acceso Aleatorio

La palabra estática indica que la memoria retiene el contenido mientras le sea aplicada alimentación. Acceso aleatorio significa que la localización de las posiciones en la memoria no sigue ningún orden donde los datos serán leídos o escritos. Las celdas de almacenamiento son registros o biestables (flip-flop).

TIPOS

- Asíncronas SRAM: independientes de la frecuencia de reloj los datos de entrada y los datos de salida están controlados por la transición de las direcciones.
- Síncronas SRAM: todas las sincronizaciones son iniciadas por el tiempo de subida y de bajada del reloj. Las direcciones, datos de entrada y otras señales de control están asociadas con las señales del reloj.

FIFO SRAM

FIFO (First In, FirstOut): primero en entrar, primero en salir, es decir el primer dato que se escribe es el primero que se lee.

- Esta memoria se puede utilizar para comunicar dos sistemas con diferentes velocidades, por ejemplo donde el sistema transmisor envía datos a una velocidad superior a la que el receptor la puede aceptar.
- Las memorias FIFO se usan comúnmente como “buffer” y en control de flujo, como en video, comunicaciones de datos, telecomunicaciones, Network Switching /routing.

Las memorias FIFO pueden ser: síncronas,una FIFO síncrona usa el mismo reloj tanto para leer como para escribir la memoria. Asíncronas,una FIFO asíncrona usa diferentes relojes leer y para escribir la memoria.

Existen en una variedad de velocidades y ancho de bus, densidades y encapsulados. También tener las siguientes configuraciones:Unidireccionales, Bidireccionales, Tri-bus, Doble sincronismo.

256K SPI BUS LOW-POWER SERIAL SRAM

Microchip Technology Inc. ha desarrollado unas memorias SRAM 23x256, dispositivos a 256 Kbit. A la memoria es accedida a través de una sencilla interfaz conocida por sus siglas en ingles como (SPI) Serial Peripheral Interface, compatible con bus serial. Las señales requeridas son una entrada de reloj (SCK), además de los datos por separado en (SI) y líneas de salida de datos (SO). El acceso al dispositivo es controlado a través de un chipSelect (CS) de entrada.

La comunicación con el dispositivo puede ser detenido a través del pin HOLD. Mientras el dispositivo está en pausa, transiciones en sus entradas serán ignorados, con laexcepción del pin Chip Select, que permite que el host de servicios con mayor prioridad haga interrupciones. El 23X256 está disponible en los paquetes estándar incluyendo 8-plomo PDIP y SOIC, y avanzado de embalaje, incluidos 8-TSSOP (Tabla 12). Este tipo de memoria es la indicada para el prototipo desarrollado debido a su integración con la interfaz SPI.

Tabla 12. Pines de Funcionamiento

Nombre	Función
cs	Control
so	Datos salida
vss	Tierra
SI	Datos entrada
SCK	Señal de reloj
HOLD	Tiempos de control
VCC	Voltaje

3.6.4 ELEMENTOS GENERALES

Los elementos complementarios a los que se hace referencia son los elementos comunes que hacen parte del diseño, su utilización no altera el funcionamiento general, es por esto que su elección es realizada basándose en elementos de bajo costo. El listado completo de los elementos seleccionados conocido como BOM (Bills of Material, Lista de Materiales) es entregado como Anexo 7 y en él son especificados todos los componentes del sistema, a continuación son mencionados algunos:

LCD

Para visualizar la información y como parte de la interfaz de usuario fue seleccionado un LCD de 2 líneas por 16 caracteres (16x2 LCD Blanco sobre Negro), esta es una pantalla

de alto contraste, profesional y fácil de adquirir en el mercado por su bajo costo, tiene todas las prestaciones para ser parte de cualquier sistema de entretenimiento doméstico o en este caso un sistema de reproducción de Streaming de Audio. Cuenta con un software de retro-iluminación controlable.

RJ45

La interfaz física para la conexión entre el dispositivo y la red se realiza a través del conector RJ45, de uso bastante común en dispositivos de red, quizás la característica especial más relevante en el elemento seleccionado es que incluye elementos como bobinas y juego de resistencias que requiere para su conexión con el microcontrolador. La referencia de este elemento es MIC24010 de WE-MIDCOM [42].

LM1117MPX-3.3TR-ND

Es un regulador de voltaje lineal desarrollado por National Semiconductor que ofrece un voltaje de salida de 3.3 V con una entrada de máxima de 15 V, la corriente de salida máxima es de 0,8 mA y la tensión de alimentación mínima para su funcionamiento es de 4.75 V. El encapsulado es tipo SOT-223. El uso de este integrado es debido a los voltajes que maneja el microcontrolador el cual para su correcto funcionamiento debe tener una alimentación de 3.3 V [43].

La lista de características que describen al IRD (Internet Radio Device, Dispositivo de Radio por Internet) está en (Tabla 13).

Tabla 13. Lista de Características
Sistema IRD

Conexion de Red
10 Base T Ethernet
HTTP Client / Webserver
Time Synchronization w/ NIST Time Std
Configuracion de IP estatica
Baja latencia TCP/IP Stack
Reproduccion
MP3 SHOUTcast Streams
ACC SHOUTcast Streams
Sistema de Gestion
Actualizacion automatica de la informacion
Compatible con la mayoria de navegadores
Flexible HTML

Procesador y Memoria
PIC18F67J60 25 Mhz
MP3 Decoder dedicado
64KB SRAM para Stream Buffering
8KB Ethernet Buffer
Formato de Audio Soportado
MP3 up to 320Kbps
Streaming AAC/MP4
Interfaz de Usuario
2x16 LCD
Interfaz Web de gran alcance
3xSwitch

3.7 DISEÑO LÓGICO DEL SISTEMA

Este diagrama muestra la conexión lógica entre cada una de las fases con cada uno de los puertos correspondientes destinados para su interconexión, una vez asignado los puertos relacionados con cada módulo el diagrama circuital es desarrollo (Ver Figura. 16), (Ver Figura. 17).

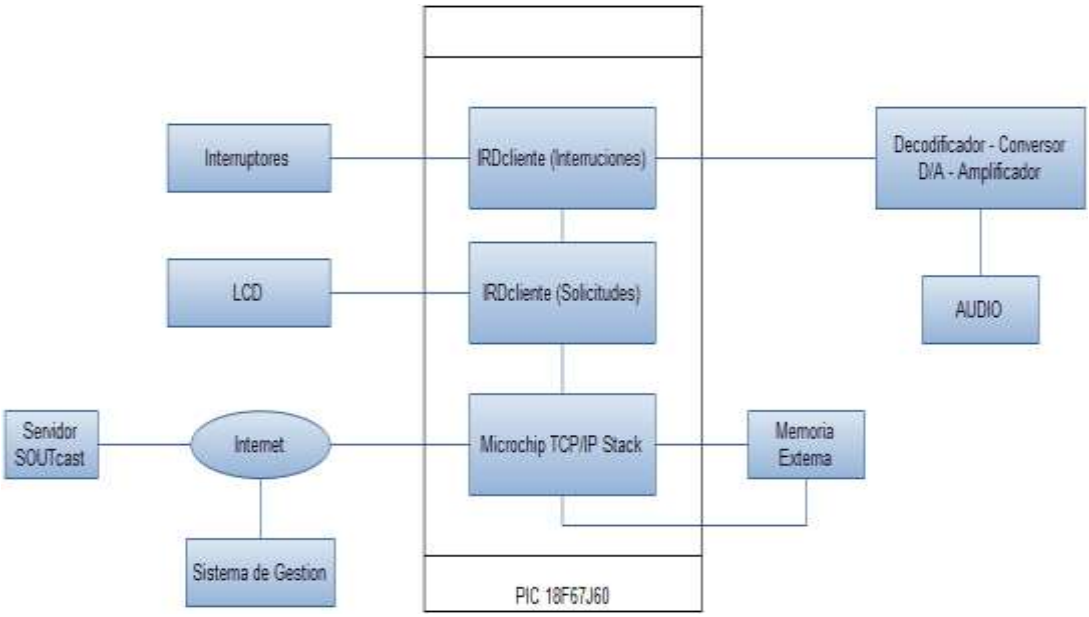

Figura 16. Diagrama en bloques del Sistema para la Captura y Reproducción de Audio Streaming

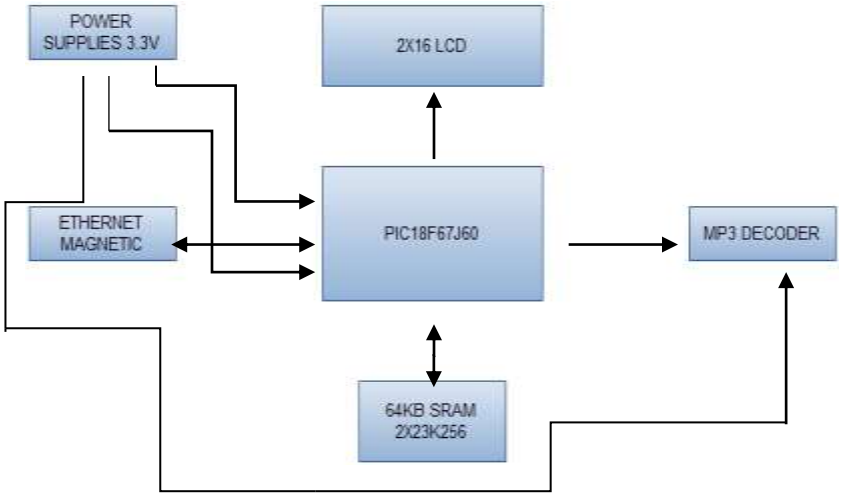

Figura 17. Diagrama de Bloques IRD

3.7.1 CONEXIÓN LÓGICA DEL MICROCONTROLADOR CON EL RJ45

Esta fase debe analizar cuáles son los puertos que necesita el RJ45 (Magjack) escogido para nuestro diseño y el microcontrolador pic18f67j60 para poder ser interconectados. Para realizar esta conexión debemos dirigirnos a los *datasheets*de cada uno de los elementos y mirar cómo debe ser conectado. Tabla 14

El RJ45 presenta 4 pines que deben ser conectados al microcontrolador pic18f67j60 (Ver Figura. 18), los cuales vienen especificados en el microcontrolador para este tipo de módulos a continuación podemos observar cómo se realiza la conexión es realizada, los otros pines deben conectarse a resistencias y capacitores, circuitería que es necesaria para el funcionamiento de este elemento, es muy importante la debida polarización y tierras para el correcto funcionamiento de este elemento. A continuación es presentado el esquemático. Es importante mencionar que la herramienta utilizada para los diagramas circuitales es conocida como EAGLE versión 7.0. Cuyo funcionamiento será explicado más adelante.

Tabla 14. Asignación de Pines RJ45

PINES	NOMBRE DEL PIN	PUERTO DEL MCU	DESCRIPCIÓN
1	TD+	51 /tpout+	Ethernet differentialsignal output.
2	TD-	50 / tpout -	Ethernet differentialsignal output.
3	TAP CENTRAL	POLARIZADO A 3.3V	
4	RD +	47/ tpin+	Ethernet differentialsignal input.
5	RD -	48 / tpin-	Ethernet differentialsignal input.

DIAGRAMA CIRCUITAL DEL RJ45

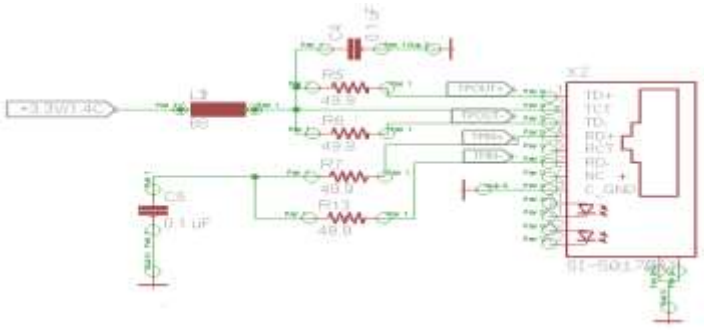

Figura 18. Esquemático RJ45 MIC24010

3.7.2 CONEXIÓN LÓGICA DEL MICROCONTROLADOR CON EL INTEGRADO VS1011

Esta fase debe analizar cuáles son los puertos que necesita el integrado Vs1053b de VSLI SOLUTION escogido para nuestro diseño y el microcontrolador PIC18F67J60 para poder ser interconectados. Para realizar esta conexión debemos dirigirnos a los datasheets de cada uno de los elementos y mirar cómo debe ser conectado. En la siguiente figura podemos apreciar los pines del VS1053b los cuales deben ser asociados con las diferentes interfaces del microcontrolador PIC18F67J60. Se describirán los pines más importantes que deben ser asociados al microcontrolador (Ver Figura. 19).

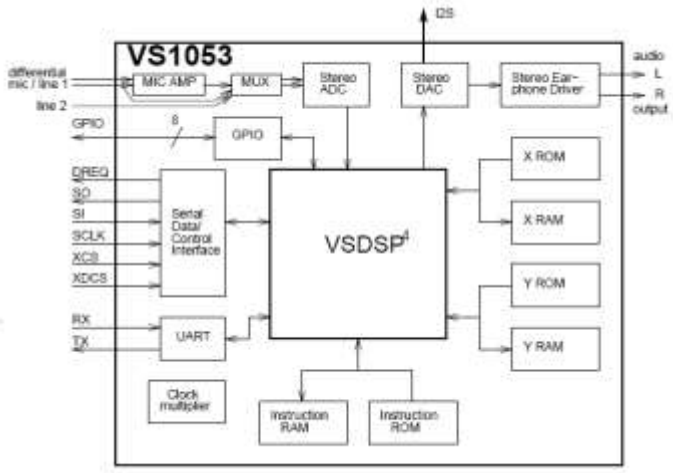

Figura 19. Componentes del vs1053b con sus pines de entrada

Como puede apreciarse en la figura anterior para la integración con el microcontrolador deben tenerse muy en cuenta 6 pines del integrado los cuales en la siguiente tabla serán relacionados con el microcontrolador con las interfaces adecuadas para esto, se indicara la polarización y elementos que deban ir a tierra, además de los elementos circuitales que requiere para su funcionamiento el cual serán especificados en el diagrama circuital. La Tabla 15 muestra la relación de estos pines con el microcontrolador pic18f67j60 (Ver Figura. 20).

Tabla 15. Conexión Lógica Integrado Vs1053b Y El Microcontrolador 18f67j60

Pin VS1053b	No del pin	Descripción	Puerto del pic18f67j60	Descripción
DREQ	8	Data request, input bus	Pin # 3 RB0/INT0/FLT0	RB0: Digital I/O. INT0: External interrupt 0. FLT0: Enhanced PWM Fault input (ECCP modules);
SO	30	Serial output	Pin # 35 RC4/SDI1/SDA1	RC4: Digital I/O SDI1: SPI data in SDA1: I2C data I/O.
SI	29	Serial input	Pin # 36 RC5/SDO1	RC5: Digital I/O. SDO1:SPI data out.
SCLK	28	Clockfor serial bus	Pin # 34 RC3/SCK1/SCL1	RC3: Digital I/O. SCK1: Synchronous serial clock input/output for SPI mode. SCL1: Synchronous serial clock input/output for I2C. mode.
XCS	23	Chip select input	Pin # 5 RB2/INT2	RB2: Digital I/O. INT2: Externalinterrupt 2.
XDCS/ BSYNC	13	Data chip select / byte sync	Pin #4 RB1/INT1	RB1: Digital I/O. INT1: Externalinterrupt 1.

DIAGRAMA CIRCUITAL DEL VS1053b

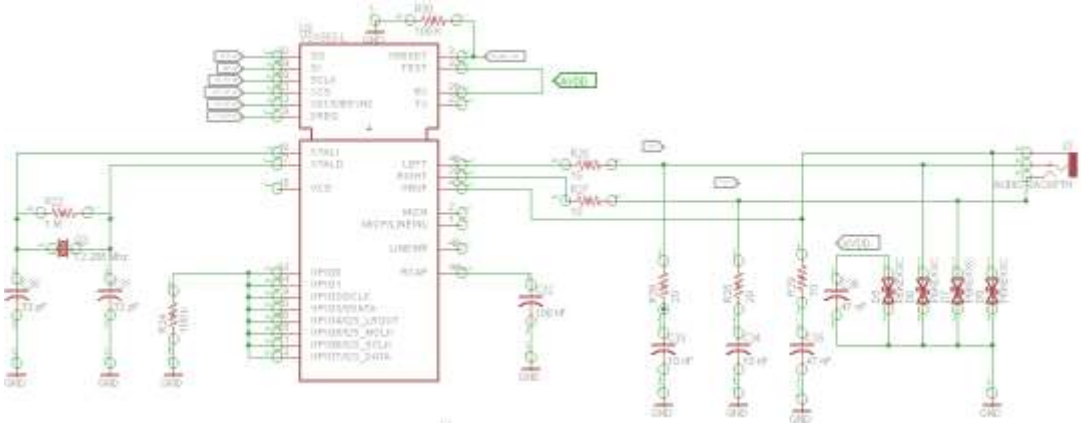

Figura 20. Diagrama Circuital del VS1053b

3.7.3 CONEXIÓN LÓGICA ENTRE EL MICROCONTROLADOR Y LAS MEMORIAS SRAM

Para esta fase debe tenerse en cuenta los puertos que son necesarios para que las memorias puedan comunicarse con el microcontrolador, para esto revisamos los datasheet de cada uno de estos elementos. A continuación son mostrados los pines de las memorias SRAM*256K SPI Bus Low-Power Serial SRAM* (Ver Figura. 21), (Ver Figura. 22).

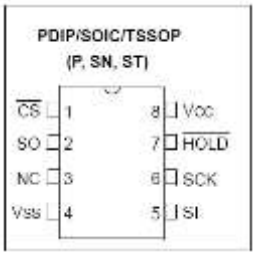

Figura 21. Pines De Las Memorias 256k SPI Bus Low-Power Serial SRAM

La Tabla 16 determina los pines de interconexión entre el microcontrolador y las memorias *256K SPI Bus Low-Power Serial SRAM.* Debido a que tenemos la necesidad de conectar dos memorias, el pin CS es el que cambia para cada memoria el resto de pines tienen un punto común en la conexión. Por lo tanto para este pin será indicado para cada memoria por separado el resto de pines tienen una conexión común. Además es muy importante recordar que las memorias también tiene una interconexión con el integrado VS1053b la cual es mostrada en la siguiente tabla.

Tabla 16. Conexión entre las memorias SRAM con el pic18f67j60 y el integrado VS1053b

Pin SRAM	Descripción	Memoria 1	Memoria 2	No. Pin microcontrolador	Descripción
CS	Chip Select Input	Pin 1	Pin 1	Pin # 62 RE4/AD12/P3B	RE4: Digital I/O. AD12: External memory address/data 12. P3B**(3):** ECCP3 PWM output B.
CS	Chip Select Input	Pin 1	Pin1	Pin # 61 RE5/AD13/P1C	RE5: Digital I/O. AD13: External memory address/data 13. P1C**:** ECCP1 PWM output C
SO	Serial Data Output	Pin 2	Pin 2	Pin # 35 RC4/SDI1/SDA1	RC4: Digital I/O SDI1: SPI data in SDA1: I2C data I/O.
VSS	Ground	Pin 4	Pin 4	NC	
SI	Serial Data Input	Pin 5	Pin 5	Pin # 36 RC5/SDO1	RC5: Digital I/O. SDO1:SPI data out.
SCK	Serial Clock Input	Pin 6	Pin 6	Pin # 34 RC3/SCK1/SCL1	RC3: Digital I/O. SCK1: Synchronous serial clock input/output for SPI

					mode. SCL1: Synchronous serial clock input/output for I2C. mode.
HOLD	Hold Input	VCC	VCC	NC	
VCC	SupplyVoltage	3.3 v	3.3 v	NC	

DIAGRAMA CIRCUITAL DE LAS MEMORIA SRAM

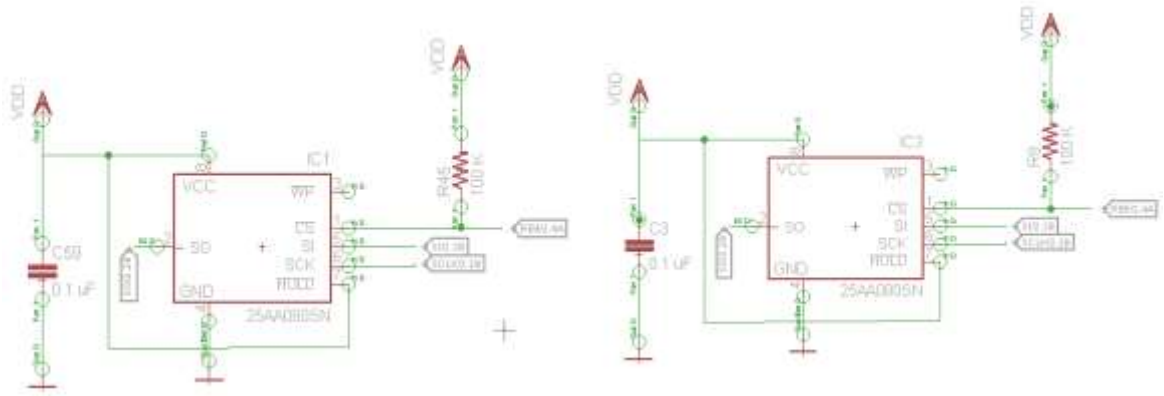

Figura 22. Diagrama circuital Memoria SRAM

3.7.4 Conexión lógica del LCD 16x2

Este elemento es bastante común en soluciones prototipo en hardware, debido a su fácil implementación ya que la conexión es bastante simple, para ello cabe mencionar que cuenta con 16 puntos de conexión distribuidos entre manejo de datos, los mismos datos enviados y la polarización. Cuenta también con un punto de conexión dedicado para el manejo del contraste de la pantalla. Las pantallas de cristal líquido LCD o display LCD para mensajes (Liquid Cristal Display) tienen la capacidad de mostrar cualquier carácter alfanumérico, permitiendo representar la información que genera cualquier equipo electrónico de una forma fácil y económica.

La pantalla consta de una matriz de caracteres (normalmente de 5x7 o 5x8 puntos) distribuidos en una, dos, tres o cuatro líneas de 16 hasta 40 caracteres cada línea. El proceso de visualización es gobernado por un microcontrolador incorporado a la pantalla, siendo el Hitachi 44780 el modelo de controlador más utilizado (Ver Figura 23) [44].

Las características generales de un módulo LCD 16x2 son las siguientes (Tabla 17):

- Consumo muy reducido, del orden de 7.5mW.
- Pantalla de caracteres ASCII.

- Desplazamiento de los caracteres hacia la izquierda o a la derecha.
- Memoria de 40 caracteres por línea de pantalla, visualizándose 16 caracteres por línea y se puede programar 8 caracteres.
- Pueden ser gobernados de 2 formas principales: conexión con bus de 4 bits conexión con bus de 8 bits, en nuestro caso son manejados 4 bits.

Tabla 17. Distribución de pines para la conexión

Pin LCD	Símbolo	Descripción	No.Pin MCU
1	Vss	Pin de alimentación de tierra	
2	Vdd	Pin de alimentación 3.3v	
3	Vo	Patilla de contraste del cristal líquido. Normalmente se conecta a un potenciómetro a través del cual se aplica una tensión variable entre 0 y +5V que permite regular el contraste del cristal líquido.	
4	RS	Selección del registro de control/registro de datos: RS=0 Selección del registro de control RS=1 Selección del registro de datos	Pin27 RA5/AN4
5	R/W	Señal de lectura/escritura R/W=0 El módulo LCD es escrito R/W=1 El módulo LCD es leído	Pin21 RA3/AN3/Vref
6	E	Señal de activación del módulo LCD: E=0 Módulo desconectado E=1 Módulo conectado	Pin28 RA4/TOCK1
7-14	D4-D7	Bus de datos bi-direccional. A través de estas líneas se realiza la transferencia de información entre el módulo LCD y el sistema informático que lo gestiona	Pin 1/2/64/63/14/13/12/11

DIAGRAMA CIRCUITAL DEL LCD

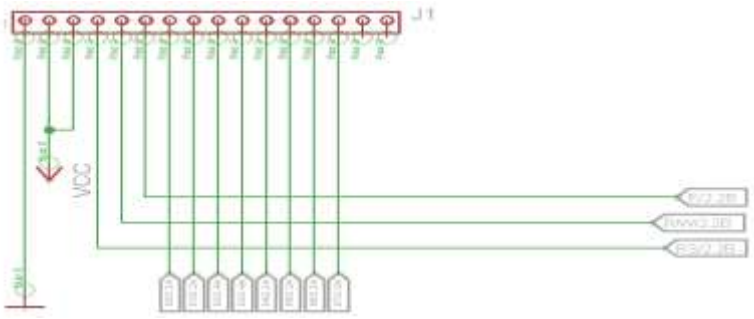

Figura 23. Diagrama Circuital del LCD 16x2

3.7.5 Conexión lógica con PWR

El dispositivo encargado de manejar la etapa de potencia que alimenta al circuito es el elemento desarrollado por *National Semiconductor* (LM1117MPX-3.3TR-ND), básicamente el dispositivo regular el voltaje que se obtenido a través de un adaptador de 110v a 9v y lleva estos 9 v a 3.3v los cuales son necesarios para polarizar el microcontrolador y los demás elementos (Ver Figura 24).

DIAGRAMA CIRCUITAL PWR

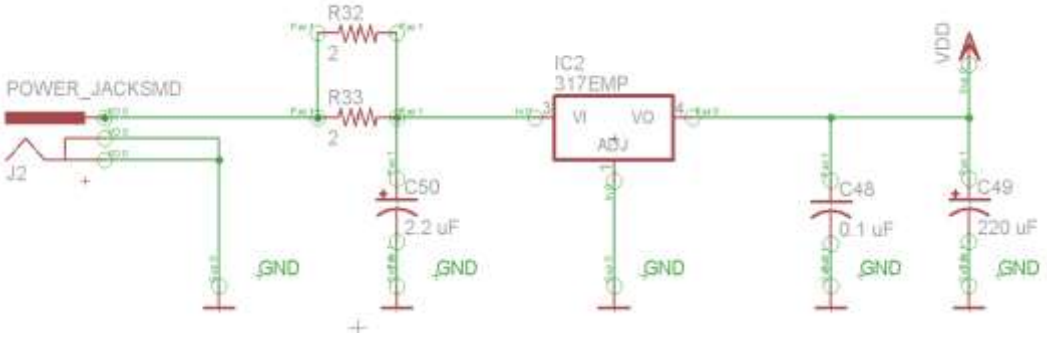

Figura 24. Diagrama Circuital de la etapa de Potencia

CAPITULO IV. IMPLEMENTACION SOFTWARE

4.1 INTRODUCCIÓN

En la segunda etapa del desarrollo del proyecto, una vez planteado el circuito eléctrico del sistema, debe elaborarse el programa que le dará la funcionalidad requerida al sistema definida en las especificaciones del capítulo I, para ello es realizado el diseño software que consta de dos elementos; en primer lugar, un programa principal (*main*) donde están descritas todas las funciones que debe realizar el IRD soportado en un stack TCP-IP encargado de proveer el establecimiento de la conexión a la red, y en segundo lugar, un módulo de selección de contenido que permitirá su configuración, para ello están vinculadas algunas herramientas como librerías y aplicaciones libres que nos permiten incursionar en este tipo de desarrollo teniendo en cuenta que el dispositivo central de este prototipo (IRD) es un microcontrolador PIC18F67J60.

También fue realizado un estudio previo sobre las posibles soluciones existentes que servirían como base para el desarrollo software, principalmente fueron explorados dos opciones, la primera tiene que ver con el uso de Sistemas Operativos Libres como por ejemplo: µC/OS-II, SALVO$_{TM}$, SOS, FreeRTOS$_{TM}$, RTOS del Compilador CCS, esto con el fin de utilizar herramientas de carácter libre; la segunda opción a estudiar tiene que ver con el uso de las herramientas que provee Microchip para el desarrollo software, puesto que este fue el microcontrolador seleccionado en el capítulo anterior.

4.2 SISTEMAS OPERATIVOS EN TIEMPO REAL

Un Sistema Operativo de Tiempo Real (RTOS) puede asemejarse a una librería de funciones que son enlazadas con la aplicación de forma que al iniciar el ordenador, es la aplicación la que toma el control del ordenador e inicializa al sistema operativo para luego pasarle el control, esto permite eliminar las partes del Sistema Operativo de Tiempo Real que no se usan usadas permitiendo un ahorro en memoria, la cual suele estar limitada en los sistemas empotrados. Un sistema de tiempo real es un sistema informático que interacciona con su entorno, sobre el que realiza acciones de controlque son producidas dentro deintervalos de tiempobien definidos [45].

Los Sistemas Operativos Tiempo Real son diseñados para administrar la respuesta a sucesos o eventos, de tal forma que sean respetados ciertos plazos de tiempo de respuesta, los cuales, en el caso de ser críticos para el sistema, requieren atención inmediata. El RTOS es determinista, en consecuencia permite que el usuario prediga los tiempos de respuesta que este proporcionará al sistema informático que administra. Estos eventos, normalmente son producidos en el entorno del sistema llegando como interrupciones al procesador, ya sean por software o hardware [46].

Los RTOS deben atender ráfagas de miles de interrupciones que activan diferentes procesos. Para no perder ningún suceso recibido la técnica de multiproceso es empleada, ella ejecuta los procesos independientemente unos de otros, pero en ciertas aplicaciones debe tenerse estrategias para asignar prioridad a los procesos y ejecutarlos según su importancia, lo cual es logrado con una planificación expropiativa o expulsiva basada en prioridades.

La funcionalidad principal y requerida para un *RTOS* es proveer de un nivel de servicio adecuado a las aplicaciones que requieran una respuesta en un intervalo de tiempo determinado. La característica principal de un *RTOS* es la respuesta ante eventos internos o externos, tales como interrupciones hardware externas, interrupciones software internas o interrupciones de reloj internas, es decir los requerimientos temporales [47].

Teniendo en cuenta la descripción y características principales de los *RTOS* a continuación son exploradas algunas alternativas que podrían ser utilizadas como herramienta para el desarrollo software del prototipo IRD.

4.2.1 RTOS CANDIDATOS

- µC / OS-II

  Es altamente portable y escalable, está escrito en ANSI C y contiene una pequeña porción de código en lenguaje ensamblador para adaptarlo a diferentes arquitecturas de procesadores. µC/OS-II tiene la capacidad de administrar hasta 64 tareas (56 tareas disponibles para el usuario), ha sido portado a más de 100 microprocesadores y microcontroladores. Es simple de usar y sencillo de aplicar, pero muy efectivo en comparación con la relación precio / rendimiento [48].

- **SALVO$_{TM}$**
  Es un sistema operativo en tiempo real de bajo costo poderoso y de alto rendimiento, fue diseñado para que requiera muy poca memoria. Es una sencilla herramienta de software que permite crear rápidamente aplicaciones potentes, fiables y sofisticadas parasistemas integrados. SALVO$_{TM}$ del grupo PumpkincTM, fue diseñado expresamente para sistemas embebidos, cuyas aplicaciones típicas usan entre 1 y 2K bytes de ROM, y entre 50 y 100 bytes de RAM. Cumple con el criterio de multitarea cooperativa, servicio de eventos, bloqueo y suspensión de tareas, y con muchos otros aspectos necesarios descritos anteriormente para el desarrollo del proyecto [49].
  SALVO$_{TM}$ puede implementar en una gran cantidad de dispositivos, como son:

  Familia 8051 y derivados.
  ARClitemicroRISC.
  ARM® ARM7TDMI®

Atmel® AVR® y MegaAVR™
Motorola M68HC11
TI's MSP430 Microcontrolador de baja potencia
Microchip PIC12|14000|16|17|18 PICmicro® MCUs

Sin embargo, la versión demo de SALVO$_{TM}$ que puede ser adquirida libremente permite crear únicamente 3 tareas como máximo, lo que impediría en gran parte un buen aprovechamiento del mismo [49]. Obligando a la compra de una licencia, lo cual no está dentro de las posibilidades porque es deseable trabajar con herramientas libres.

- **SOS**

  Es un sistema operativo de tiempo real de alto rendimiento que ha sido optimizado para un mínimo de consumo de memoria. Es muy pequeño y simple. SOS implementa un sistema de mensajería que permite múltiples enlaces entre la base del sistema operativo y las aplicaciones, las cuales pasan a ser módulos que pueden ser cargadas o descargadas en tiempo de ejecución sin interrumpir la base del sistema operativo [50]. El principal objetivo de SOS es la "*reconfigurabilidad".*

  Además de las técnicas tradicionales usadas en el diseño de sistemas embebidos, las características del kernel de SOS son: Módulos cargados dinámicamente, programación flexible de prioridades, simple subsistema de memoria dinámica.

- **FreeRTOS**

  FreeRTOS$_{TM}$ ofrece bajo consumo memoria RAM puesto que ha sido desarrollado específicamente para aplicaciones con sistemas embebidos (microcontroladores). Presenta además tres modelos posibles de asignación de memoria para la implementación de las aplicaciones. Utiliza un tipo de planificación expropiativa que implementa colas por cada nivel de prioridad, recorridas cíclica y equitativamente según el mecanismo FIFO; también ofrece un sistema de sincronización por medio de semáforos binarios. FreeRTOS$_{TM}$ es de libre distribución. También puede ser implementado en una gran variedad de dispositivos [51].

  - LPC2106, LPC2124 and LPC2129 (ARM7). Incluye código para el manejo de I2C. Renesas H8S2329 (Hitachi H8/S) con un demo de EDK2329.
  - Atmel AT91SAM7
  - AT91FR40008 con demo de ATEB40X.
  - MSP430 con manejador de LCD.
  - HCS12 (MC9S12C32).
  - Cygnal 8051 / 8052.
  - Microchip PICMicro (PIC18).
  - Atmel AVR (MegaAVR) con código de demostración STK500.

- RDC8822 Microcontroller (AMD clon embebido del 186) con demo para Flashlite 186 SBC.

- **RTOS del CCS**
  El Compilador C para PICmicro de CCS, Inc tiene integrado un sistema operativo multitarea cooperativo, permite que las tareas programadas sean ejecutadas en intervalos de tiempo especificados por el programador. Debido a que el RTOS no utiliza interrupciones, el programador debe tener cuidado de hacer uso de la rtos_yield () en cada tarea, para así evitar que una tarea pueda quedarse ejecutando para siempre.

Es evidente que casi todos los sistemas operativos seleccionados cumplen con los requisitos para el desarrollo de este proyecto, sin embargo, hay que recalcar que algunas versión disponibles son demos que no permiten utilizar todas los beneficios que ofrecen los *RTOS* es el caso de *SALVO*$_{TM,}$ por otra parte algunas herramientas no cuentan con una compatibilidad con el hardware seleccionado o de alguna manera resultaba muy complejo su adaptación tal es el caso de *freeRTOS*y los bancos para la asignación de memoria.

Es difícil llegar a una comparación precisa entre los Sistemas Operativos en Tiempo Real mencionados anteriormente y el Stack de protocolos provisto por Microchip ya que analizándolos independientemente cada uno tiene sus ventajas y desventajas, pero es fácil llegar a una selección considerando que las herramientas para el desarrollo software de Microchip están bien definidas en cuanto a las aplicaciones que pueden implementarse, son libres a comparación de algunos de los *RTOS* estudiados y principalmente porque la sencillez de su implementación nos dan una base clara para la consecución del objetivo principal de este proyecto el cual es la captura y reproducción de Streaming de Audio. Adicional a esto, existe un sin número de aplicaciones en las que resalta el uso de estas herramientas, permitiendo agilizar el proceso de diseño, implementación, costos, etc. [52],[53],[54].

4.3 STACK TCP/ IP

Para la realización del IRD es indispensable definir los elementos que le proveerán al sistema el correcto funcionamiento manipulando los componentes hardware y realizando las correspondientes tareas que sean necesarias para poder capturar y reproducir audio. El primer reto a solucionar a nivel software corresponde a la capacidad del IRD para conectarse a Internet porque el funcionamiento de este prototipo depende en gran medida de establecer una correcta conexión a la red.

El Modelo TCP/IP es una descripción de los protocolos que permiten la comunicación entre computadores sobre una red. El Stack TCP/IP son una serie de funcionalidades con

los cuales son implementados los protocolos del modelo TCP/IP y así materializar el modelo de comunicación en redes de trabajo, Microchip por parte proporciona la versión disponible PIC's del sistema BSD *(Berkley Software Distribution*) [55], el cual es de código abierto, y sirvió como base y punto de partida para el desarrollo de este proyecto permitiendo así, enfocarse en la optimización del diseño Hardware y en la aplicación final. A continuación es presentado el Stack TCP/IP que provee Microchip el cual es una solución software libre y de alto grado de desarrollo que permitió cumplir con este primer objetivo el cual es determinante para el correcto funcionamiento del sistema.

4.3.1 MICROCHIP TCP/IP STACK

Microchip ofrece una pila de software TCP/IP optimizado para los PIC18, PIC24 y familia de microcontroladores dsPIC [56]. La pila es un conjunto de programas que proveen de los servicios TCP/IP a las aplicaciones. Basada en el modelo de referencia TCP/IP, la pila es dividida en múltiples capas, la misión de cada capa es proveer servicios a las capas superiores haciendo transparente el modo en que esos servicios son llevados a cabo. De esta manera cada capa debe ocuparse exclusivamente de su nivel inmediatamente inferior, a quien solicita servicios, y del nivel inmediatamente superior, a quien devuelve resultados. Por especificaciones muchas de las capas están "vivas", en el sentido de que no sólo actúan cuando es solicitado un servicio, sino también cuando los eventos como el tiempo de espera o la llegada de un nuevo paquete son producido. La pila sigue un diseño modular y está escrita en el lenguaje de programación C. En conjunción con el controlador Ethernet permite reducir los tiempos y costes de desarrollo de aplicaciones a través de una red de área local o Internet (Ver Figura 25).

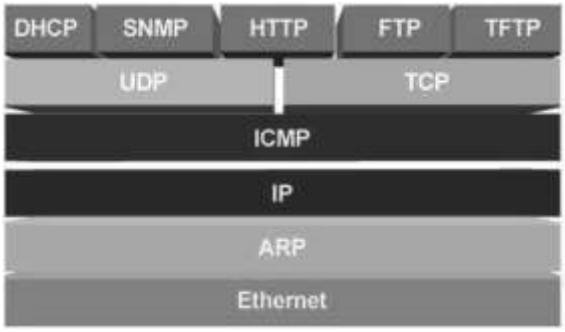

Figura 25. Pila TCP/IP

Dentro de las características de la versión de Microchip para PIC´s del Stack BSD están:

- Soporte para aplicaciones cliente/servidor.
- Optimización del Stack para su uso en sistemas embebidos.
- Comunicación bidireccional completa.

- Soporte para paquetes de Stream y UDP.

También es importante mencionar las limitaciones existentes con respecto al uso del Stack de Microchip, es claro que es diseñado exclusivamente para el uso en PIC´s lo cual trae limitaciones propias para el uso de estos dispositivos. Dentro de estas, las limitaciones más importantes son:

- El Stack de Microchip no tiene la implementación completa del Stack BSD original.
- El comportamiento de algunas funciones puede diferir ligeramente de las funciones originales.
- Todas las funciones son de naturaleza no-bloqueante.

Los requerimientos básicos para el uso del Stack de Microchip están referidos principalmente a los recursos de memoria por parte del microcontrolador es necesario que el PIC tenga un mínimo de 25 KB en FLASH y 3KB en RAM, como puede apreciarse en la tabla resumen de las características del microcontrolador seleccionado vemos que los requerimientos de memoria son cumplidas.

Es posible segmentar el Stack TCP/IP de Microchip en los siguientes módulos para su mayor entendimiento [57]:

DRIVER'S PARA LOS CONTROLADORES ETHERNET

Este módulo implementa la interfaz del Hardware a bajo nivel para el control del chip Ethernet. Todas las especificaciones y detalles están ocultas a las capas superiores. La idea de este módulo es recuperar mensajes desde el controlador Ethernet y empaquetarlos para ser usados en capas superiores de la implementación.

ADMINISTADOR DE STACK

Este módulo es el encargado de ejecutar las máquinas de estado y los distintos procesos dentro del stack. El módulo está encargado además de recuperar los paquetes de entrada y enviarlos a controladores de alto nivel en un formato apropiado. El uso del administrador permite que el stack sea ejecutado de manera independiente a la aplicación montada sobre él.

CONTROLADOR DEL PROTOCOLO DE RESOLUCIÓN DE DIRECCIONES (ARP)

Aquí son creados y procesados los paquetes referidos a ARP. Las aplicaciones o el usuario del stack no tienen que llamar a funciones de la capa ARP ya que el administrador de IP inicia el proceso de resolución de direcciones de manera automática.

ADMINISTRADOR DE IP

Este módulo controla la capa IP. Recibe los paquetes entrantes y enviados desde el administrador de stack y son verificados. Todos los paquetes validos son enviados a los

controladores de los protocolos de capas superiores. Este stack soporta tres niveles distintos de protocolos de IP:

- TCP
- UDP
- ICMP

El administrador de IP también genera solicitudes ARP para los paquetes de salida. Si el MAC-ID de la dirección de destino no está disponible, el administrador de IP genera una solicitud ARP y deja este archivo pendiente en cola.

Para facilitar el proceso de configuración, la pila usa la instrucción del compilador 'C', "defines" para habilitar o deshabilitar un parámetro en particular, el usuario tiene la posibilidad de manipular uno o más de estos "defines". La mayoría de éstos son definidos en el archivo de encabezado "*StackTsk.h*". Algunos de los "*defines"* que están definidos en otros archivos son mostrados con el nombre correspondiente del archivo. Una vez que estos archivos hayan sido modificados, es necesario recompilar el proyecto de aplicación para incluir los cambios realizados.

4.4 APLICACIÓN (MAIN)

En la actualidad encontramos una gran cantidad de aplicaciones para sistemas embebidos basados en microcontroladores proporcionadas por los proveedores como parte de las herramientas de soporte para sus implementaciones hardware, es así como Microchip promueve la utilización de sus Kits de desarrollo ofreciendo como veremos en este capítulo el Stack TCP/IP el cual es explicado con más detenimiento en la sección siguiente. Hay que resaltar que mediante la utilización de este compilado de librerías que proporciona Microchip de manera gratuita es posible hacer uso de algunos Firmwares o códigos pre-implementados con algunas funcionalidades específicas, por ejemplo un servidor web embebido, aplicaciones TCP-IP en general y, dentro de estas, una aplicación pre-diseñada para la captura de flujos de datos de audio.

Teniendo en cuenta nuestro objetivo general, el cual se referido a la captura y reproducción de Streaming de Audio, nos centraremos en la utilización de algunos de estos Firmwares proporcionados de manera libre y gratuita por parte de Microchip a razón de ser códigos bastante optimizados lo cual es una ventaja frente a la limitante de recursos en memoria, pero principalmente porque como segundo objetivo del proyecto es planteado el uso de elementos de bajo costo así mismo de herramientas para el desarrollo software como lo son este tipo de aplicaciones libres. Por lo anterior nuestro proyecto tendría un componente software basado en estas herramientas y en adelante nos centraremos en la comprensión y adecuación de estas librerías a nuestro diseño

hardware así como a la adaptación de una función de selección de contenidos como una función extra ya que este tipo de herramientas permiten realizar estas modificaciones

A continuación están descritas las características más relevantes de la aplicación *MainDemo*, (Ver Figura 26), el código correspondiente al programa principal así como las librerías que en él están incluidas, son ofrecidas como parte del Anexo 4.

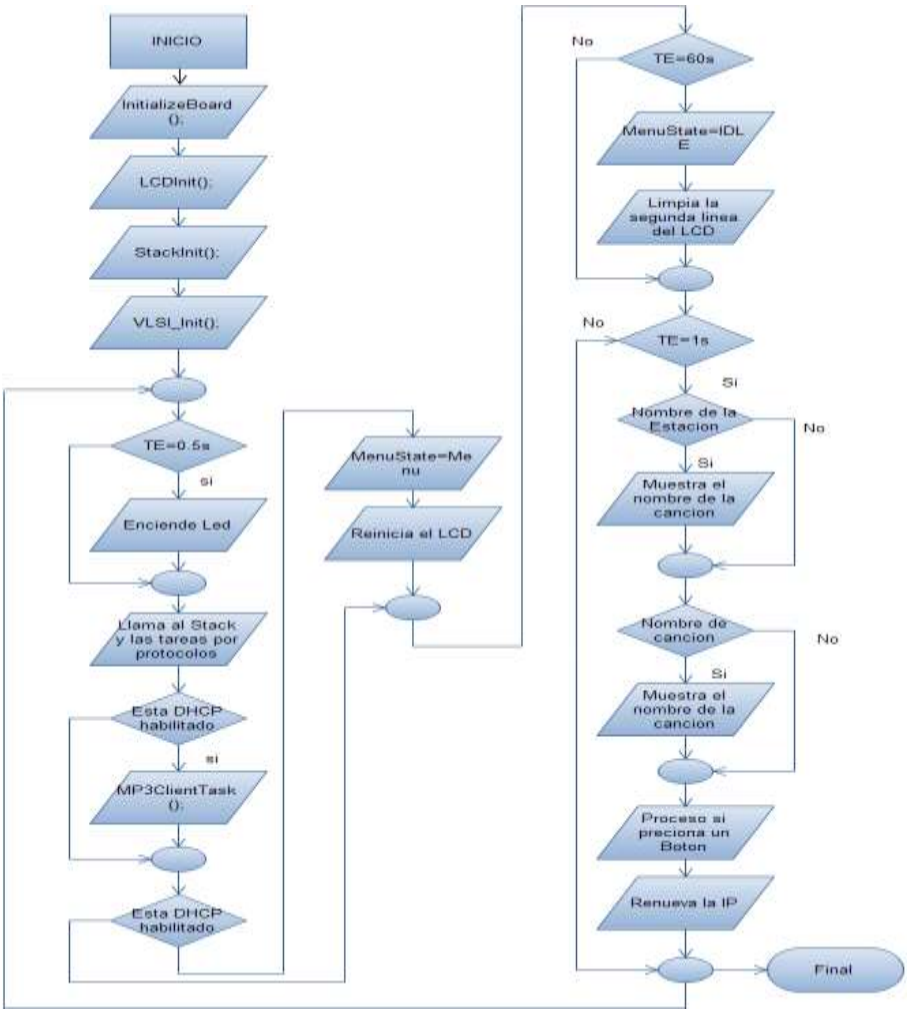

Figura 26. Diagrama de Flujo Diseño software

La primera tarea realizada dentro de la aplicación principal es inicializar todos los componentes Hardware del sistema: Microcontrolador, Periféricos (Decodificador, LCD, SRAM).

Esta tarea consiste básicamente en la definición de los parámetros a utilizar, así como la asignación y configuración de los pines de cada componente, este proceso es bastante complejo por la cantidad de factores a tener en cuenta, afortunadamente Microchip provee

dentro de su Stack de librerías una encargada de realizar todas estas asignaciones basándose en perfiles para los MCU's PIC18, PIC24F, PIC24H, dsPIC30F, dsPIC33F, PIC32. Cabe aclarar que esta librería llamada *HardwareProfile.h* es de uso libre por lo cual es posible realizar modificaciones sobre el código fuente, esta característica resulta bastante conveniente ya que en sistemas como el IRD podemos incorporar a uno de los perfiles mencionados anteriormente, la configuración hardware para los componentes extras como el decodificador, las memorias SRAM y el LCD, en general el uso de esta librería resulta bastante práctico por su portabilidad y modularidad.

Seguidamente de la inicialización del sistema es realizada la petición para el proceso de conexión a la Red, de la misma manera que cuenta con un perfil para los componentes Hardware, en el Stack TCP/IP provisto por Microchip posee un perfil para los protocolos que intervienen en la conexión mediante la librería *TCPIPConfig.h,* en ella son definidos todos los protocolos incluidos en el Stack y a la vez es habilitado o no su uso dependiendo de las necesidades y características de la aplicación, como nuestro objetivo es capturar Streaming de Audio entonces es en esta librería donde es definido el uso de los protocolos necesarios para ello por ejemplo a nivel de transporte UDP, en aplicación HTTP como fue definido en el capítulo 2 y entre otros protocolos fue habilitado el uso de DHCP, ICMP, DNS. El Stack junto con sus características específicas para el proyecto puede verse más a fondo en la sección siguiente.

También fue configurado bajo esta librería un HostName el cual permitirá identificar en la red mediante *MY_DEFAULT_HOST_NAME* al prototipo, como hemos venido haciendo referencia durante el proyecto el dispositivo tiene como HostName las siglas (IRD, Internet Radio Device) por ejemplo (http://ird), también encontramos en esta librería la asignación de la dirección MAC a través de *MY_DEFAULT_MAC_BYTE5* para la implementación

El Stack posee un conjunto de funciones propias las cuales son descritas a continuación:

- *StackTask*: maneja el paquete Rx interno para pre-procesarlo previamente a ser despachado a las capas de aplicación superiores. Esta función chequea nuevos paquetes entrantes y guía a ellos hacia los componentes de la pila apropiada.

- *HTTPServer*: servidor de páginas dinámicas para los navegadores web tales como el Explorer de Microsoft, Mozilla Firefox, etc.

- *DiscoveryTask*: examina los paquetes de Broadcast para saber si se trata de una petición de descubrimiento, luego comienza el envío de nuestra dirección MAC en formato legible (convertimos la dirección MAC de bytes a hexadecimal), al último nodo de donde recibimos la petición.

- *DHCPServerTask*: proporciona una dirección IP automática, máscara de subred, dirección de puerta de enlace, la dirección del servidor DNS, y otros parámetros de configuración DHCP utilizados en redes.

- *ETH97J60Task:* provee acceso a toda la familia de microcontroladores PIC 18F97J60, basados en el estándar *IEEE 802.3*

- *IPTask*: provee el transporte a los mensajes, *TCP*, *UDP* y *ICMP*, basado en la el RFC 791[58]

Debido a que el programa necesita una serie de funciones que son llamadas de manera automática unas a otras, para cumplir con una tarea específica, hemos utilizado cabeceras o archivos '.h'. Las cabeceras contienen única y exclusivamente los prototipos de las funciones y en algunos casos, contienen las estructuras de datos que ya hemos mencionado.

Las principales cabeceras que utilizamos son: *ARP.h, DHCP.h, DNS.h, ETH97J60.h, HTTP2.h, ICMP.h, IP.h, MAC.h, MPFS2.h, NBNS.hREBOOT.h, SPIRAM.h, STACKTSK.h, TCP.h, TCPIP.h, TICK.h, UDP.h.*

Con el código fue realizada la configuración de los registros del PIC18F67J60 y al Stack TCP/IP, permitiendo hacer las inicializaciones respectivas en el PIC. Con esto asegura un buen desempeño del sistema, tanto en el proceso interno del microcontrolador, como en las diferentes interfaces que son manejadas. Las funciones que permiten llamar y utilizar las distintas funciones, y estructuras del Stack, ayudan a que el dispositivo pueda comunicarse con la red.

4.5 MÓDULO DE SELECCIÓN DE CONTENIDOS

El módulo para la selección de contenidos consiste en permitirle al usuario del dispositivo reproducir el Streaming de Audio generado por la Radio Universidad del Cauca o si desea acceder a otro tipo de información debe autenticarse con el sistema para poder ingresar a ese contenido. Esto es realizado con el fin de que el contenido a reproducir sea el que personas con autorización decidan que debe reproducirse.

4.5.1 UNICAUCACLIENTETASK(VOID)

Nuestro aporte en cuanto a la función para la selección de contenidos es evidente mediante la implementación de la función *UnicaucaClienteTask(void)*, definida en el archivo de cabecera *MP3Client.h* en esta rutina describe una máquina de estados capaz de comunicarse con el servidor de ShoutCast, para el caso específico de la función la

petición que realiza al servidor tiene como respuesta la ubicación y posterior captura y reproducción de la emisora Radio Universidad del Cauca, todo esto es logrado mediante la definición en el archivo cabecera MP3Cliente.h de una ruta al servidor de Streaming la cual hace uso de la palabra reservada *search* en la siguiente petición:
http://yp.shoutcast.com/sbin/newxml.phtml?search="Unicauca"

Como fue mencionado el comportamiento de la rutina encargada de comunicar al prototipo (IRD) con el servidor de Streaming de Audio a través de mensajes HTTP es una máquina de estado que puede modelarse así (Ver Figura. 27).

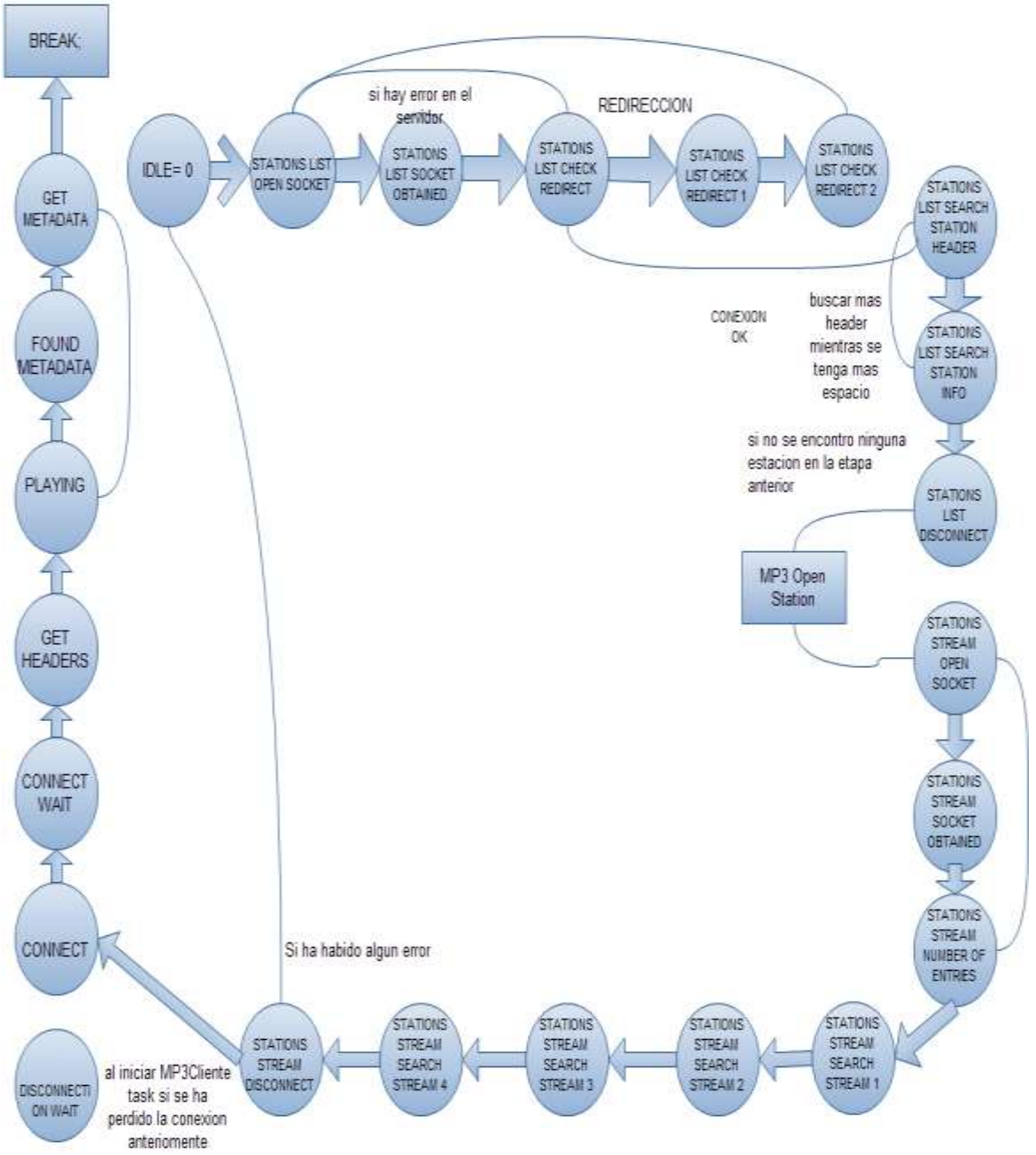

Figura 27. Máquina de estados UnicaucaCleinteTask()

Es importante recalcar que la gran mayoría de los estados en la figura anterior incorporan un *timeout*en el cual es definido un tiempo de espera mínimo para verificar la llegada de datos por ejemplo, a través de un socket; esto con el fin de no acaparar el tiempo de procesamiento en una sola tarea como fue definido en el *main()* bajo la filosofía de *Multitasking Cooperativo*. Los casos más relevantes a describir están a continuación y son incluidos en la función UnicaucaTaskCliente():

SM_STATIONS_LIST_OPEN_SOCKET

Este estado es encargado de crear el socket por el cual será pedido al servidor de ShoutCast la lista que incluye la Radio Universidad del Cauca, como fue mencionado al definir una ruta específica la petición para la creación del Socket.

```
case SM_STATIONS_LIST_OPEN_SOCKET:
    // Inicializa la lista de estaciones
    stationCount = 0;
    for (i = 0; i<STATIONUNI_COUNT; i++)
    {
        stations[i].ID[0] = 0;
    }

    MySocket = TCPOpen((DWORD)(PTR_BASE)(BYTE *)shoutcastHost, TCP_OPEN_RAM_HOST, SHOUTCAST_PORT, TCP_PURPOSE_MP3_CLIENT);
    if(MySocket == INVALID_SOCKET)
        return;

    Timer = TickGet();
    genreRetryCount = 0;
    smMP3State++;
    break;
```

- Inicializa la lista de estaciones

SM_STATIONS_LIST_SOCKET_OBTAINED

Revisa el estado del Socket, su conexión y que no pase demasiado tiempo para el establecimiento de la misma, por último verifica la disponibilidad para escribirle realizando la siguiente petición:

```
case SM_STATIONS_LIST_SOCKET_OBTAINED:
    // Espera a que el servidor remoto acepte nuestra peticion
    if(!TCPIsConnected(MySocket))
    {
        // Aplica un Time out si pasa demasiado tiempo en este estado
        if(TickGet() - Timer > 5*TICK_SECOND)
        {
            iRadioStatus.bConnect1Error = 1;
            iRadioStatus.bShowError = 1;
            stationCount = 0;
            smMP3State = SM_STATIONS_STREAM_DISCONNECT;
        }
        break;
    }

    // Se asegura de que pueda escribir
    if(TCPIsPutReady(MySocket) < 125u)
        break;

    Timer = TickGet();

    // Make the HTTP request
    TCPPutROMString( MySocket, (ROM BYTE*)"GET " );
    TCPPutROMString( MySocket, (ROM BYTE*)SHOUTCAST_PATH_SEARCH );    // Path
    TCPPutROMString( MySocket, (ROM BYTE*)(genres[genre].ShoutcastName) );
    TCPPutROMString( MySocket, (ROM BYTE*)" HTTP/1.0\r\nHost: " );
    TCPPutString    ( MySocket, (BYTE *)shoutcastHost );                    // Host
    TCPPutROMString( MySocket, (ROM BYTE*)"\r\nConnection: close\r\n\r\n" );
    TCPFlush(MySocket);

    smMP3State++;
    break;
```

- Espera a que el servidor remoto acepte nuestra petición

- Aplica un tiempo de espera, para que no pase demasiado tiempo en este estado

SM_STATIONS_STREAM_OPEN_SOCKET

Solicitaun Socket a ShoutCast

```
case SM_STATIONS_STREAM_OPEN_SOCKET:
    //Abre el Socket para que podamos solicitar la informacion de la estacion
    MySocket = TCPOpen((DWORD)(PTR_BASE)(BYTE *)shoutcastHost, TCP_OPEN_RAM_HOST, SHOUTCAST_PORT, TCP_PURPOSE_MP3_CLIENT);
    if(MySocket == INVALID_SOCKET)
        return;

    Timer = TickGet();
    genreRetryCount = 0;
    smMP3State++;
    break;
```

- Abre el socket para solicitar la información de la estación

SM_STATIONS_STREAM_SOCKET_OBTAINED

Verifica que el Socket esté conectado y que pueda escribir más de 125 caracteres.

```
case SM_STATIONS_STREAM_SOCKET_OBTAINED:
    //Espera a que el servidor remoto acepta nuestra peticion
    if(!TCPIsConnected(MySocket))
    {
        // Aplica un Time out si pasa demasiado tiempo en este case
        if(TickGet() - Timer > 5*TICK_SECOND)
        {
            iRadioStatus.bConnect2Error = 1;
            iRadioStatus.bShowError = 1;
            smMP3State = SM_STATIONS_STREAM_DISCONNECT;
        }
        break;
    }

    // Se asegura de que puede escribir
    if(TCPIsPutReady(MySocket) < 125u)
        break;

    Timer = TickGet();

    //Hace la peticion HTTP
    TCPPutROMString( MySocket, (ROM BYTE*)"GET " );
    TCPPutROMString( MySocket, (ROM BYTE*)SHOUTCAST_PATH_STREAM );  // Path
    TCPPutString   ( MySocket, stations[station].ID );
    TCPPutROMString( MySocket, (ROM BYTE*)" HTTP/1.0\r\nHost: " );
    TCPPutString   ( MySocket, (BYTE *)shoutcastHost );            // Host
    TCPPutROMString( MySocket, (ROM BYTE*)"\r\nConnection: close\r\n\r\n" );
    TCPFlush(MySocket);

    smMP3State++;
    break;
```

- Espera a que el servidor remoto acepte la petición
- Aplica un tiempo de espera
- Se asegura que pueda escribir

SM_CONNECT

Pide un Socket directo a la radio seleccionada en este caso a Radio Unicauca

```
// Inicia la conexion TCP
case SM_CONNECT:
    //if(!DHCPIsBound(0))
        //break;


    #define ACCEPT_STRING           "\r\nAccept: */*\r\nIcy-MetaData:1\r\nConnection: close\r\n\r\n"

    MySocket = TCPOpen((DWORD)(PTR_BASE)&stationHostName[0], TCP_OPEN_RAM_HOST, stationPort, TCP_PURPOSE_MP3_CLIENT);
    if(MySocket == INVALID_SOCKET)
        return;


    oledFillLine(0x00,0);
    oledFillLine(0x00,1);
    oledFillLine(0x00,2);
    oledFillLine(0x00,3);

    strcpypgm2ram( (char *)stationMessage, (ROM void *)"GET /" );
    strcat( (char *)stationMessage, (char *)stationPath );
    strcatpgm2ram( (char *)stationMessage, " HTTP/1.0\r\nHost: " );
    strcat( (char *)stationMessage, (char *)stationHostName );
    strcatpgm2ram( (char *)stationMessage, ACCEPT_STRING );

    rstrMessage = stationMessage;
    smMP3State = SM_CONNECT_WAIT;
    break;
```

- Inicializa la conexión TCP

SM_GET_HEADERS

Dentro del estado es definida una nueva máquina de estados que en realidad es la que lleva el proceso de interpretación de los datos obtenidos, su funcionamiento es descrito a continuación:

- **SM_FIND_HEADERS:** este estado realiza las siguientes tareas, busca "*nicy-name*" y "*nicy-metaint*" y mediante *smHeaderParser* indica que fue encontrada la información correspondiente al nombre de la emisora y los metadatos, posteriormente la variable que desplaza a través de la máquina de estados *smMP3State* ubica en el estado *SM_PLAYING*, inicializa el puntero *dwNextMetaData*, deshace de todos las cabeceras innecesarias, dejando al final al puntero justo donde empieza el mp3 o carga útil y pasa a *SM_PLAYING*.

- **SM_FOUND_NAME:** este estado lee lo que esta después de "*nicy-name*", normalmente corresponde al título de la emisora que está dentro de la cadena strTitle y es lo que es visualizado en el LCD, posteriormente vuelve al estado *SM_FIND_HEADERS*.

- **SM_FOUND_METAINT**: aquí lee el tagicy-metaint este le informa al cliente cuantos bytes del Stream leer antes de esperar el inicio del metadata (que es donde el título de las canciones y otra información están contenidas). La cuenta comienza al principio del Stream (no en la cabecera), luego vuelve al estado *SM_FIND_HEADERS*.

- **SM_PLAYING:** Mediante un ciclo *while* infinito, calcula la máxima cantidad de datos que pueden escribir en el buffer, contenido en la memoria SRAM2, calcula la máxima cantidad de datos que puede escribir en el buffer, (SRAM2) tomando en cuenta: los bytes que están esperando en la cola FIFO TCP y de no almacenar una mayor

cantidad de bytes, este espacio máximo debe ser guardado en un arreglo llamado w. Luego pide los datos al buffer TCP mediante la petición TCPGetArray(MySocket, vBuffer, w) y escribe los datos en la SRAM2 mediante la función SPIRAM2PutArray(wHeadPtrShadow, vBuffer, w), por ultimo cuando cumple esto es decir si llegan los metadatos, pasa al estado SM_FOUND_METADATA.

- **SM_FOUND_METADATA:** este estado lee 1 byte de la metadata en *dwNextMetaData* y lo multiplica por 16 (esto corresponde al protocolo ShoutCast, donde define que el primer byte de la metadata corresponde al largo de esta dividido por 4), seguidamente pasa al estado *SM_GET_METADATA*. En el protocolo *ShoutCast* es definido a *lenght byte* como la longitud del *tagmetadata* dividido por 16, por ejemplo si ese valor es de 4 bytes, significa que el tamaño del metedata es de 4x16=64 bytes. Si los datos no utilizan todos los 64 bytes, el espacio sin utilizar es completado con saltos de línea "\n". El tamaño máximo de metadatos es 4080 bytes, lo que significa que el *lenght byte* puede llegar a ser de hasta 255 bytes.

- **SM_GET_METADATA: f**inalmente este estado lee del socket los bytes de metadata que alcance en *strHeader (BYTE strHeader[65];)*, el resto que no sean metadatos los elimina. Posteriormente actualiza el puntero *dwNextMetaData*. Busca "*StreamTitle*='" y lee lo siguiente a strTitle, hasta que llega a un '\', luego de lo cual normalmente llamaría a *NewStreamTitleProc(strTitle)* para actualizar el LCD. Finalmente cambia al estado *SM_PLAYING*.

4.5.2 SELECCIÓN DEL MODO DE FUNCIONAMIENTO

Siguiendo con la descripción del desarrollo Software que fue implementado en el prototipo (IRD) y habiendo explicado el comportamiento que sigue la máquina de estados definida dentro de la función *UnicaucaTaskCliente()*, se continuo con la implementación desde el m*ain()* o programa principal quien es encargado de invocar todas aquella funciones necesarias para el funcionamiento del dispositivo. En la sección *4.2 Aplicación (Main)* fue mencionado el procedimiento que sigue el dispositivo en el momento de inicialización de todos los elementos Hardware y Software, como parte de nuestro aporte y buscando dar cumplimiento a nuestro objetivo de reproducción de Streaming de Audio concretamente el generado desde Radio Universidad del Cauca fueron incluidas dos rutinas definitivas en el momento de selección del modo de funcionamiento. La primera consiste en crear un menú inicial donde el usuario tiene la posibilidad de seleccionar una ruta específica para acceder ya sea a Radio Universidad del Cauca o a la que hemos denominado otras emisoras porque al seleccionarla cabe la posibilidad de que el usuario navegue entre varios géneros pre-programados y a su vez seleccione entre un compendio

de 12 emisoras, cabe aclarar que la reproducción de algunas emisoras en algunos casos no podría darse por factores ajenos al funcionamiento del dispositivo factores que serán analizados con detenimiento en el capítulo final donde son mostradas las pruebas y conclusiones de este trabajo.

Para llevar a cabo esta selección se hace uso de banderas de estados básicamente son manejadas dos tipos de banderas: (*flag_estado* y *flag_emisora*), por defecto ambas son inicializadas en 0 y así son mantenidas hasta el momento en que es ejecutado el ciclo *while(1)* implementado dentro del *Main()* y el cual abarca todas las tareas incluidas en el *Multitasking Cooperativo*, como es sabido esto es considerado un ciclo infinito en el cual son ejecutados una u otra tarea dependiendo de los estados en que este el proceso además de las interrupciones programadas para ejecutarse por medio de los botones de manejo del sistema.

Los estados por los cuales el proceso dentro del main es desplazado son:

- MENU
- AUDIO
- GENERO
- ESTACION
- VOLUMEN
- BAJOS
- ESTADO

Haciendo uso de la variable de tipo *BYTE menuState* lo cuales iniciado en ESTADOpodemos ubicar al usuario en el menú inicial ya que la bandera *flag_estado* es inicializada en 0, en adelante dependiendo del botón presionado la bandera *flag_emisora* cambia de valor de 0 a 1 si selecciona OtrasEmisoras o 2 si selecciona Unicauca. El funcionamiento y la manera en que es operado el dispositivo es explicado en detalle en el Anexo 6, correspondiente al manual de usuario.

Habiendo seleccionado una de las dos opciones es expuesta la segunda rutina encargada de seleccionar el modo de funcionamiento, en ella mediante el manejo de sentencias de decisión *if*es consultado el estado de la bandera *flag_emisora:*

```
//BOTON DE LA DERECHA
if (flag_emisora == 1)
    {
        MP3ClientTask();
    }
else
    {
    //BOTON DE LA IZQUIERDA
    if (flag_emisora == 2)
        {
        UnicaucaClientTask();
        }
    }
```

Revisa el estado de la bandera

Lo explicado anteriormente en cuanto al modo de selección puede apreciarse dentro del diagrama de flujo mostrado en la figura 28.

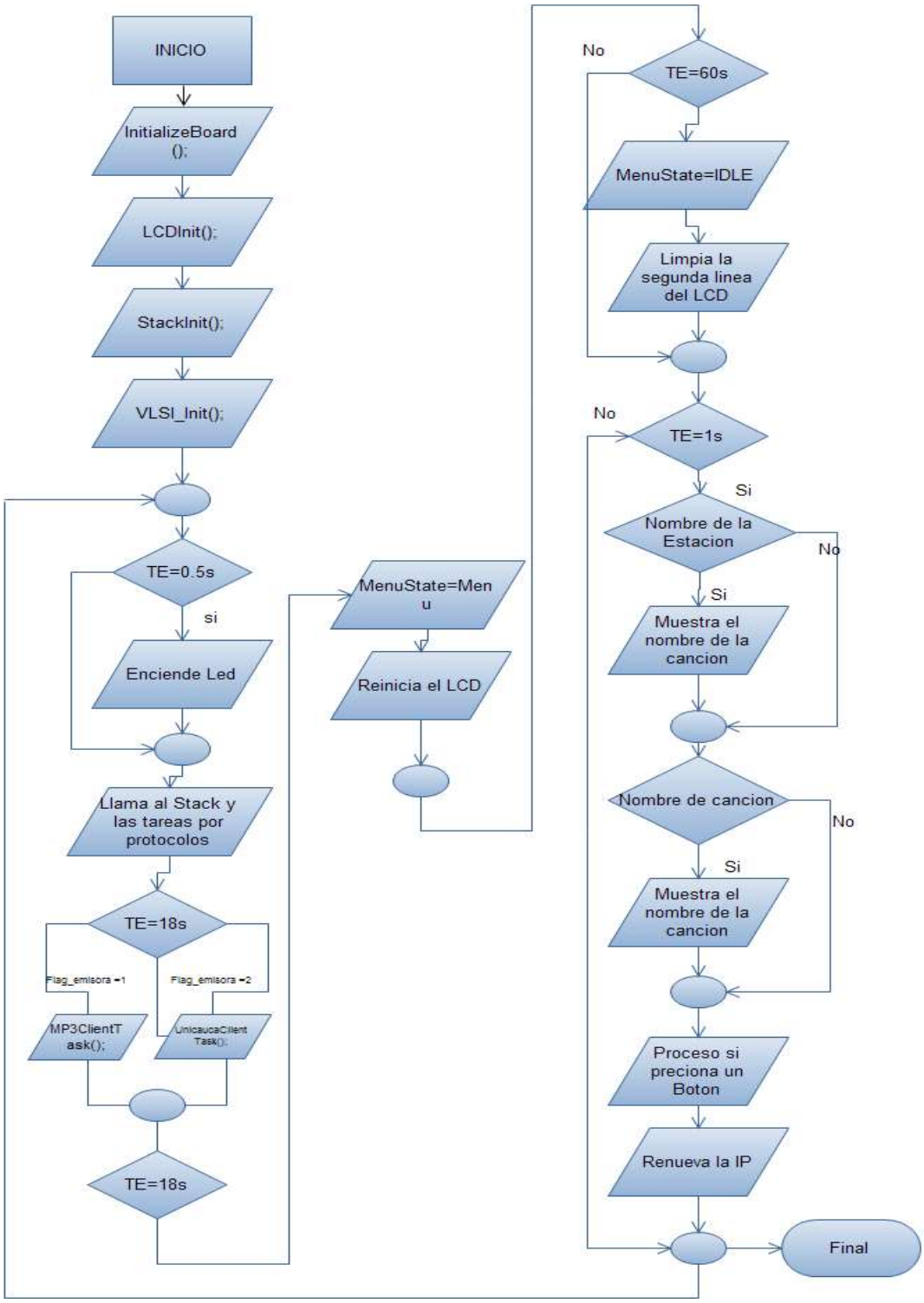

Figura 28. Diagrama de Flujo del Main() para Dispositivo IRD

4.6 FUNCIÓN DE GESTIÓN LOCAL

La función de Gestión Local incorpora las interfaces físicas propias del dispositivo como lo son el LCD como medio de visualización de la información en reproducción y los interruptores como mecanismos para navegar entre las diferentes opciones (Ver Figura 29) como lo son Selección de Modo de Funcionamiento, Genero, Estaciones, Volumen, Bajos.

Podría decirse que el funcionamiento de la función de Gestión Local ha venido dándose a conocer a lo largo del Capítulo 4, ya que su lógica está basada en lo explicado en las secciones anteriores además de que presenta una metodología bastante intuitiva que se basada en la navegación de menús, adicional a esto es importante mencionar que en el prototipo desarrollado al involucrar elementos externos como el LCD fue necesario desarrollar una librería encargada del manejo de este elemento, esta es incluida dentro del Anexo 4 correspondientes a el código fuente, fue denominada *LCD.h* y en ella están declaradas las funciones necesarias para inicializar y controlar los datos de control y de despliegue que son manejadas.

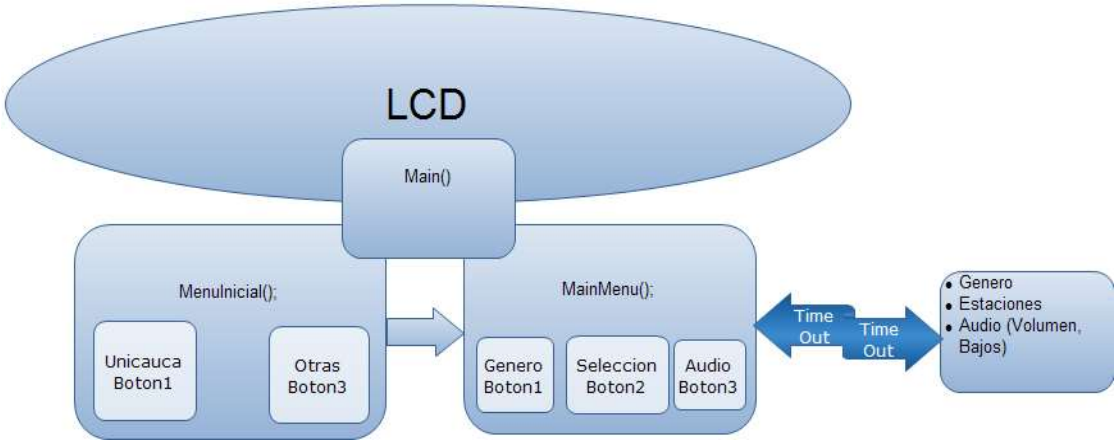

Figura 29. Sistema de Gestión Local

4.7 FUNCIÓN DE GESTIÓN REMOTA

Interfaz Remota

En esta interfaz el sistema despliega una página web en la que da la bienvenida al dispositivo y le permite escoger las opciones de escuchar la Radio Universidad del Cauca o de acceder a otro contenido Streaming de su preferencia (pidiéndole una autenticación). A continuación es mostrada la interfaz principal que se le despliega al usuario (Ver Figura 30). El Anexo 6 entrega un manual con las respectivas instrucciones de cómo utilizar estas interfaces. Es necesario indicar que las páginas deben mantener un diseño sencillo debido a los escasos recursos en memoria con los que puede contarse.

Figura 30. Interfaz Web

Cabe resaltar en este punto que Microchip provee una herramienta denominada MPFS2 Utility[59] la cual permite embeber paginas creadas en lenguaje HTML, JavaScript, entre otros en la memoria del microcontrolador o en memorias EEPROM externas. El funcionamiento de esta herramienta es explicada continuación y la herramienta que fue utilizada para el diseño y apariencia de la página fue Dreamweaver.

La utilidad MPFS 2 tiene muchas características (Ver Figura 31). El propósito principal es empaquetar páginas web en un formato para el almacenamiento eficiente en un sistema embebido. Esto también adhiere variables dinámicas y genera el archivo HTTPPrint.h para asegurar que las llamadas a las mismas sean realizadas el número necesario de veces.

Debido a que usada la memoria interna del programa, indicamos “Arreglo C18” que es el lenguaje que utiliza nuestro compilador para usar con secciones de 8 bits. La herramienta viene integrada en la carpeta que es carga con las páginas web en cada uno de los proyectos, es un archivo por lotes .bat que está integrada con Windows en cualquiera de las versiones y puede utilizar esta herramienta directamente para insertar el código generado en HTML en el código C a compilar.

Figura 31. Herramienta MPFS

La herramienta MPFS escribe el archivo de la imagen en el directorio del proyecto, y también actualiza el archivo HTTPPrint.h, si es necesario. A continuacion es mostrada la forma como incluye la herramienta el archivo sobre el directorio del proyecto (Ver Figura 32).

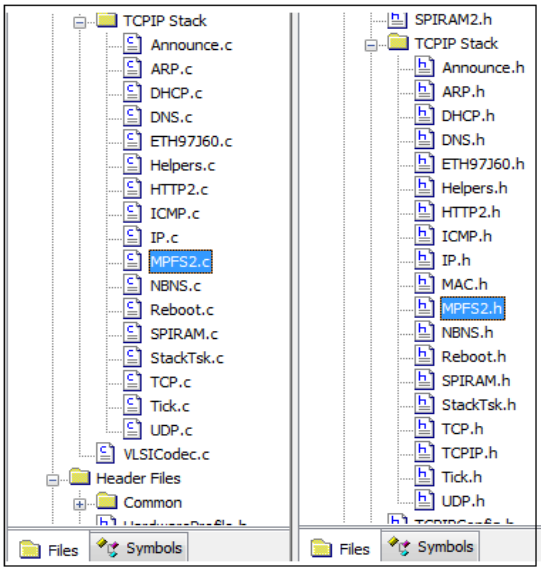

Figura 32. Imagen MPFS2 en el directorio del proyecto.

Es importante indicar que la lógica correspondiente a las diferentes funciones que debe implementar el sistema es definida en la librería CustomHTTPApp.c y es ahí donde fue implementada la función de gestión remota con algunas variables dinámicas y otros botones que tienen funciones específicas como seleccionar un género, la emisora a reproducir, aumentar o disminuir el volumen, etc, y relacionarlas con las diferentes páginas diseñadas. A continuación es mostrado el funcionamiento de estas herramientas (Ver Figura 33).

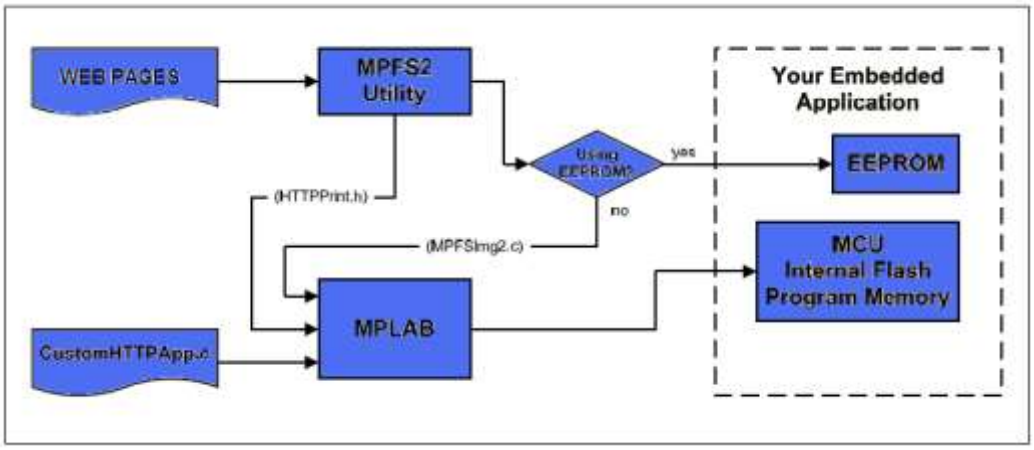

Figura 33. Funcionamiento de Herramienta MPFS

Las variables dinámicas consisten en combinar el sistema de datos dentro de las páginas web. Para poder hacer uso de estas en el código HTML debe colocarse el nombre de la función entre ~~ esto permite que cada vez que en el código sea encontrada esta nomenclatura entienda que es una función conocida como callback lo que simboliza que esta implementada para manejar un evento del programa. Por ejemplo una de la variables dinámicas utilizadas más importantes es la correspondiente a ~genre_list~ y es la encargada de cargar la lista de las emisoras pre-definidas en el main, el sistema busca la implementación de esta función en el *CustomHTTPApp.c*una función del tipo *HTTPprint_genre_list()* y ejecuta las tareas definidas, es así como el sistema interpreta los datos enviados vía http al dispositivo y puede controlar el sistema.

Otro de los elementos importantes en esta fase son los métodos GET y el método POST los cuales permiten enviar valores sobre la URL vía http para que el dispositivo a nivel software pueda interpretar las peticiones solicitadas al hardware ejecutando las peticiones que vienen vía HTML. El método GET fue utilizado en lo correspondiente al envío de peticiones como subir el volumen, cambiar el género, seleccionar una emisora y el método POST fue utilizado para el re direccionamiento entre las páginas web por ejemplo en el momento que el usuario desea acceder al contenido Streaming protegido por contraseña lo re direcciona hacia la página principal. Todo lo que se defina como un método GET debe ser definido en el *customHTTPApp.c* dentro de la función *HTTPExecuteGet*() y el método POST es dirigido en el customHTTPApp.c por una función denominada *HTTPExecutePost*().

Una vez comprendido el funcionamiento de las páginas web y la forma como son integradas al sistema cabe mencionar la forma en que fue realizada la autenticación vía HTML, en primer lugar fueron creados unos botones los cuales utilizan el método POST este método re direcciona al usuario a una carpeta que fue definida como "*protec*" en la

que está alojada la página principal que le permitirá al usuario acceder a otro contenido Streaming diferente al emitido por la Radio Universidad del Cauca.

El sistema despliega una interfaz (Ver Figura 34), esta autenticación fue implementada en el *CustomHTTPApp.c* por una función denominada *HTTPNeedsAuth ()* la cual indica que todas las páginas que estén alojadas en la carpeta "protec" necesitan de una autenticación para poder acceder al contenido. El nombre de usuario y contraseña fueron asignados por defecto como:

Usuario: admin Contraseña: unicauca

Figura 34. Interface de Autenticación

CAPITULO V. ANALISIS Y PRUEBAS

En esta capitulo se realizó el análisis de costos el cual consistió en hacer una proyección en número de unidades equivalentes a comprar los elementos embebidos para realizar el dispositivo IRD y compararlo con un equivalente hardware disponible en el mercado con características similares, posteriormente se indicará los resultados de las pruebas las cuales fueron encaminadas a identificar el correcto funcionamiento del sistema.

5.1 ANÁLISIS DE COSTOS

A continuación esta una tabla con el resumen del análisis realizado correspondiente a los costos para la implementación de este proyecto y una proyección que recoge la información más relevante en cuanto a costos frente a una posible inversión para la producción de hasta 100 unidades IRD (Tabla 18), la información completa referente a este análisis puede encontrarse él en Anexo 5.

Tabla 18. Relación entra la inversión y el costo para cada tarjeta

CANTIDAD	INVERSION	BOARD MICROCHIP	INVERSION	IRD UNICAUCA	AHORRO
1	149.99 U$	99.99 U$	141.27 U$	91.27 U$	8.72 U$
10	1049.9 U$	999.9 U$	819.74 U$	769.74 U$	230.16 U$
50	5099.9 U$	4999.9 U$	3238.6 U$	3138.6 U$	1860.9 U$
100	10099 U$	9999 U$	5028 U$	4928 U$	5071 U$

INVERSION = Costo de la tarjeta + Gasto de envió
PARA 1 Y 10 tarjetas Gasto de envió = 50 DOLARES
PARA 50 Y 100 tarjetas Gasto de envió = 100 DOLARES

Como puede apreciarse el dispositivo implementado por el trabajo de grado en concordancia con el objetivo de desarrollar un dispositivo de bajo costo mantuvo un precio inferior a la solución más económica disponible en el mercado con características similares al prototipo aquí presentado, como era de esperarse al ahorro obtiene al implementar un IRD respecto a la inversión es de alrededor de un 5.18% para lo cual no sería adecuado todo el esfuerzo realizado y tendría mayor factibilidad adquirir la solución propuesta por Microchip.

Sin embargo es evidente que para una cantidad de 10 unidades un ahorro del 21.92 %, para 50 unidades un ahorro del 36.48%, y para 100 unidades un ahorro del 50.21%, indican que es posible llegar a obtener un producto con características similares a un costo inferior en comparación con las soluciones que se existen actualmente en el mercado por otra parte se debe tenerse en cuenta los gastos correspondiente a los

desarrolladores (en este caso específico el desarrollo de este trabajo de grado es una estrategia que aporta en gran medida al desarrollo), costos de desplazamientos, maquinaria utilizada, que elevan en un porcentaje el desarrollo de este tipo de aplicaciones. Pero debe considerarse que este trabajo evidencia que es posible crear dispositivos que cumplen con funciones específicas comparados con soluciones realizadas por fabricantes reconocidos en el mercado, obteniendo un beneficio económico representado en un ahorro para la institución, ahorro que podría ser invertido en las herramientas necesarias para el proceso de diseño y elaboración hardware como el logrado, a su vez estas herramientas benefician al aprendizaje de los estudiantes de próximas generaciones invitándolos a crear soluciones propias y generar sistemas que respondan a las necesidades de la comunidad.

En resumen para la creación de una sola unidad del dispositivo autónomo de reproducción de audio Streaming sobre IP deben evaluarse diferentes aspectos; para comenzar si es desde el punto de vista económico implementar una sola tarjeta un ahorro de 8 dólares no justifica todo el proceso que es requerido para poder llegar a una solución como la lograda en este trabajo de grado, sin embargo sería indebido evaluar este proceso solo en el aspecto económico, debido a que si es considerado que el conocimiento adquirido en la implementación hardware permite ajustar de una manera más eficaz este tipo de sistemas según las necesidades existentes en nuestro entorno. A su vez ese conocimiento permite generar nuevas funcionalidades debido a que se conoce a profundidad el funcionamiento hardware y software lo que permitirían crear sistemas adaptados a las necesidades que existen en determinados ambientes, esto acompañado de una estrategia adecuada para aumentar la producción de más dispositivos (asignaturas como laboratorios) que realmente es donde puede lograrse el beneficio económico es evidente que proyectos como esté pueden beneficiar a la Universidad del Cauca de una manera positiva.

La implementación a nivel software da una visión que es posible crear dispositivos autónomos que llevándolos a un nivel de producción hardware permitan disminuir costos y abrir posibilidades de crear sistemas o prototipos con diferentes funcionalidades, lo que se traducirse en generación de empleo y desarrollo, situación que en estos momentos es de vital importancia para nuestro país.

5.2 Pruebas

Son planteadas tres tipos de pruebas todas encaminadas a comprobar el funcionamiento del dispositivo a nivel Hardware y de su implementación Software: La primer prueba consiste en comprobar el estado de los protocolos que fueron definidos en la implementación de la Pila de Protocolos, para ello fue utilizada una herramienta conocida como *IPTOOLS* y *SnifferHTTP* estas herramientas permiten hacer descubrimiento (Scan) de los dispositivos que se está dentro de la subred (entre ellos el IRD). Identificando protocolos como *ICMP*, *ARP*, *IP* y *HTTP*, comprobando la dirección *IP* y la dirección

*MAC*asignadas por software las cuales habían sido predefinidas en el sistema, las pruebas consistieron en identificar el correcto funcionamiento de estos protocolos.

La segunda prueba consiste en la determinación del comportamiento del dispositivo en cuanto al tráfico recibido por él, para ello fue utilizada la herramienta *ETTERCAPng.*

También fueron realizadas pruebas con respecto al tiempo de funcionamiento del dispositivo sin fallas en sus componentes hardware y software conectado a la red, teniendo presente que estas fallas dependan del dispositivo como tal y no de fallas en la red, para ello se utilizó la herramienta *(PRTG, Network Monitor)* [60]. La prueba consiste en la creación de un sensor *HTTP* que entrega información sobre el tiempo de descarga del dispositivo con respecto a una emisora, concretamente Radio Universidad del Cauca.

Los parámetros para tener en cuenta en las pruebas son la dirección MAC pre establecida en el momento de la configuración del dispositivo MAC: 00:04:A3:01:04:A6 y la dirección IP asignada por el área de servidores de la Universidad del Cauca IP: 192.168.120.11

Protocolo ARP AddressResolutionProtocol (Protocolo de resolución de direcciones).

Es un protocolo de nivel de red responsable de encontrar la dirección hardware (MAC) que corresponde a una determinada dirección IP. Para ello es enviado un paquete ARP request a la dirección de difusión de la red broadcast (MAC = xx xxxxxxxx xx) que contiene la dirección IP por la que se pregunta, y espera a que esa máquina (u otra) responda (ARP reply) con la dirección IP y MAC que le corresponde. Cada máquina mantiene una caché con las direcciones traducidas para reducir el retardo y la carga. ARP está documentado en el RFC 826 (RequestForComments).

Como puede observase en la figura, al momento de descubrir todos los dispositivos de la red 192.168.120 en el intervalo 1 – 254 la herramienta desplegó la dirección IP y la MAC asignadas por software al dispositivo correctamente (Ver Figura 35).

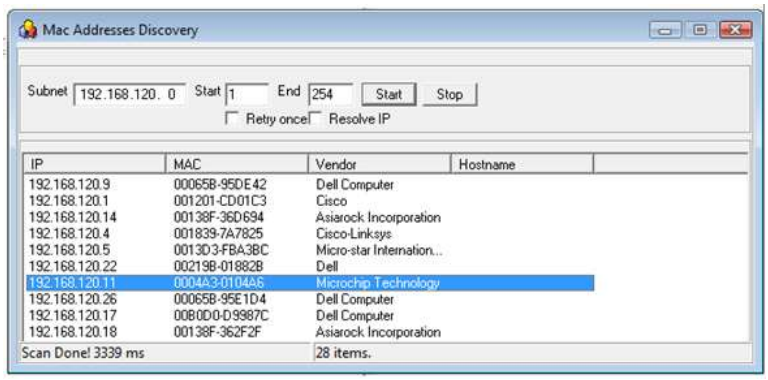

Figura 35. Verificación del Protocolo ARP mediante IPTOOLS

Protocolo ICMP Protocolo de Mensajes de Control de Internet o ICMP *(Internet Control MessageProtocol)*

Fue utilizado un mensaje ICMP en la modalidad de Ping el cual es usado como una utilidad de diagnóstico en redes de computadoras o en dispositivos con características como el implementado, comprueba el estado de la conexión con uno o varios equipos remotos por medio del envío de paquetes ICMP de solicitud y de respuesta. Mediante esta utilidad puede diagnosticarse el estado, velocidad y calidad de una red determinada.Fue útil para medir la latencia o tiempo que tardan en comunicarse un PC y el dispositivo IRD (Ver Figura 36). En la siguiente figura se muestra los tiempos de respuesta al realizar la petición ICMP a la dirección IP 192.168.120.11 asignada al dispositivo IRD, el tiempo promedio de la respuesta fue de 1.066 milisegundos.

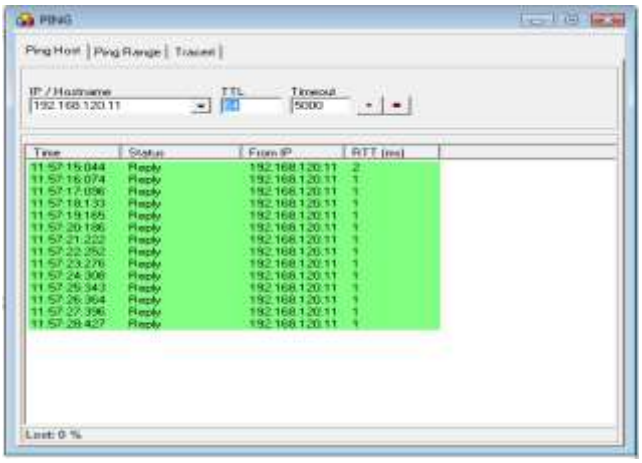

Figura 36. Protocolo ICMP (Ping) desde IPTOOLS

En esta parte fue incluida una verificación extra del protocolo ICMP realizado desde un computador que está dentro de la subred 192.168.0, como vemos la subred se cambió en esta prueba debido a la necesidad crear una pequeña LAN en donde se contara únicamente con el Computador encargado de verificar el estado de la red y el dispositivo desarrollado IRD. Mediante la herramienta de análisis de red provista en la versión 9.04 de Ubuntu, en la gráfica se muestran las estadísticas de paquetes enviados así como el tiempo que tarda el dispositivo en responder la solicitud (Ver Figura 37)

Figura 37. Verificación de Protocolo ICMP desde Ubuntu 9.04

Podemos observar la totalidad de paquetes recibidos satisfactoriamente fueron en 137 paquetes transmitidos e igual número de paquetes recibidos en una media de tiempo de 7.14 ms

Prueba correspondiente al Tráfico recibido por el dispositivo.

Para poder realizar esta prueba fue usado un entorno Linux específicamente con Ubuntu versión 9.04 para ello fue instalado el paquete *Ettercap NG 0.7.3*, esta es una herramienta que permite el análisis de tráfico en una red poniendo en modo promiscuo la tarjeta de red de un computador (Pc Observador) quien será encargado de filtrar el tráfico que tiene como destino el prototipo IRD. El procedimiento fue el siguiente:

1. Abrir la interfaz *Ettercap* y en el menú sniff seleccionar UnifiedSniffing, con el fin de escoger la interface de la tarjeta de red o utilizar y como sniff (eth0).

2. Explorar la red mediante la opción *Scanhost* de esta manera existe la posibilidad de ver los dispositivos conectados a la red dentro de ellos está el dispositivo identificado por su dirección IP y su dirección MAC, con el comando ctrl S inicia el proceso de descubrimiento host (Ver Figura 38).

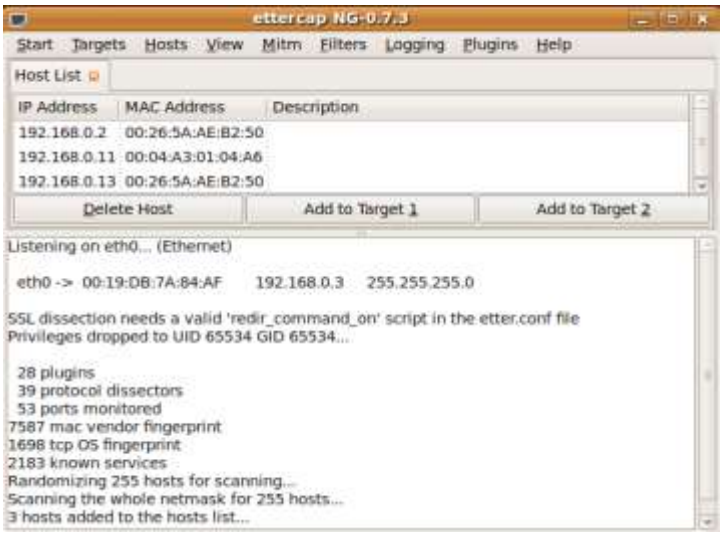

Figura 38. Verificación de dispositivos en Ettercap

3. Seleccionar mediante las opciones Add Target 1 (IRD) sniff (Pc Observador) y en Add Target 2 nuestra “victima” (Dispositivo IRD).

4. En el paso 3 lo realizado es un envenenamiento de las tablas ARP del Pc Observador y del Dispositivo IRD estas acciones son conocidas como un ataque “*Man in theMiddle*” y consiste básicamente en hacer pasar entre la víctima y el PC Observador, es decir, el tráfico desde IRD sea dirigido a nuestro PC Observador convenciendo a la víctima de que se trata del router o dispositivo que da la salida al internet y después de analizar el tráfico es enviado de nuevo al router simulando ser el tráfico que proviene de la víctima, y viceversa.

5. Ahora procede a envenenar las tablas ARP, para ello seleccionar en la opción Mint del *ettercap* y luego ARP Poisoning, y sobre esta opción seleccionar sniffremoteconnections.

6. Por último iniciar el proceso de sniff, con el uso del comando ctrl C es posible visualizar la información correspondiente a las direcciones IP de las estaciones seleccionadas y los puertos por donde es establecida la conexión así como el tráfico que es recibido de la red por el dispositivo IRD medido en bytes (Ver Figura 39).

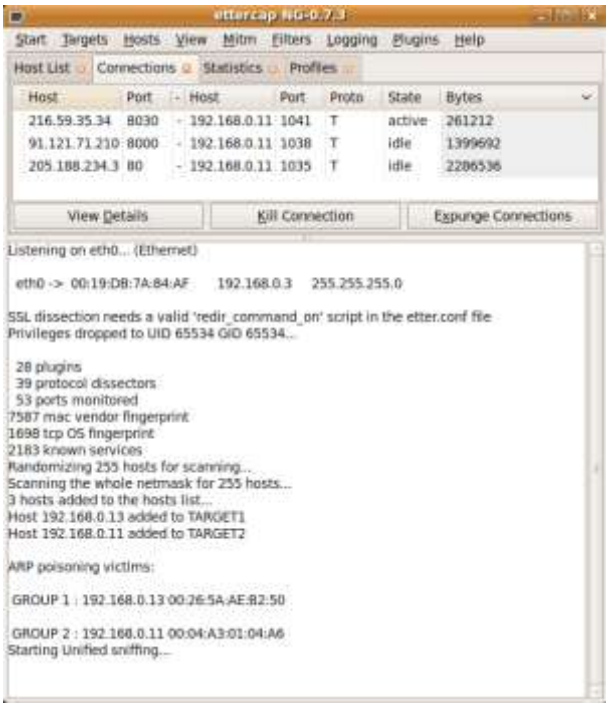

Figura 39. Información del tráfico mediante envenenamiento de las tablas ARP

Las pruebas realizadas mediante la herramienta *Ettercap*permitieron nuevamente evidenciar el buen funcionamiento de los protocolos ARP recuperando correctamente la dirección IP y MAC del prototipo IRD, así mismo mediante el uso de técnicas para captar el tráfico que cursa una red local con destino a un dispositivo especifico fue posible constatar el origen del flujo de Streaming de Audio que estaba reproduciendo en determinado momento el prototipo indicando como puede verse en las figuras anteriores la dirección IP y la medida en bytes del tráfico total que tiene como destino el dispositivo IRD.

PROTOCOLO HTTP Protocolo de transferencia de hipertexto (Hypertext Transfer Protocol)

HTTP define la sintaxis y la semántica que utilizan los elementos software de la arquitectura Web (clientes, servidores, proxies) para comunicarse. Es un protocolo orientado a transacciones y sigue el esquema petición-respuesta entre un cliente y un servidor. El cliente que efectúa la petición es conocido como *useragent* (agente del usuario). La información transmitida es llamada recurso y es identificada mediante un localizador uniforme de recursos (URL). A continuación podemos apreciar la respuesta del protocolo HTTP aunque la herramienta simplemente permite apreciar el estado del protocolo respondiendo OK y debe asignársele el puerto que revisara (Ver Figura 40).

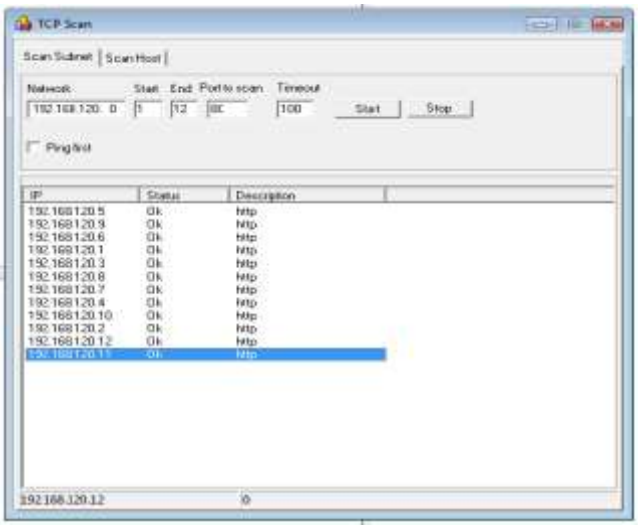

Figura 40. Verificación del Protocolo HTTP

El sistema respondió correctamente a la petición *HTTP* aunque la información arrojada por la herramienta no permite observar con mayor precisión la respuesta del dispositivo. Es posible concluir que el sistema a nivel software y hardware cuenta con el correcto funcionamiento de todos los protocolos implementados realizando cada uno la función para la cual fueron implementados tanto con protocolos *ARP*, *ICMP* y del protocolo *HTTP*. Es importante también aclarar que aunque las pruebas planteadas fueron satisfactorias si es deseado obtener mayor precisión en cuento al tráfico que recibe el dispositivo es necesario la implementación de protocolos como *SNMP* el cual es un protocolo de la capa de aplicación que facilita el intercambio de información de administración entre dispositivos de la red y del establecimiento de un *Agente* el cual es un módulo software de administración de red que reside en el dispositivo administrado, en este caso su alojamiento dentro de la memoria resulto algo imposible, razón por la cual no fue posible implementar dicha prueba pero a pesar de contar con recursos limitados fueron planteadas pruebas alternas como el uso de *snifers* y otras herramientas para el análisis de redes.

La última prueba consistió en obtener una medida del tiempo de descarga del dispositivo IRD Unicauca así como ver el comportamiento estable del dispositivo, primero ubicar la ruta URL asignada por el servidor Nullsoft para la radio Universidad del Cauca y por medio del software para análisis de trafico PRTG asignar un sensor a la URL y medir el tráfico generado con esta conexión, se agregar un sensor PING para medir el estado del dispositivo y comprobar la disponibilidad de la conexión este proceso fue realizado por una hora (Ver Figura 41), (Ver Figura 42), los resultados son mostrados en las siguientes gráficas, cabe resaltar que estas pruebas fueron realizados por fuera de la universidad con el fin de poder medir con exactitud el tráfico generado por la conexión, esto debido a que PRTG mide todo el tráfico que esté en curso en toda la subred en el momento y al estar en un ambiente como Unicauca son miles las solicitudes HTTP que están en curso,

lo cual no permitía tener una medida más acertada para esta prueba la IP asignada en el dispositivo para esta prueba fue:

IP: 192.168.0.11
MAC: 00:04:A3:01:04:A6

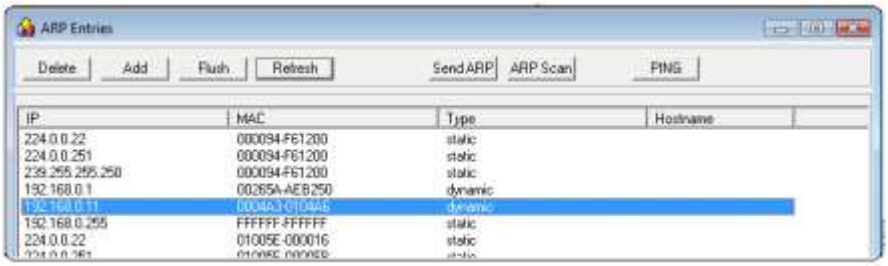

Figura 41. Asignación IP para medida del trafico HTTP

URL asignada por Nullsoft:

http://www.shoutcast.com/shoutcast_popup_player?station_id=1327172&play_status=1&stn=Unicauca Estreo 104.1 FM 180 Años – Popayán

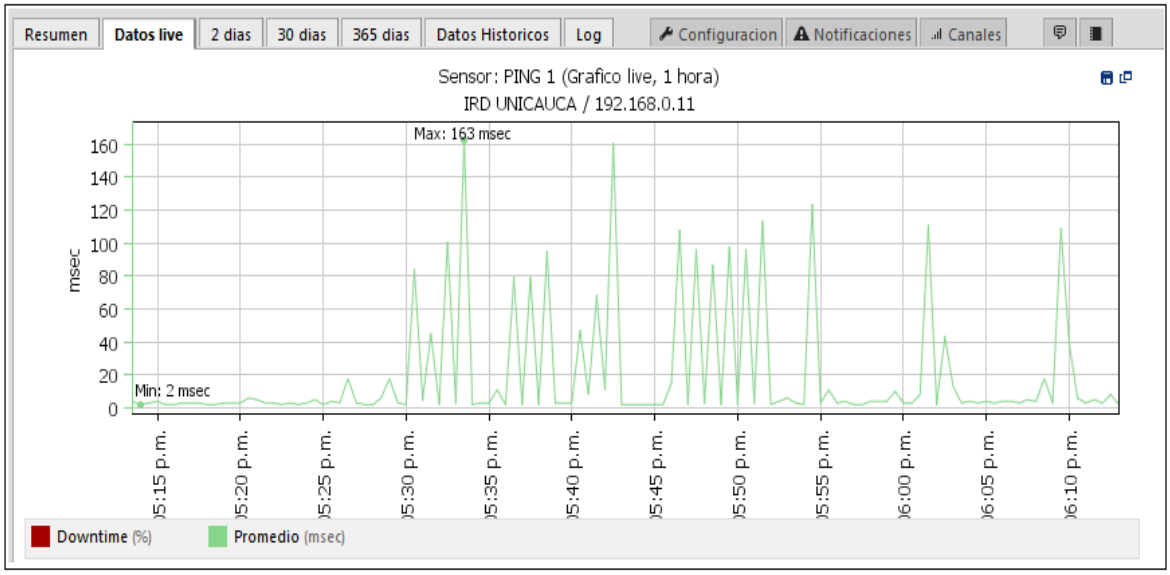

Figura 42. Respuesta Ping realizado al dispositivo IRD desde PRTG

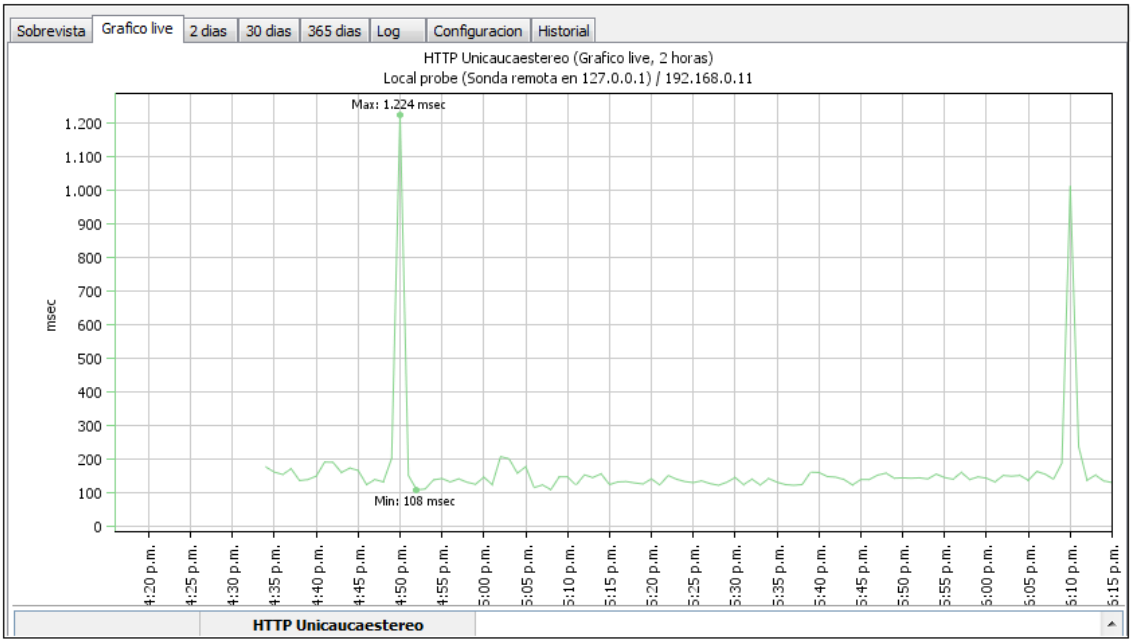

Figura 43. Respuesta del sensor HTTP creado con la dirección IP de Radio Unicauca

La grafica anterior (Ver Figura 43) fue obtenida al asignar un sensor HTTP a la dirección IP correspondiente al dispositivo IRD (192.168.0.11) para el correcto funcionamiento de este sensor es especificado el URL del cual proviene el Streaming de Audio generado por Radio Universidad del Cauca, este sensor permite el monitoreo de un WebSite o de un elemento especifico en el sitio Web, la información generada a través de este sensor nos indica el tiempo en el cual el sensor estuvo activo alrededor de 2 horas (eje x), también es posible apreciar el tiempo de carga medido en milisegundos (eje y).

El comportamiento como lo indica claramente la gráfica fue bastante estable durante el tiempo que transcurrió la prueba manteniendo un tiempo de carga de entre 100 a 200 milisegundos, existieron 2 eventos que incrementaron el tiempo de carga llevándolo hasta 1.224 milisegundos pero no afectaron en ningún sentido la continua reproducción de la emisora Radio Universidad del Cauca y fueron debidos a mas eventos relacionados con la fuente.

La siguiente prueba consiste en verificar los tiempos de carga máximos y mínimos que requieren para realizar una petición HTTP la tarjeta IRD y una tarjeta de red de un computador DELL Inspiro 1545 y así poder realizar una comparación de este protocolo en cada interfaz.

Primero asignamos el sensor por medio de la herramienta PRTG a cada una de las interfaces y las relacionamos con la IP asignada para la Radio Universidad del Cauca

por el servidor de NULLSOFT la cual es http://190.69.2.5:8000. Posteriormente la petición HTTP, los resultados fueron los siguientes (Ver Figura. 44), (Ver Figura. 45):

Sensor Emisora Unicauca Estéreo Respuesta IRD

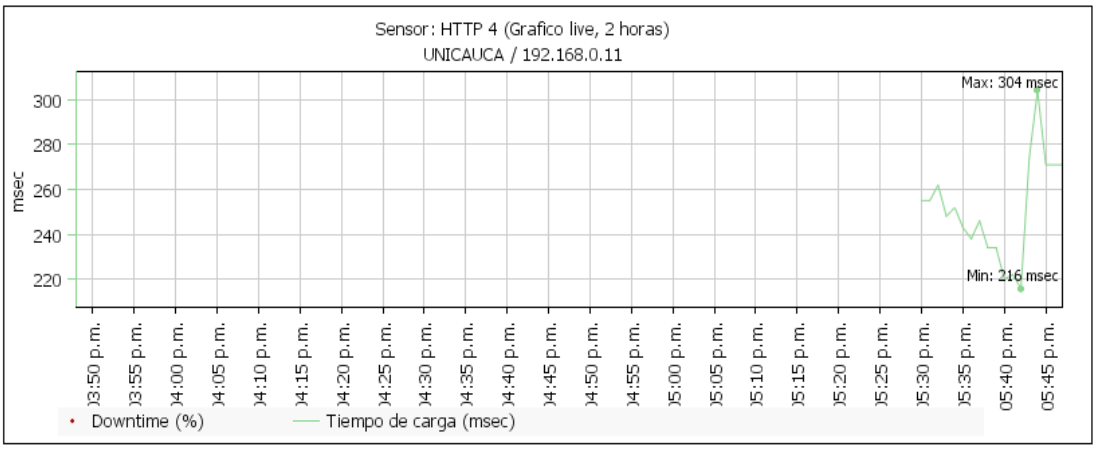

Figura 44. Dispositivo IRD IP (192.168.0.11)

Tiempo de Carga Máximo (msec)=304 msec
Tiempo de Carga Mínimo (msec) = 216 msec

Sensor Emisora Unicauca Estéreo Respuesta Computador Dell

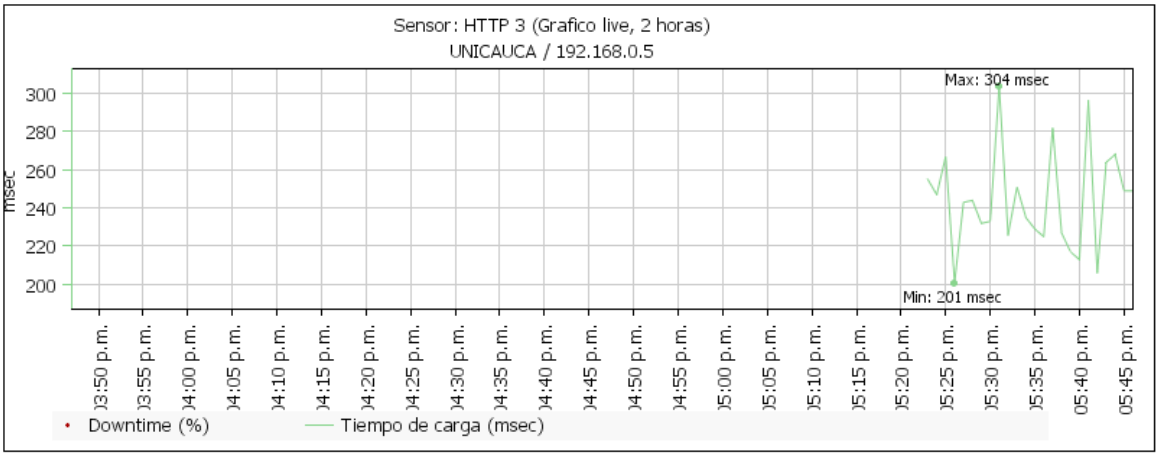

Figura 45. Tarjeta de Red Dell 1545 IP (192.168.0.5)

Tiempo de Carga Máximo (msec)=304 msec
Tiempo de Carga Mínimo (msec) = 201 msec

Posteriormente fue realizado el mismo procedimiento pero esta vez cambiando de emisora la siguiente IP fue la asignada http://66.55.148.27:10054 los resultados fueron los siguientes (Ver Figura 46), (Ver Figura 47):

Respuesta HTTP Tarjeta IRD Emisora 2

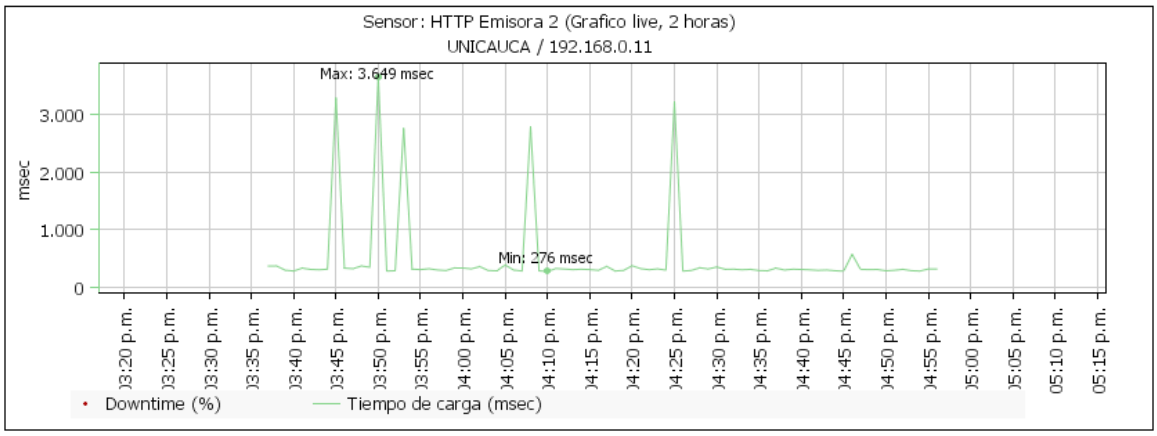

Figura 46. Dispositivo IRD IP (192.168.0.11)

Tiempo de Carga Máximo (msec)=3.649 msec
Tiempo de Carga Mínimo (msec) = 276 msec

Respuesta HTTP Computador DELL Emisora 2

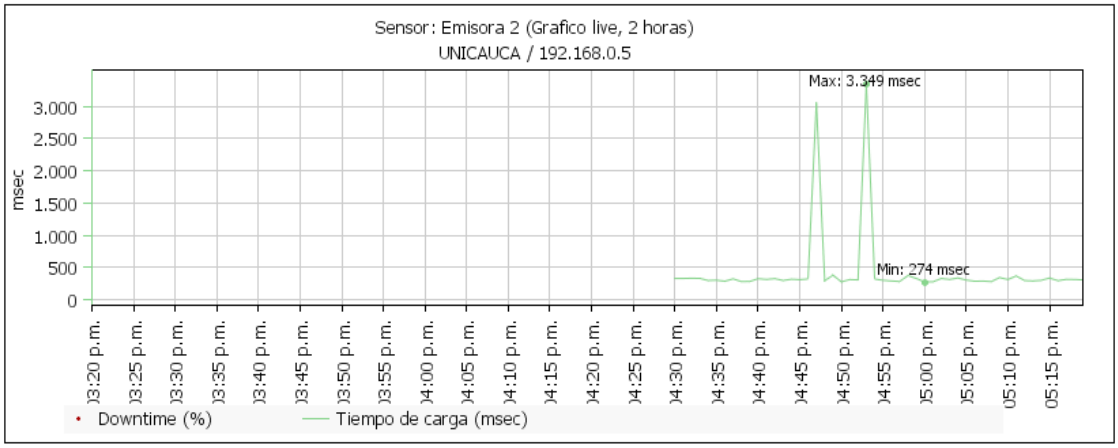

Figura 47. Tarjeta de Red Dell 1545 IP (192.168.0.5)

Tiempo de Carga Máximo (msec)=3.349 msec
Tiempo de Carga Mínimo (msec) = 274 msec

Es posible concluir que a pesar que los recursos hardware con los que cuenta una tarjeta de red como la de un computar DELL 1545 los cuales son superiores a los recursos del prototipo IRD, es posible observar que los tiempos máximos y mínimos de carga son muy semejantes, lo que permite concluir que el hardware y software asignados para esta tarea está en condiciones de funcionamiento bastante cercanas a dispositivos que realizan la misma función y que cumple con altos estándares de calidad.

Para finalizar el dispositivo fue sometido a una prueba final, la cual pretendía evaluar el desempeño de la tarjeta colocándola a funcionar por cinco horas continuas recepcionando la Radio Universidad del Cauca, la prueba pretendió evaluar si el sistema presentaba fallas en su funcionamiento hardware o software exponiéndolo a trabajar durante este periodo de tiempo "exigente", para poder tener una forma de comparar este funcionamiento se dispuso en el mismo instante un portátil Dell 1545 en el cual recepcionaba la radio Universidad del Cauca de esta manera conocer si en algún momento la reproducción se perdía fuera por problemas del dispositivo y no por problemas en la red o el servidor.

El comportamiento de la prueba fue el siguiente: haciendo uso del sensor PING del PRTG el cual verificaba el estado de los dispositivos fue posible observar que durante las cinco horas no existió perdida en el flujo de Streaming en ningún momento, ni por parte del dispositivo ni en el computador lo que nos indicó el correcto funcionamiento de la red, el servidor y los dos dispositivos, además a nivel hardware los diferentes elementos no sufrieron ningún daño como por ejemplo: sobrecalentamiento y a nivel de software el dispositivo respondía a las peticiones que le fueron hechas. La calidad del sonido fue bastante buena y el flujo continuo de la reproducción permitió escuchar el contenido sin ningún contratiempo

Estas pruebas permiten concluir que el dispositivo está en condiciones de ser instalado y puesto a funcionar autónomamente cumpliendo con el objetivo general de desarrollar un dispositivo autónomo que permitiera reproducir flujos de Streaming de audio sobre IP e invitar a la comunidad a interactuar con el dispositivo cuando lo deseen.

CAPITULO VI. CONCLUCIONES Y LINEAS FUTURAS

En este capítulo se presenta las principales conclusiones que arrojo el desarrollo del prototipo funcional IRD y se planteara algunas líneas futuras que evidencian la gran aplicabilidad que puede tener el desarrollo de este tipo de sistemas.

6.1 SOBRE LOS RESULTADOS OBTENIDOS

Los capítulos anteriores evidencian el proceso por el cual se obtuvo el diseño de un prototipo funcional que captura y reproduce Streaming de Audio, partiendo de un diseño físico en el cual fueron estudiados las diferentes soluciones hardware que podrían hacer parte del sistema, teniendo en cuenta y siempre como premisa el desarrollo de un dispositivo de bajo costo, fácil de implementar y de reproducir basándose en herramientas gratuitas y de fácil acceso. Es así como a continuación son planteados una serie de comentarios a manera de conclusión y posteriormente son enunciadas algunas líneas futuras de trabajo que pueden plantearse a partir de lo desarrollado en este trabajo de grado.

- Al plantear el desarrollo de un Dispositivo Autónomo y de bajo costo capaz de capturar y reproducir Streaming de Audio fue generado un estudio comparativo entre elementos como microcontroladores, decodificadores, amplificadores y conversores digital-analógico que proporcionaron una base de selección para la implementación de este dispositivo, al constatar que la elección de estos elementos hace parte de una solución viable no solo económicamente sino también funcionalmente, es posible observar que es posible llegar a realizar desarrollos con características similares y adaptables a nuestro entorno en comparación a las soluciones que encontramos en el mercado y que fueron mencionadas en la *sección 2.4.*

- Es posible mencionar que el proceso seguido para el desarrollo de este trabajo de grado constituye en sí mismo unos lineamientos para el desarrollo e implementación de dispositivos con características similares ya que abarca desde un estudio teórico de la temática a desarrollar, un estudio comparativo de diferentes elementos, el planteamiento de un diseño hardware y su posterior implementación software apoyado de una serie de herramientas igualmente seleccionadas bajo la necesidad de desarrollar un dispositivo de bajo costo y por último una serie de estadísticas de costos que demuestran la posible implementación de este tipo de soluciones acompañadas de unas pruebas que evidenciar su correcto funcionamiento.

- La utilización del stack de TCP/IP provisto por Microchip fue uno de los principales aciertos a nivel software permitiendo la finalización exitosa de este proyecto, en primer lugar porque es una solución gratuita y con una robustez que está siendo probada por múltiples aplicaciones, así mismo, hace que el prototipo IRD funcione de una manera autónoma objetivo cumplido, cabe resaltar que la implementación software en gran parte estuvo encaminada hacia la interpretación y adecuación del Stack de Protocolos provisto por Microchip, teniendo en cuenta que es un código bastante complejo fue necesario un estudio a profundidad de su funcionamiento para posteriormente

modificar en cierta medida algunas funciones y parámetros tratando de adaptarlo a las necesidades y objetivos específicos del proyecto.

- El correcto funcionamiento del dispositivo valida que los protocolos TCP/IP asignados para el Streaming de Audio, HTTP a nivel de aplicación y TCP a nivel de transporte, fueron una solución acertada. Esta conclusión plantea un escenario interesante de estudiar hacia trabajos futuros debido a que actualmente en la parte de Streaming existe una arquitectura bastante robusta la cual indica que a nivel de aplicación deben utilizarse protocolos conocidos como ligeros RTP, RTSP entre otros y en transporte el protocolo UDP. Pero para sistemas embebidos como el desarrollado en este trabajo de grado al presentarse el limitante de recursos hardware es necesario tener estrategias como lo plantea el Stack de Microchip utilizando TCP a nivel de transporte combinado con una función llamada TCP SIN ESPERA DE ACK la cual ejecuta unos tiempos para cada tarea y no necesita esperar los mensajes ACK para ejecutar o cambiar de estado, esta característica es posible aprovechando el concepto de multitasking cooperativo con el que fue diseñado el código, lo que a su vez permite utilizar a nivel de aplicación el protocolo HTTP conocido por su sencillez y fácil interpretación por elementos como el microcontrolador . Esto evita los problemas que presentan protocolos como RTSP tanto para la fuente como para los clientes que desean acceder al contenido Streaming y para este tipo de estructura, tal como fue visto en el capítulo 2, existen empresas como Apple que buscan implementarla para el acceso de sus dispositivos finales y exponen las diferente ventajas y desventajas de esta arquitectura la cual puede estudiarse en una línea de trabajos futuros.

- Con la implementación de la función de selección de contenido fue posible acceder a una oferta de 22 géneros pre-establecidos, a su vez cada uno de ellos cuenta con una lista de 12 emisoras, ello permitió tener a disposición un total de 264 emisoras de todo tipo. Esto indica el potencial que tiene el prototipo IRD y abre las puertas a nuevos escenarios para aprovechar contenidos especializados como por ejemplo emisoras con metodologías para la enseñanza de idiomas por medio de internet y dispositivos como el IRD entre otros miles de contenidos especializados disponibles en la web y otros escenarios en los que podría actuar el dispositivo.

- A nivel de costos es importante mencionar que la selección del hardware y las herramientas utilizadas permitieron mantenernos debajo del promedio de las soluciones existentes, cumpliendo con el objetivo de desarrollar un dispositivo de bajo costo. Pero el impacto real es evidente cuando la producción es mayor a 10 unidades, generando un ahorro con respecto a la inversión requerida para adquirir este mismo número de unidades de la solución disponible más económica. Este ahorro en porcentajes seria para 10 unidades del 21.92 %, para 50 unidades del 36.48%, y para 100 unidades del 50.21%, lo que indica que es posible llegar a desarrollar e implementar un dispositivo con características similares a los productos disponibles en el mercado y a un menor costo.

- Las pruebas propuestas referentes al establecimiento de la conexión del dispositivo con la red y el estado de los protocolos fueron satisfactorias, además la configuración de red asignada estáticamente y mediante el protocolo *DHCP* nos da una idea de la portabilidad del dispositivo. Teniendo en cuenta las características funcionales del

dispositivo y los recursos en memoria disponibles no son considerados elementos extras para el alojamiento de rutinas, como por ejemplo la implementación de un agente SNMP y algunas otras funcionalidades lo cual en cierta forma limitó la exploración de estos protocolos a nivel de pruebas, así como en la creación de una interfaz para el Sistema de Gestión más amigable, aunque teniendo en cuenta que este tipo de desarrollo cumplen con aplicaciones específicas no fue considerado relevante pero si podría ser un aporte como trabajo futuro.

6.2 LÍNEAS FUTURAS

Teniendo en cuenta que este trabajo está enmarcado como un proyecto de desarrollo y considerando el alto impacto que podrían llegar a generar al interior de nuestra facultad, las siguientes son temáticas que podrían proponerse dando continuidad a este tipo de proyectos.

- Aunque el diseño propuesto es totalmente funcional existen diferentes mejoras para plantear que incluso podrían considerarse trabajos futuros, entre ellas son propuestas: La implementación de un módulo para la conexión Wi – Fi disponible en el mercado y que tiene su respectivo soporte dentro del Stack TCP/IP seleccionado en este trabajo, así como el incremento en la capacidad de memoria disponible como por ejemplo el uso de memorias Flash SD con su respectivo modulo o quizás una memoria EEPROM que le dé mayor robustez a los procesos de configuración aquí planteados y así poder incluir entre otras mejoras los protocolos necesarios para llevar a cabo un análisis de tráfico y una aplicación Web embebida con muchas más funcionalidades.

- El diseño general de conexión planteado en este proyecto sirve como base para la implementación de otras aplicaciones Web embebidas como por ejemplo Sistema de Monitorización de Ambientes Controlados, también es posible desarrollar aplicaciones específicas que podrían entrar a aliviar la carga en la red de dispositivos como por ejemplo un servidor de autenticación.

- La ventaja de tener un microcontrolador como unidad central con múltiples puertos de entrada y salida y conectado a internet permite la monitorización, control y evaluación de diferentes variables brindando la posibilidad de desarrollar diferentes sistemas aplicados por ejemplo en control domótica, procesos industriales, medicina, etc. obviamente este escenario exige la implementación de protocolos seguros como por ejemplo HTTPS y algoritmos de cifrado más eficientes para las claves de usuarios y servidores evitando el uso indebido de este tipo de dispositivos.

- Otra posibilidad que queda sin haber sido abordada es la interconexión de varios microcontroladores entre sí, para la creación de una red local de sistemas embebidos.

BIBLIOGRAFIA

[1] Laura Fernández. "*Etek*". España 2009. (Visitado 2009, 9 de noviembre). Disponible en: http://www.ethek.com/escucha-miles-de-emisoras-de-radio-en-internet/

[2] José Ignacio López Vigil. "*Manual Urgente para Radialistas Apasionados*" Este libro tiene todos los DERECHOS COMPARTIDOS (copyleft). Quito, Ecuador. Disponible en: http://www.scribd.com/doc/15073915/ManualUrgenteRadialistas

[3] Universidad del Cauca. "*Información General*". Red de Datos. (Visitado 2009, 9 de Noviembre). Disponible en: http://www.unicauca.edu.co/portafolio/red_de_datos/red_infogral.php

[4] Universia. "*Radio y Televisión Universitaria: dos medios de comunicación en la universidad – La llegada de la red: Internet y los proyectos de radio universitaria*", (Visitado 2009, 10 de noviembre), España 2007. Disponible en: http://universidades.universia.es/

[5] Jorge E. Pereira. "*Radioemisoras e Internet*". Revista Electrónica Mercadeo.com Edición 72 2008. (Visitado 2009, 9 de noviembre). Disponible en: http://www.mercadeo.com/72_radio.htm

[6] Awox, "*Awox Network Media Solutions*". (Visitado 2009, 25 de mayo). Disponible en: http://www.awox.com/

[6.1] Philips: "*Philips Network Music Player*". (Visitado 2009, 25 de mayo). Disponible en:http://www.consumer.philips.com

[6.2] Roku, "*SoundBridgeMusic*". (Visitado 2009, 26 de Mayo). Disponible en: http://www.roku.com/products_soundbridgeradio.php

[6.3] Analisis de Radios en Internet. (Visitado 2009, 25 de Mayo). Disponible en: http://www.ideasgeek.net/2009/07/03/cinco-de-las-mejores-radios-para-escuchar-emisoras-en-internet/

[7] David Austerberry. "The Technology of Video and Audio Streaming". Second Edition. [Documento PDF].

[8] Dapeng Wu*, Student Member, IEEE.* "*Streaming Video over the Internet: Approaches and Directions*" IEEE Transactions On Circuits And Systems For Video Technology, VOL. 11, NO. 3 MARCH 2001. (Visitada 2010, Febrero 2). Disponible en: http://www.ece.cmu.edu/~peha/streaming_video.pdf

[9] H. Schulzrinne et al. "*RTP: A Transport Protocol for Real-Time Application*", Request For Comments (RFC) 3550, Internet Society, 2003.

[10] H. Schulzrinne, A. Rao, y R. Lanphire, "*Real Time Streaming Protocol (RTSP)*", Request For Comment (RFC) 2326, Internet Society, 1998.

[11] J. Postel et al. "*User Datagram Protocol*", Request For Comment (RFC) 768, Internet Society, 1980.

[12] H. Schulzrinne, A. Rao, y R. Lanphire, "*Real Time Streaming Protocol (RTSP)*", Request For Comment (RFC) 2327, Internet Society, 1980.

[13] R. Fielding et al, "*Hypertext Transfer Protocol (HTTP 1.1)*", Request For Comment (RFC) 2616, Network Working Group, 1999.

[14] R. Pantos. Apple Inc. "HTTP Live Streaming".Junio 5 de 2010. (Visitado 2010, 1 de Septiembre). Disponible en: http://tools.ietf.org/html/draft-pantos-http-live-streaming-04

[15] Práctica de Trasmisión de datos Multimedia. "*Transmisión en Internet: Streaming de Audio y Video*". Universidad Politécnica de Valencia. [Documento PDF].

[16] "Awox: Awox Network Media Solutions". (Visitado 2009, 25 de mayo). Disponible en: http://www.awox.com/

[17] Philips: "Philips Network Music Player". (Visitado 2009, 25 de mayo). Disponible en: http://www.consumer.philips.com

[18] "Roku: SoundBridgeMusic". (Visitado 2009, 26 de Mayo). Disponible en: http://www.roku.com/products_soundbridgeradio.php

[19] SilicaAnAvnetDivision. YordiMayne, Ingeniero de Aplicaciones. "*Nuevas Tendencias en los Microcontroladores*". (Visitado 2009, 6 de septiembre). Disponible en: www.silica.com

[20] J. María Angulo Usategui. "*Microcontroladores PIC*". España 2001. 1era Edición. [Documento PDF]

[21] JhonCatsoulis. "*Designing Embedded Hardware*". 2ª Edición. Ed O'Reilly, 2005.

[22] Peatman J. (2003). "*Design whit Microcontroller*". New York, USA, Editorial McGraw Hill.

[23] Douglas V. (2002). "*Microprocessors and Interfacing*".Massachusetts, USA, Editorial McGraw Hill.

[24] Héctor D Ureña Poirier - Juan F Rodríguez Martín. Programa en Nuevas Tecnologías. "*Montaje y Configuración de una LAN*". España. (Visitado 2010, 4 de Mayo). Disponible en: http://www.gobcan.es/educacion/conocernos_mejor/paginas/10base-t.htm

[25] IEEE 802: "*Lan/Man Overview And Architecture*", Disponible en: http://standards.ieee.org/getieee802/802.3.html

[26] lastfm, 2010 Last.fm Ltd,Actualización: febrero de 2009 Disponible en: http://www.lastfm.es/

[27] *Icecast Release*,"***Icecast** is free server software for streaming multimedia". 2009* Disponible en: http://www.icecast.org/

[28] Radio Universidad del cauca, Unicauca estéreo, Disponible en: http://www.unicauca.edu.co/RadioUnicauca/radiounicauca1.php

[29] ISO. "*MPEG Standards - Coded representation of video and audio*".(Visitado 2010, 3 de Mayo). Disponible en:http://www.iso.org/iso/prods-services/popstds/mpeg.html

[30] Tocci, Ronal J – Widmer, Neal S. "*Sistemas Digitales Principios y Aplicaciones*". México 2003. Prentice Hall, 8 Editions.

[31] LM3S6100 Microcontroller, Luminary Micro, Inc, Texas Instruments, [Documento Pdf] disponible en: http://www.arm-cpu.ru/Luminary/pdf/Datasheet_LM3S6100.pdf

[32] LPC2364, Single-chip 16-bit/32-bit microcontrollers up to 512 kB flash with ISP/IAP, Ethernet, USB 2.0, CAN, and 10-bit ADC/DA, NXP Semiconductors, September 2006[DocumentoPdf]. Disponible en: http://www.alldatasheet.com/datasheet-pdf/pdf/159117/PHILIPS/LPC2364.html

[33] Hardware Manual Renesas 32-Bit RISC *Microcomputer SuperHTMRISC Engine Family* SH7780 Series, *RenesasTecnology* 2005 [DocumentoPdf] Disponible en: http://www.renesas.com/products/mpumcu/superh/sh7780/sh7780/sh7780_root.jsp

[34] *Microcontroller solutions for customers' most demanding needs, dallas semiconductor, July 2009*, [DocumentoPdf] disponible en http://russia.maxim-ic.com/design_guides/en/MICROCONTROLLERS_3.pdf

[35] MCF5225x, Family One-stop-shop connectivity MCU with USB, Ethernet and CAN Featuring Freescale MQX software solutions, FreeScale Semiconductor, 2008, [DocumentoPdf] Disponible en: http://cache.freescale.com/files/32bit/doc/fact_sheet/KRN3MCF5225XFS.pdf?fpsp=1

[36] Microchip PIC18F67J60. "*PIC18F97J60 Family Data Sheet 64/80/100-Pin High-Performance, 1-Mbit Flash Microcontrollers with Ethernet*".Microchip Application. [Documento PDF].

[37] Princeton Technology Corp, MP3/WMA Decoder IC , PT8406, March, 2006, [DocumentoPdf] Disponible en: http://www.datasheetdir.com/PT8406+download

[38] *Mpeg 2.5 Layer III Audio Decoder*,STA013, febrero de 2004, [Documento Pdf] disponible en: http://www.st.com/stonline/books/pdf/docs/6399.pdf

[39] VLSI SOLUTION VS1053b. "*OggVorbis/MP3/AAC/WMA/MIDI AUDIO CODEC*". VLSI Aplicación. [Documento PDF].

[40] 16-Bit High Speed Current-Output DAC, Analog Devices, Inc 1999 [DocumentoPdf] Disponible en: http://www.chipcatalog.com/Analog/DAC16.htm

[41] Microchip MCP4725. "*12-Bit Digital-to-Analog Converter with EEPROM Memory in SOT-23-6*".Microchip Application. [Documento PDF].

[42] WE –WIDCOM, RJ45 Transformer, 2009 [DocumentoPdf] Disponible en: http://datasheet.octopart.com/MIC24010-5101T-LF3-W%C3%BCrth-Elektronik-Midcom-datasheet-616900.pdf

[43] LM1117MPX-3.3 800mA Low-Dropout Linear Regulator - National Semiconductor Corporation, 2004 [DocumentoPdf], Disponible en: http://pdf1.alldatasheet.com/datasheet-pdf/view/8602/NSC/LM1117MPX-3.3.html

[44] **HD44780U (LCD-II)**, Dot Matrix Liquid Crystal Display Controller/Driver, documentopdfdisponibleen :http://www.datasheetcatalog.org/datasheets/400/81271_DS.pdf

[45] Dr. Ing. Álvaro Rendón Gallón. "Introducción a los Sistemas de Tiempo Real". Popayán, septiembre de 2006 [Documento PDF]

[46] DIEGO LÓPEZ ZAMARRÓN, "Análisis de Sistemas Operativos de Tiempo Real Libres"2004 [Documento PDF]

[47] MILENKOVICH, Milan. "Sistemas Operativos: Conceptos y Diseño". Mc. Graw Hill, Madrid, España. 1998.

[48] GrupMicriμm. "μC/OS-II TM Real-Time Operating System". [Documento PDF]

[49] $SALVO_{TM}$. "the RTOS that runs in the place tyni". User Manual. Version 3.2.2 [Documento PDF]

[50] "SOS - Small Operating System" http://sos.skydan.in.ua/

[51] DIEGO LÓPEZ ZAMARRÓN, "Análisis de Sistemas Operativos de Tiempo Real Libres"2004.[Documento PDF]

[52] Roberto Blesa Sierra. "*Agente SIP embebido para establecimiento de sesiones VoIP y mensajería instantánea*". UNIVERSIDAD POLITÉCNICA DE MADRID. Noviembre de 2007. [Documento PDF]

[53] Efrén Montenegro Viera. Eduardo Sandoya Tinoco.Ronald Ponguillo Intriago. "*Diseño e Implementación de una Tarjeta de Monitoreo y Control en forma remota a través de Internet, utilizando la tecnología del microcontrolador PIC18F97J60*". Guayaquil, Ecuador. [Documento PDF]

[54] Claudio Iván Durán Marroquín. Emmanuel Gutiérrez Rojas. "*Diseño y Aplicación de un Sistema de Adquisición y Transmisión de Datos vía Ethernet*". Universidad Autónoma del Estado de Hidalgo. Agosto del 2009. [Documento PDF]

[55] Explaining BSD. "Berkley Software Distribution".(Visitado 2010) Disponible en: http://www.freebsd.org/doc/en_US.ISO8859-1/articles/explaining-bsd/index.html

[56] NileshRajbharti. Microchip Technology Inc. "*The Microchip TCPIP Stack*". Microchip Documentation AN833.2008 [Documento PDF].

[57] Rodger Richey – Steve Humberd. Microchip Technology Inc. "*Embedding PICmicro Microcontrollers in the Internet*".Microchip Documentation AN731.2008 [Documento PDF].

[58] Information Sciences Institute University of Southern California. "*IP: A Internet Protocol*", Request For Comments (RFC) 791, Internet Society, 1981.

[59] TCP/IP Networking, *Part 3 Advanced Web-Based Control*, *Microchip TCP/IP Stack* HTTP2 Module, *MPFS2 UTILITY*, [Documento PDF], disponible en: http://techtrain.microchip.com/webseminars/documents/TCPIP_p3_092007.pdf

[60] *PRTG, Network Monitor, disponible en* http://www.paessler.com/prtg/