Reproducción colaborativa inalámbrica de audio en tiempo real

Juan José Hincapié Mazo

Director
Ing.Gustavo Ramírez MSc

Pontificia Universidad Javeriana
Facultad de Ingeniería
Departamento de Electrónica

1 Junio de 2018

Resumen

Este trabajo de grado solucionó de manera parcial el problema de comunicación multicast usando la tecnología de Bluetooth para aplicaciones de tiempo real en Bogotá D.C, Colombia, a partir de los conceptos de piconet y sincronización basadose en principios básicos del protocolo IEEE 1588, que permite sincronizar distintos nodos esclavos a partir de una solo fuente maestra.

Índice general

Índice de figuras

Capítulo 1

Introducción

El desarrollo y popularización de redes inalámbricas en el área de multimedia trajo consigo nuevos retos. Cuando se trata de una reproducción colaborativa se habla de un grupo de dispositivos que conjuntamente generan una mejora visual u auditivo, en ella se pueden lidiar con pérdidas de datos temporales, pero no con la asincronía por lo que se requiere de protocolos de tiempo real. Esto evidencia que la transmisión de audio/vídeo debería ser una parte crítica de la infraestructura digital inalámbrica y aunque existen protocolos de tiempo real como NTC (*Network Time Protocol*) desde hace un tiempo considerable, son muy pocos los orientados hacia la reproducción colaborativa de multimedia, entre ellos están el PTP (*Precision Time Protocol*) y el *eSync* que, aunque permiten la sincronización son muy robustos y se aplican tanto en multimedia como en control industrial.

Con el auge tecnológico de redes PAN como Bluetooth y WLAN (*Wireless Local Area Network*) la necesidad de compartir información en tiempo real aumenta y siendo la multimedia uno de los ítems de más consumo, es casi imperativo desarrollar o mejorar, técnicas o protocolos orientados al área de reproducción colaborativa multimedia de alta calidad. Optimizar métodos para la reproducción colaborativa no solo resultaría en un ahorro de cableado, sino también innovación en publicidad, usando un grupo de pantallas interactuando entre sí, por ejemplo, o mejoras en la acústica mediante control de sincronización para generar efectos de reverberación, son unas de las muchas aplicaciones para tiempo real que se pueden lograr para la creciente demanda de tecnológica en el área multimedia.

La idea inicial de este trabajo de grado deriva del interés en la tecnología Bluetooth en combinación con el audio. Originalmente concebido como una alternativa inalámbrica a los cables RS-232, Bluetooth pretendía conectar varios dispositivos a lo largo de una distancia. Desde su concepción en 1994, Bluetooth ganó popularidad a través de muchas industrias y continúa extendiéndose. Para la industria del audio, Bluetooth comenzó como la tecnología principal detrás de la aplicación de transmisión de audio entre dispositivos portátiles [1].

La transmisión de audio, es una transmisión constante a través de una red de datos que permite a los usuarios escuchar audio casi en tiempo real. Esta tecnología detrás de la transmisión de audio utiliza un sistema de almacenamiento en búfer y una plataforma segura de transmisión de datos para permitir a los usuarios escuchar el audio sin interrupción [11]. Los avances actuales del estándar Bluetooth permiten la transmisión inalámbrica de audio con alta fidelidad, confiabilidad y eficiencia energética por lo que la cantidad de dispositivos Bluetooth con capacidad de transmisión de audio continúa creciendo, específicamente en el sector comercial. Los fabricantes integran cada vez más ésta tecnología en dispositivos de salida de audio (teléfonos celulares, computadoras portátiles, reproductores de audio portátiles, etc.) y en dispositivos de entrada de audio (auriculares, parlantes, etc.) para la transmisión inalámbrica de audio. La compra de dispositivos compatibles con Bluetooth sigue siendo la solución ideal para la transmisión de audio inalámbrica punto a punto, sin embargo, para muchos usuarios y fabricantes resulta inquietante la capacidad de esta tecnología para realizar conexiones de audio multi-punto.

En la actualidad, el estándar IEEE 802.11 ha sido la base para casi todo el desarrollo de conexiones multi-punto en audio, bien sea analámbrica (Ethernet) o inalámbrica (WiFi)[5]. Éste estándar tiene también buenas características de fidelidad y confiabilidad sumado a las funciones naturales de broadcast y multicast, sin embargo, carece de una de las caracterísitcas más importantes por las que Bluetooth es más usado en dispositivos móviles, la portabilidad. Desarrolar un tipo de conexión multi-punto en Bluetooth permitiría entonces tener ventajas similares a las que se tienen con 802.11 sumando a la portabilidad del sistema. Por lo anterior, este trabajo de grado establece un método práctico de transmisión de audio inalámbrico hacia multiples nodos y analiza disintas topologías que permitirían la reproducción colaborativa de audio en tiempo real (retraso inadvertido entre la fuente y el receptor). Para esto, se deben tener claro los conceptos generales de Bluetooth, las estructuras o procesos que limitan la conexión multicast y los métodos de sincronización como se muestra en el Capítulo 2, así, con los conceptos y limitaciones claras, pasar al Capítulo 3 donde se diseña un marco de especificaciones que permita identificar los requerimientos del sistema, diseñar las topologías e identificar protocolo de pruebas. En el Capítulo 4 con la información ya especificada escoger componentes que permitan implementar lo diseñado y finalmente con los resultados concluir la viabilidad de lo planteado.

1.1. Objetivo del proyecto

Se desarrollo un algoritmo basado en un protocolo de sincronización para lograr una transmisión multicast en tiempo real que permitiera el control de fase y la reproducción colaborativa de audio inalámbrica dentro de una red PAN (en inglés, *Personal Area Network*) basada en Bluetooth (IEEE 802.15.1) los objetivos específicos que se plantearon para este trabajo de grado fueron:

1. Identificar los parámetros de IEEE 802.15.1 que limiten la transmisión multicast en tiempo real.

2. Identificar los requerimientos de jitter y estabilidad para escoger la fuente de reloj dentro del transmisor.

3. Desarrollar el algoritmo de sincronización basándose en IEEE 1588 como referencia para la sincronización que garantice un error de 20 ms de acuerdo con EBR (European Broadcast Recommendation) R37-2007.

4. Implementar el algoritmo diseñado sobre una tarjeta de computación embebida.

5. Diseñar un protocolo de pruebas que permita evaluar la sincronización de al menos dos receptores con respecto a la fuente para que cumplan los requerimientos de retardos permitido en la reproducción de audio en tiempo real.

Objetivo Específico 1

En la sección 2.1 se discute en detalle todas las características que limitan la comunicación multicast en Bluetooth.

Objetivo Específico 2

En el Capítulo 3 donde se realiza el diseño de las topologías el cuadro 3.2 trata las especificaciones mínimas necesarias del sistema embebido para cumplir con los requerimientos de resolución.

Objetivo Específico 3

En el Capítulo 5 sección 5.3 se describe el algoritmo con las figuras 5.8 y 5.9 y el método para calcular el retardo en la figura 5.10 que en conjunto plantean una solución para sincronizar .

Objetivo Específico 4

Se desarrolla el algoritmo sobre la Teensy y se anexaron los códigos respectivos en el Anexo 7.3.

Objetivo Específico 5

En la sección de Diseño se especificó los requerimientos para realizar las pruebas asi como una prueba con LabView para ver efectos en distancias más largas

Capítulo 2

Marco Conceptual

En este capítulo se busca crear un marco que permita identificar limitantes y requerimientos generales para establecer una conexión multicast de audio Bluetooth en tiempo real. En la sección 2.1 se analiza desde la perspectiva del stack de Bluetooth y la configuración interna. En la sección 2.2 se enfoca en los métodos de sincronización y conceptos importantes en implemetación para audio.

2.1. Limitaciones Multicast/Broadcast Bluetooth

2.1.1. Seguridad

Previo a Bluetooth 2.1 los datos de transmisión eran fáciles de interceptar y poco a poco se convirtió en una preocupación para los usuarios, la SIG (*Special Interest Group*) creo entonces el protocolo de seguridad SSP (*Secure Simple Pairing*) que sigue vigente en Bluetooth 4.0, permite un emparejamiento mucho más rápido y encripta los datos mejorando la seguridad de la comunicación. La figura 2.1 muestra el proceso básico mediante el cual los dispositivos Bluetooth en un enlace cifrado comparten una "llave de enlace" común utilizada para intercambiar datos encriptados. Cómo se crea esa llave de enlace depende del método de emparejamiento.

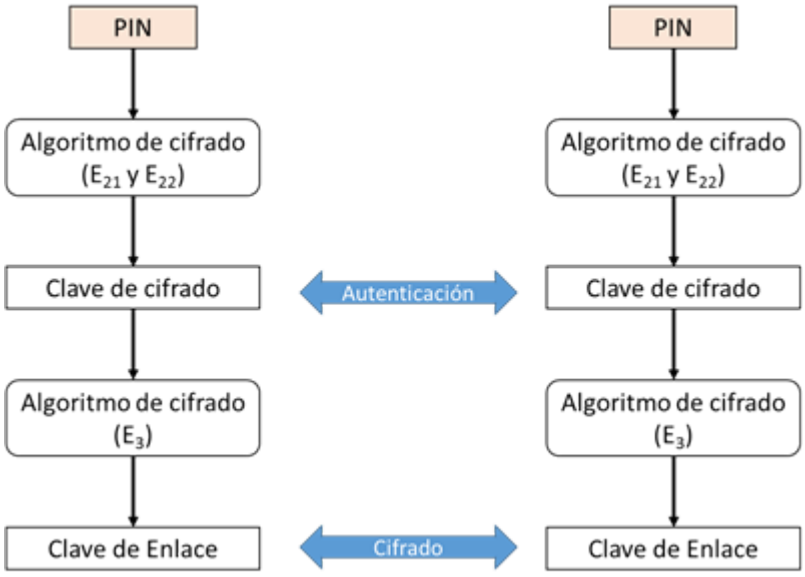

Figura 2.1: Proceso de Seguridad[2]

Para que un *sniffer* Bluetooth pueda descifrar los datos cifrados, también debe tener esta llave de enlace común. Por razones de seguridad obvias, la llave de enlace nunca se envía por aire, por lo que el usuario debe extraer la llave de alguno de los dispositivos emparejados y suministrarla al *sniffer*. Esto implica una limitación importante en comunicación *multicast* ya que existirían dos formas de compartir audio: La primera es quitar el protocolo SSP y retroceder en materia de seguridad o compartir la llave con todos los integrantes de la *piconet* lo cual sería poco versatil si se quiere compartir información con distintos dispositivos.

2.1.2. Acceso y Duplexación

Los dispositivos Bluetooth funcionan en una banda de frecuencia sin licencia entre 2,4 y 2,4835 GHz. Para evitar la interferencia con otros dispositivos que operan en la misma banda, la tecnología utiliza un algoritmo de *frequency hopping* con 1600 saltos por segundo. El tiempo durante el cual los dispositivos operan en una determinada frecuencia se denomina intervalo de tiempo y tiene una duración de 625 microsegundos. Las unidades en una *piconet* cambian la frecuencia al comando de la unidad maestra (*polling*), con base a una secuencia de salto pseudoaleatorio. La banda de frecuencia se divide en 79 canales espaciados a 1 MHz de distancia. Los datos se transmiten en *frames*, que pueden abarcar 1, 3 o 5 *slots*. Los tipos de conexión son asíncrona (ACL) y síncrona (SCO) esta última se usa para el audio ya que prioriza latencia sobre integridad.

En [8] se evidencia un intento de crear comunicación multicast a través de una conexión asíncrona y priorizando paquetes con BNEP (*Bluetooth Network Encapsulation Protocol*) y aunque solo simula el método diseñado, deja un antecedente importante. Para lograr una comunicación *multicast* sin entrar en problemas de seguridad se tiene que poder modificar la capa HCI (*Host Controller Interface*) que es la entidad que maneja el establecimiento de los enlaces, de esta manera tener la facultad de priorizar paquetes y tiempos que permitan mejorar la transferencia de datos hacia distintos nodos de la *piconet*.

2.1.3. Licencias y Accesibilidad

De acuerdo a las secciones 2.1.2 y 2.1.3 es evidente la necesidad de modificar el protocolo de Bluetooth si se quiere generar una conexión multicast tradicional, bien sea a través del acceso al medio o seguridad y aunque es evidente representa un problema también. El desarrollo de comunición *multicast* en tiempo real sigue siendo actualmente un tema en desarrollo, por lo que la mayoría de artículos de investigación desarrollan algoritmos y simulan sobre plataformas, se puede decir que la implementación es casi nula. Existen antecendes como [4] mencionando chips de desarrollo como el BlueCore 3 que fue desarrollado por CSR donde permitía a través de un software modificar los parámetros del *stack* de Bluetooth ,sin embargo posteriormente fue adquirido por Qualcomm restringuiendo el acceso a la información como muestra la figura 2.2, lo mismo sucedió con Circuito Integrados como el RN52 y nRF52840 entre varios otros que usan tecnología CSR.

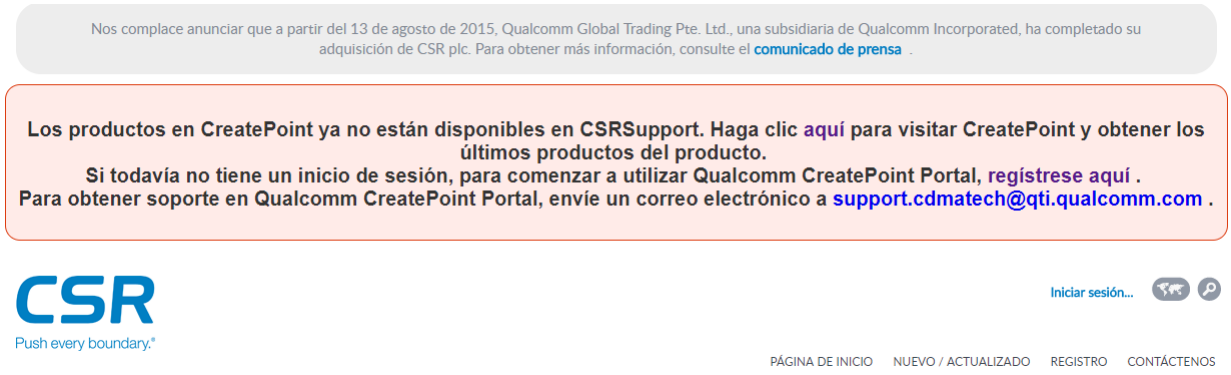

Figura 2.2: Información Protegida[3]

El problema de trasfondo no son las licencias ni restricciones ya que el propietario tiene el derecho por ley, de proteger su desarrollo frente a terceros. El verdadero problema es que debido a las licencias se requiere de permisos del fabricante que valida la solicitud y autoriza el uso de la información, adicionalmente se requiere de una inversión ya que todas las licencias tienen un precio determinado. No siendo menos, si se lograse adquirir una licencia junto con un kit de desarrollo la información de programación a niveles de *stack* es muy limitada, esto implica que se tendría que hacer un análisis exhaustivo para hacer algún tipo de ingeniería inversa y pruebas para lograr la funcionalidad deseada. En palabras más simples, para lograr comunicación *multicast* en Bluetooth la solución ideal es trabajar sobre las capas LLC y MAC, sin ambargo no esta dentro del alcance de el presente trabajo de grado ya que entrar a modificar cualquiera de éstas capas implicaría cambiar el estándar Bluetooth y por ende el funcionamiento tanto de maestro como esclavo. De esta manera se cumple con el primer objetivo específico, si no es posible modificar el protocolo desde el stack se debe proceder a diseñar tipos de topologías que usen la tecnología actual pero que permitan la comunicación multi-punto.

2.2. Sincronización y Latencia

Existen una gran cantidad de métodos para sincronización, cada uno depende principalmente de la implemetación para la que se necesite. El audio particularmente no requiere un nivel de sincronización muy preciso (del orden de milisegundos), de acuerdo con la EBR (*European Broadcast Recommendation - R37-2007*) un retardo entre dos nodos de 70 ms apenas empieza a ser perceptibles por el oído humano. Si se trata de una red PAN los tiempos de transmisión entre maestro y esclavo son del orden de los nano segundos por lo que la latencia en los dispositivos de audio es la más determinante a la hora de lidiar con sincronización no los trayectos. En la sección 2.2.1 se describirá el protocolo IEEE 1588 PTP que es usado para aplicaciones en control y audio [9], y aunque esta diseñado para trabajar sobre internet, lo interesante del protocolo son los conceptos y procesos que maneja para lograr una sincronización alta. Finalmente, en la sección 2.2.2 se tratan los items mas importantes que generan latencia en el protocolo de Bluetooth y en los dispositivos de audio.

2.2.1. Protocolo IEEE 1588

Éste protocolo es implementado para sincronizar relojes de tiempo real (*RTC*) que compartan una misma red. El principio más básico se muestra en la figura 2.3 donde en un primer momento el maestro genera un mensaje de sincronía, el esclavo guardo el momento es que lo recibe. Posteriormente, el maestro genera un nuevo mensaje con el valor del tiempo en que envío el mensaje inicial, en este momento el esclavo tiene ambos el tiempo es que envío el mensaje y el tiempo en que lo recibió. El esclavo contesta con un mensaje de retardo y guarda el tiempo en el que se envía, el

maestro recibe el mensaje y marca el tiempo en el que lo recibe y finalmente envía el valor de ese tiempo al esclavo. Al final del procedimiento el esclavo termina con 4 datos, el offset de tiempo y valor de retardo entre maestro y eslavo y con esos valores compensa el reloj. Este proceso lo hace por intervalos de tiempo alternando con el envío de datos.

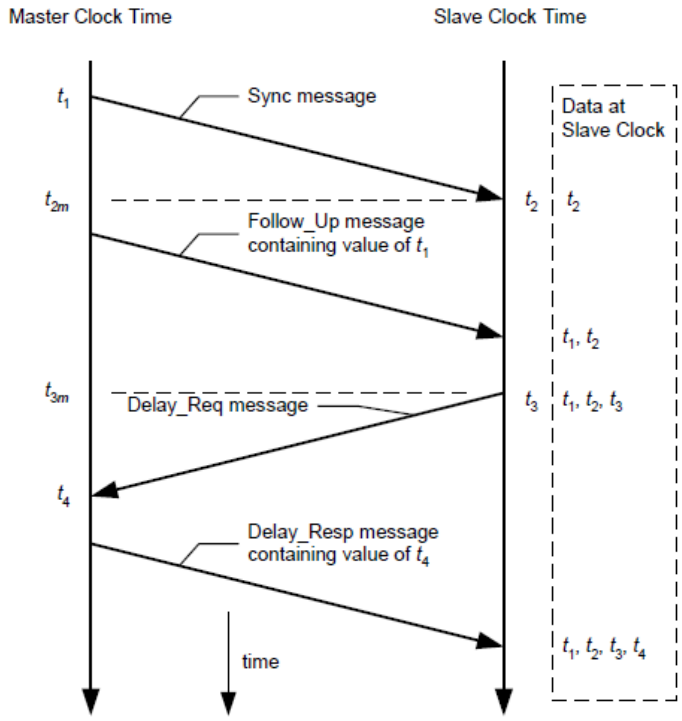

Figura 2.3: Protocolo IEEE 1588[7]

De este proceso existen aspectos relevantes y otros que no son necesarios. El *RTC* aunque permite evaluar el tiempo de llegada y salida, para la aplicación de audio no es necesario, esto debido a que por tiempo real se refiere a un tiempo inadvertido entre el maestro y el esclavo. En un red pequeña los tiempos entre nodos son muy pequeños por lo que se requeriría de un reloj extremedamente preciso y la medida de todas formas no sería relevante. Quiza lo más interesante y aplicable es la estampa de tiempo ya que permite llevar un control constante en tiempo del retardo y la medición períodica también permite obtener y monitorear los valores de retardo que son bastante aleatorios.

2.2.2. Latencia

Cuando se trata de audio en Bluetooth el perfil usado para el streaming es el A2DP (Advanced Audio Distribution Profile). El códec SBC estándar es muy flexible, pero la mayoría de las implementaciones ofrecen un rendimiento de audio significativamente inferior a la calidad de CD. El uso de SBC con A2DP puede conducir a latencias de sistema largas con retrasos de más de 200 ms y fluctuaciones de hasta 50 ms (jitter) [6]esto debido principalmente a 2 elementos: 1) El retardo en la codificación y 2) el retardo en la capa de transporte que usa una estructura de paquetización. Como resultado, las implementaciones estándar de Bluetooth tienden a ofrecer un rendimiento de audio inferior al óptimo.

Las nuevas versiones de Bluetooth (a partir de la 4) han empezado a remplazar el códec SBC por uno de nueva generación denominado APTX, el cual reduce drásticamente la latencia (alrededor de 40 ms) en la codificación y obtiene resoluciones tipo CD ya que no usa compresión con pérdidas, por lo que valdría la pena hacer un análisis más a fondo de aplicaciones en tiempo real basados

en esta nueva tecnología que utilicen control de retardos entre los nodos y que permitan generar efectos de audio deseados con alta calidad.

Capítulo 3

Diseño

Este capítulo basado en las limitaciones definidas en el Capítulo 2 busca establecer posibles topologías que permitan la comunicación de una fuente hacia varios nodos receptores tratando de mantener el menor retardo posible y garantizar la comunicación multi-punto. De las topologías se obtendrán unas especificaciones mínimas que se determinarán en la sección 3.2. Finalmente, se plantea un protocolo de pruebas que permita vizualizar el reatrdo entre maestro y esclavo/s.

3.1. Diseño Topologías

Para el diseño de las topologías se basó en conceptos de tipos de conexiones posibles en Bluetooth como piconet y scatternet para comparar funcionamientos y eventualmente llegar a una topología ideal para multi-conexión.

3.1.1. Topología Multiplexación Unicast

Esta topología se basa en el concepto de *piconet* y requiere que el maestro tenga la habilidad de hacer conexión punto a punto de audio y multiplexar de canal, es decir, saltar de esclavo en esclavo envíando paquetes de audio como muestra la figura 3.1.

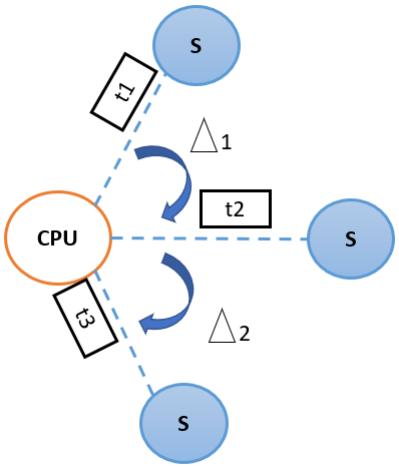

Figura 3.1: Comunicación Unicast

Se debe enviar el mismo paquete a todos los nodos y debido a que es *unicast* el primer paquete debería esperar un ciclo completo de multiplexación para que la reproducción sea simultánea. Es decir, si t1 es el primer paquete y se supone que los deltas (Δ) de multiplexación son iguales, el

esclavo superior debería espera (2*Δ)+ retardo (t2) + retardo (t3). El siguiente esclavo debería esperar solo Δ + retardo (t3). Esta topología entonces agrega por cada nodo un Δ y un retardo (t) al tiempo total de espera.

3.1.2. Topología Cadena

Esta topología se basa en una *Scatternet* la cual busca a través de un audio-link contectar dos tipos de piconets como muestra la figura 3.2. Esto logra que el retardo no este limitado por la multiplexación como la topología unicast, sino por la comunicación entre el esclavo y maestro dentro del *Audio-link*.

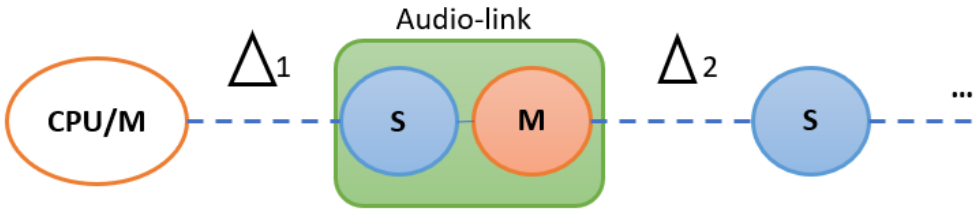

Figura 3.2: Audio-Link

Esta topología elimina el problema de multiplexación pero no el de retardo. Como se observa, por cada Audio-link se agrega un nuevo delta de tiempo lo que muestra que el retardo es acumulativo y los primeros nodos deberá retrasarse N deltas dependiendo del número de nodos que se encuentren en cadena.

3.1.3. Topología Multi-transmisor

Esta topología busca eliminar la acumulación del retardo, en la figura 3.3 se observa que todo los maestros tienen el mismo acceso de información, es decir, que pueden reproducir simultáneamente los datos de audios entonces no existe la necesidad de sincronizar esclavos ya que los retardos son similares.

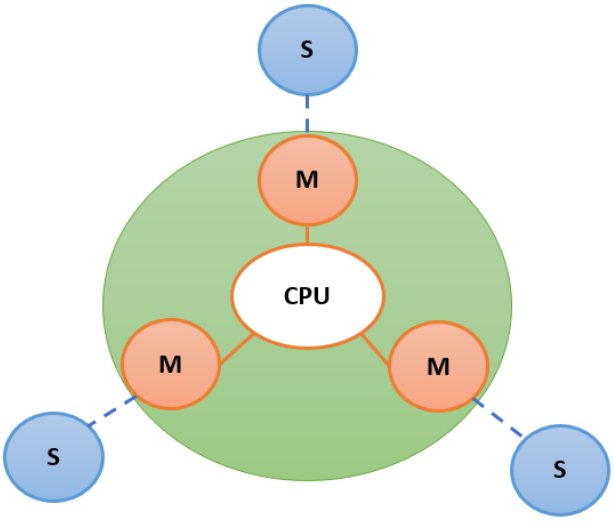

Figura 3.3: Multi-transmisor

Como la topología no requiere sincronización se podría complementar con las anteriores unicast o cadena, ya que no es eficiente usar por cada esclavo un maestro para transmitir audio pero es una buena base para distribuir el flujo de audio a varias ramas y quizas empezar a formar conexiones tipo árbol.

3.2. Especificaciones

Con las topologías de la sección 3.1 medianamente desarrolladas se procede a generar unas especificaciones básicas que deberían tener tanto el módulo Bluetooth como la tarjeta embebida para cumplir los requisitos mínimos de sincronía y retardo.

3.2.1. Módulo Bluetooth

El corazón del este trabajo de grado es el módulo, es el que debe tener las características completas ya que es la parte del sistema que más limitaciones puede generar, el cuadro 3.1 muestra las especificaciones mínimas que debería tener el módulo.

Especificaciones	Justificación
DAC y ADC Resolución min. 16 bits Frecuencia de muestreo min. 44.1 kHz	Esas especificaciones son las mínimas para calidad de CD y garantizar audio de calidad
Clase 2 min.	El alcance es de 10 mts la siguiente Clase ya es para 100 mts que no es necesario
Perfiles: A2DP, AVRCP, SPP y BLE	El perfil A2DP y AVRCP son necesarios para el streaming de audio y control remota de los dispositivos. SPP y BLE permiten comunicación para enviar información de retardos o estados de los dispositivos
Interfaces: UART, I2S, PCM y ANALOG	La interfaz UART permite la comunicación por puerto serial ya sea para recibir o enviar datos o comandos. Las otras interfaces son necesarias para la comunicación de audio entre la tarjeta y otros módulos
Capacidad dual	La capacidad dual es necesaria ya que mientras transmite audio debe ser capaz de enviar y estimar retardos simultáneamente
Multiplexación de canal	Desde hardware y software el módulo debe ser capaz de transmitir por los distintos canales y multiplexar el audio
Códecs: SBC y AptX	Debe ser compatible tanto con SBC y Aptx que debería bajar la latencia total del sistema

Cuadro 3.1: Especificaciones Bluetooth

3.2.2. Sistema Embebido (CPU)

La tarjeta por su parte es secundaria, pero debe cumplir con unos requermientos mínimos al igual que el módulo Bluetooth. El sistema que se use debe garantizar una resolución de reloj menor a un mili segundo esto para garantizar una medida fiable en el calcula del reatardo entre maestro y esclavo

Especificaciones	Justificación
DAC y ADC Resolución min. 16 bits Frecuencia de muestreo min. 44.1 kHz	Esas especificaciones son las mínimas para calidad de CD y garantizar audio de calidad
Interfaces: UART, I2S, PCM y ANALOG	La interfaz UART permite la comunicación por puerto serial y se necesita que tenga por lo menos 3 puertos. Las otras interfaces son necesarias para la comunicación de audio entre la tarjeta y otros módulos
RAM	A 16 bits con 44.1 kHz de frecuencia de muestreo para almacenar al menos un segundo de audio se requiere de mínimo 88.2 kbytes de memoria volátil
FLASH	Debido a que el maestro llevo el reloj de ambos esclavos debe tener más capacidad de procemiento por lo que para una memoria no volátil se requieren de mínimo 256 kbytes
Reloj	Los relojes dependen de la aplicación. Para determinar retardos en audio permite resoluciones de mili segundos por lo que un reloj de 30 kHz es más que suficiente. No obstante, la comunicación por I2S requiere de al menos un frecuencia de 2.8 MHz por lo que el dispositivo necesita como mínimo un reloj de MHz
Progrmación en Python o C++	Con programación de nivel alto se tiene acceso a librerías que facilitan el desarrollo del algoritmo, con nivele bajos sería más tediosa

Cuadro 3.2: Especificaciones Embebido

3.3. Diseño del Protocolo de Pruebas

Como las principales variables del sistema son el retardo y la sincronización, el fundamento más básico que debe tener el protocolo de pruebas es que sea cual sea la medida, todas deben compartir la misma base de tiempo. En una red personal las tiempos de transmisión son despreciables, específicamente en audio. Por lo que vale la pena observar mejor el comportamiento de la sincronización en el tiempo. Por esto se usó el sistema de medida de la figura 3.4 que permite bajo una misma base de tiempo observar el nivel de sincronía de la señales

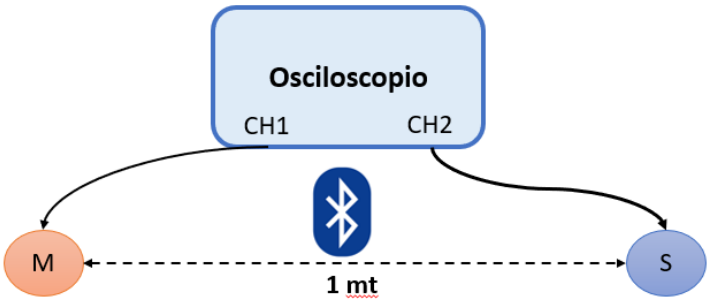

Figura 3.4: Medida Simple

Adicionalmente, se realizó una prueba que se muestra en la figura 3.5. Esto no cambia en nada el retardo que se tenía en la prueba 1 por lo anteriormente señalado, pero permite ver otro tipo de fenómenos como pérdida de señal u obtrucciones.

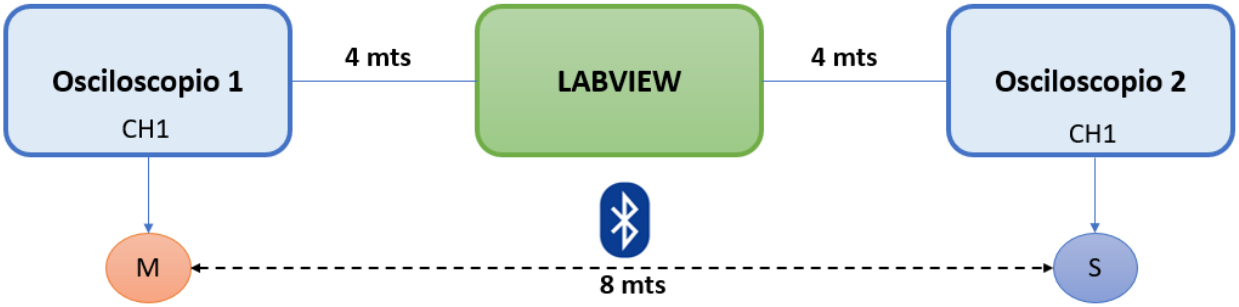

Figura 3.5: Medidas a distancia

Para estas medidas se diseño en LabView una interfaz que permitiera vizualizar ambas señales en sincronía mediante la generación de un *trigger* común, las pruebas y el diseño se encuentras en el Anexo 7.1.

Capítulo 4

Implementación

Este capítulo aterrizará los conceptos tratados en el Capítulo 3 seleccionando los componentes ideales para realizar las topologías planteadas

4.1. BC127 - Módulo Bluetooth

Las capacidades básicas que debe tener el módulo Bluetooth son la transmisión de audio, los modos de transmisión o recepción y la antena integrada. El módulo Bluetooth BC127 de BlueCreation proporciona estas funciones en un paquete relativamente económico y pequeño. El diagrama de bloques del módulo BC127 en la figura 4.1 muestra las principales características del módulo.

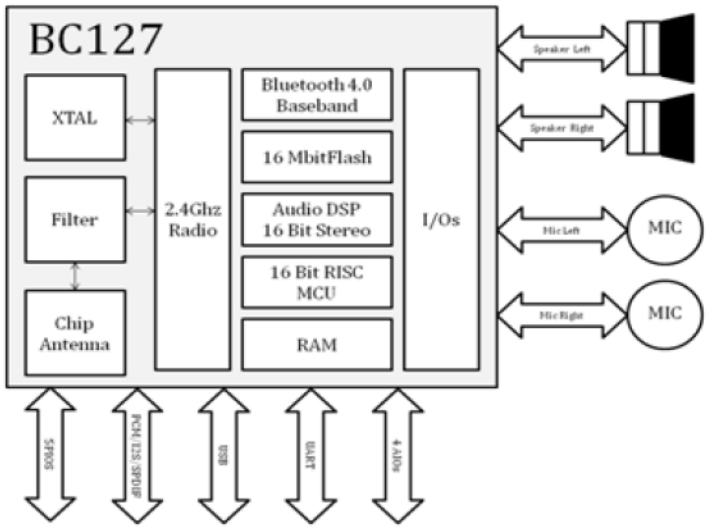

Figura 4.1: BC127 Diagrama de Bloques

4.1.1. Especificaciones de Audio

En términos de la funcionalidad de transmisión de audio, el módulo BC127 tiene todo lo que se necesita para admitir y cumplir con las especificaciones de audio de este sistema. Es compatible con varios perfiles Bluetooth, incluidos los perfiles A2DP y AVRCP, que se utilizan para la transmisión de audio. A2DP (perfil de distribución de audio avanzado) es el perfil que define cómo el audio puede transmitirse a través de Bluetooth de un dispositivo a otro. AVRCP (perfil de control remoto de audio / video) es el perfil utilizado para controlar de forma remota varios dispositivos de audio.

El uso más común para AVRCP es omitir pistas, ajustar el volumen y pausar / reproducir audio de forma remota. El módulo BC127 contiene dos canales de muestreo, lo que permite la transmisión de audio estéreo. La resolución de los convertidores (ADC y DAC) para cada canal es de 16 bits (BC127-HD admite resolución de 24 bits), lo que introduce una degradación mínima del audio transmitido. Cada canal tiene una etapa de ganancia programable analógica y digital, que permite el ajuste del volumen. El diagrama de audio analógico del módulo BC127 en la figura 4.2 muestra el proceso de conversión de alto nivel. Cada módulo puede actuar como fuente (transmisor) o receptor (receptor) del audio, permitiendo la capacidad de intercambio de modos.

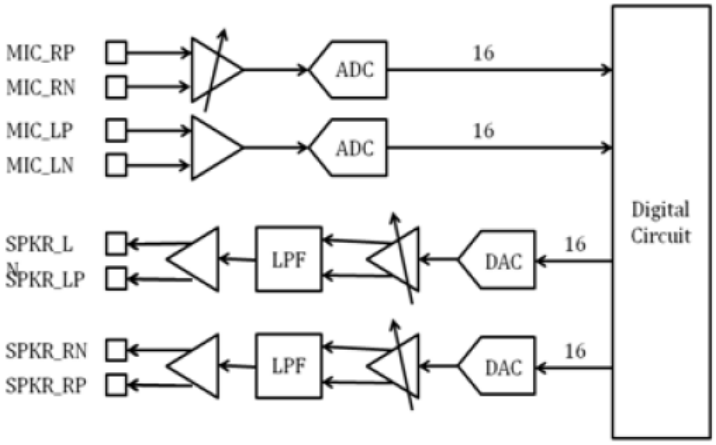

Figura 4.2: Diagrama de Audio

4.1.2. Software (Melody V.5 o V.6)

El módulo BC127 viene precargado con el software BlueCreation Melody (Ahora Sierra Wireless), lo que permite que el módulo se integre y controle específicamente para cualquier sistema. Utiliza UART como su interfaz de comando, lo que permite una comunicación en serie simple entre el módulo y un controlador. El software Melody tiene dos modos de operación que definen cómo se procesa la información proveniente de UART: comando y modo de datos. El modo de datos se usa para los perfiles Bluetooth BLE, IAP o SPP. En modo comando, el módulo acepta comandos del host (microcontrolador) a través de UART. Hay más de sesenta comandos que el módulo admite para controlar las diversas funciones, así como para solicitar datos pertinentes del módulo estos se encuentra definidos en el manual del usuario [10]. Ciertos parámetros del módulo son configurables y se pueden almacenar en la memoria flash para mantener los ajustes después de un ciclo de encendido. Al usar estos comandos de control y configuración, el módulo BC127 se puede controlar para adaptarse a muchas aplicaciones como muestra la figura 4.3.

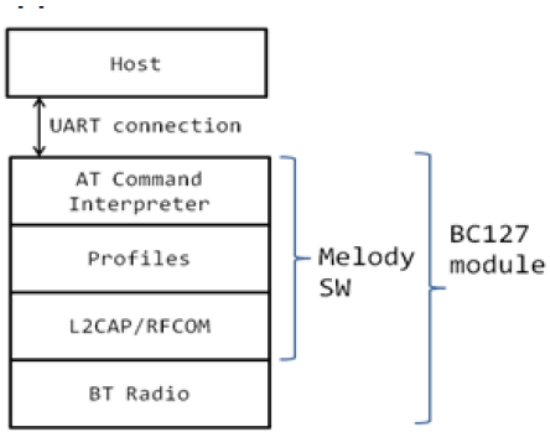

Figura 4.3: Bloques de Componentes

4.2. Teensy 3.5 - Sistema Embebido

Teensy es un sistema completo de desarrollo de microcontroladores basado en USB, en un espacio compacto, capaz de implementar muchos tipos de proyectos. Toda la programación se realiza a través del puerto USB. Específicamente Teensy 3.5 se basa en un Cortex M4 de 120 MHz que trabaja con 32 bits, entre sus características más destacadas están memoria Flash (512 kbytes), RAM (192 kbytes), y las interfaces I2S, I2C y múltiples puertos UART.

4.2.1. Audio Adaptor

Este adaptador de audio de la figura 4.4 le permite agregar fácilmente audio de alta calidad de 16 bits y 44.1 kHz (calidad de CD) a sus proyectos con Teensy 3.0, 3.1, 3.2, 3.5 o 3.6. Admite auriculares estéreo y salida de nivel de línea estéreo, y también entrada de nivel de línea estéreo o entrada de micrófono mono. La Biblioteca de audio Teensy le permite usar la entrada y salida simultáneamente junto con un conjunto de herramientas de objetos de procesamiento de audio, para crear fácilmente todo tipo de sofisticadas aplicaciones de audio. Puede reproducir múltiples archivos de sonido, crear formas de onda sintetizadas, aplicar efectos, mezclar múltiples flujos y emitir audio de alta calidad a los auriculares o alfileres de salida.

Teensy 3.x tiene las instrucciones del Cortex-M4 DSP que proporcionan una gran cantidad de potencia computacional para realizar FFT en tiempo real (análisis de espectro), lo que abre la posibilidad de crear proyectos avanzados de sonido.

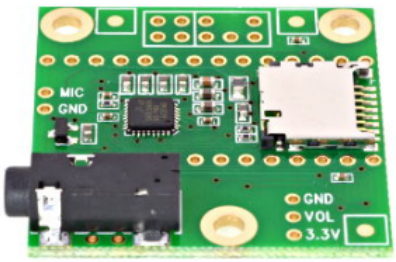

Figura 4.4: Adaptador Audio

El chip de audio SGTL5000, se conecta a Teensy usando 7 señales. Los pines I2C SDA y SCL se utilizan para controlar el chip y ajustar los parámetros. Los datos de audio usan señales I2S, TX

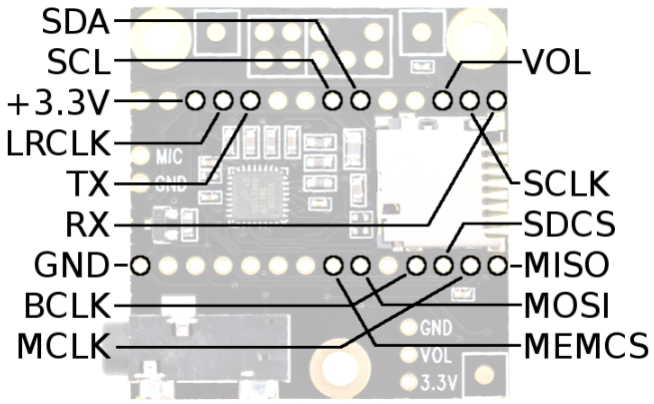

Figura 4.5: Comunicación I2S

(para auriculares y/o línea) y RX (de línea o micrófono), y 3 relojes, LRCLK (44,1 kHz), BCLK (1,41 MHz) y MCLK (11,29 MHz) como muestra la figura 4.5. Los 3 relojes son creados por Teensy 3. El SGTL5000 funciona en "modo esclavo", donde todos los pines del reloj son entradas.

Se accede al *socket* de la SD a través de 4 pines SPI, SCLK y MOSI que se pueden usar en ubicaciones alternativas. La tarjeta SD Sandisk y otras tarjetas SD de buena calidad son capaces de reproducir 2 archivos WAV simultáneamente. Los cables para MCLK, BCLK, LRCLK, TX y RX deben ser cortos ya que al tratarse de señales digitales un cable largo puede prácticamente desaparecer una señal pequeña de audio. Para su óptima conexión se diseño un PCB para facilitar la entrada y salidas de los I/O que se encuentra en el Anexo 7.4

4.2.2. Software

PJRC (Fabricante) se tomó el trabajo de hacer que todas las librerías fueran compatibles con las de Arduino por lo que, a pesar de usar un Cortex se puede usar el IDLE con el que se programa Arduino (C++). Esto facilita mucho el aprendizaje ya que todas las funciones de asignación y ejemplos clásico de Arduino sirven en la Teensy sumándole eso si, potencia computacional. No obstante, también tienen su propio programa que se muestra en la figura 4.6 éste permite hacer conexiones internas a través de la conexión de diagramas de bloque y al exportar se genera un código listo para colocarlo en IDLE de arduino.

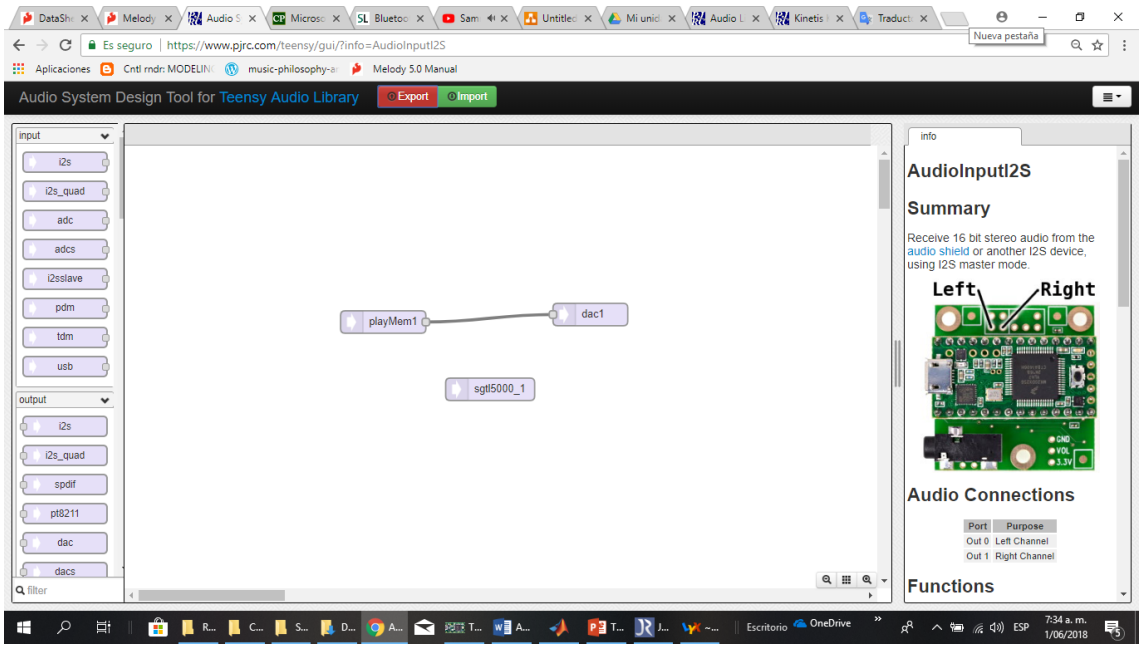

Figura 4.6: Herramienta de Diseño

Del dimensionamiento se tiene entonces un módulo bluetooth controlable por comandos AT y un microcontrolador Cortex que vienen libres desde fábrica. El siguiente reto después del dimensionamiento es lograr la integración entre ambos dispositivos, tanto en comunicación de audio digital por I2S o PCM y una comunicaxión UART para que el controlador pueda acceder al módulo bluetooth y la comunicación de audio efectiva. Una vez realizada ésta integración se puede proceder al desarrollo del protocolo en software para logra el nivel de sincronización de audio requerido y obtener unos resultados finales.

Capítulo 5

Resultados

Teniendo en cuenta lo desarrollado en el Capítulo 3 y el Capítulo 4 se procedió a implementar las topología con el hardware ya dimensionado y las funciones claras.

5.1. SBC vs AptX

Antes de empezar con las topologías estaba la inquietud sobre los beneficios que podría traer el nuevo tipo de códec Aptx con respecto al SBC como se mencionó en la sección 2.2.2. La figura 5.1 muestra la latencia promedio de una conexión punto a punto usando SBC, se esperaría entonces una menor latencia al implementar el otro códec.

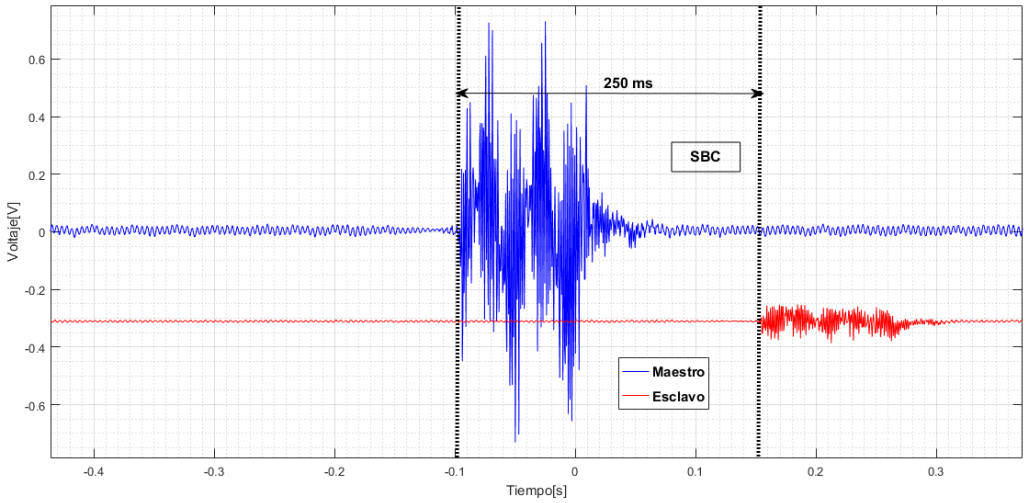

Figura 5.1: SBC

Con el nuevo códec se debe configurar ambos tanto maestro como esclavo ya que los códec SBC y AptX no son compatibles, en la figura 5.2 se realiza la misma configuración punto a punto y se mide la latencia

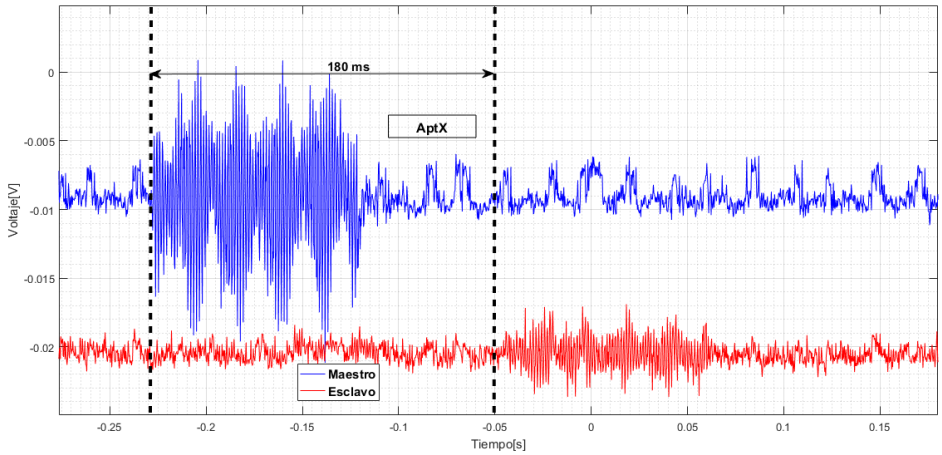

Figura 5.2: AptX

De las imágenes se puede confirmar que el AptX mejora el rendimiento de codificación, ya que sólo en este ejemplo se toma 80 ms menos en codificación que el SBC por lo que de ahora en adelante se implementará en todas las topologías.

5.2. Resultados topología Multiplexación Unicast

Para esta topología se requería de paquetizar las muestras de audio y el módulo Bluetooth debía multiplexarlas a través de sus canales. Desde la librería Teensy existe un objeto que se denomina AudioConnection(), esta librería define por defecto que la transmisión de datos se hace cada 128 bloques o 2,9 milisegundos, asi que en teoría modificando la librería se podría controlar la cantidad de paquetes que son transmitidos. Por su parte, el módulo tiene la capacidad de multiplexar entre los dos canales que posee. Sin embargo, al hacer las medidas de multiplexación se obtuvo como resultado la figura 5.3.

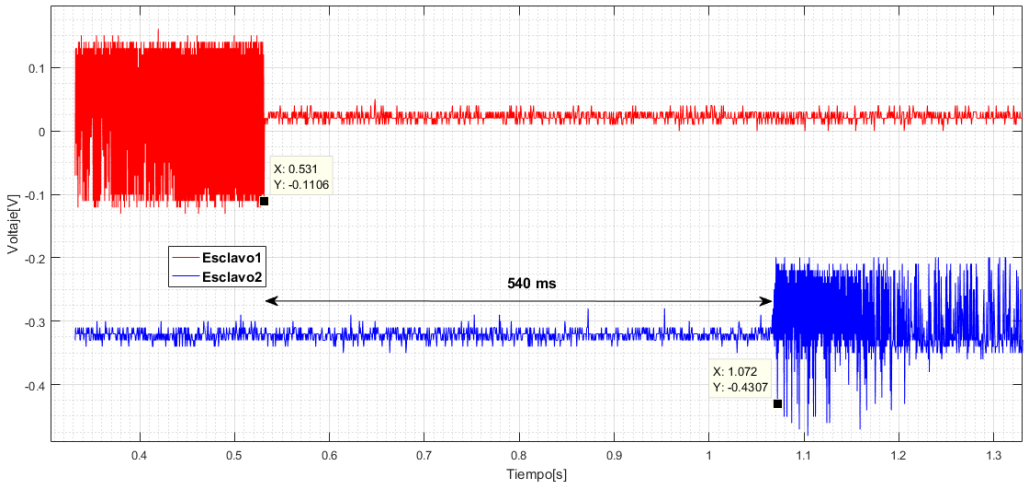

Figura 5.3: Retardo Comando AT

Esto presenta un gran problema ya que 500 ms de tiempo de multiplexación implica que solo

para dos nodos, uno se debe retrasar aproximadamente 1 segundo lo cual no cumple ni criterios de tiempo real ni de sincronización por lo que esta topología no es válida para este trabajo de grado.

5.2.1. Modo dual

Existe una alternativa al problema de multiplexación y se basa en que el módulo BC127 tiene una funcionalidad que se denomina *Dual-Stream*. Esta función permite enviar a dos nodos esclavos audio simultáneamente, entonces no habría necesidad de ningún protocolo de sincronismo y si se requiere generar efectos de eco o reverberación se puede usar las características de BLE y enviar datos sin interrumpir el *streaming* de audio, la figura 5.4 muestra la configuración física implementada.

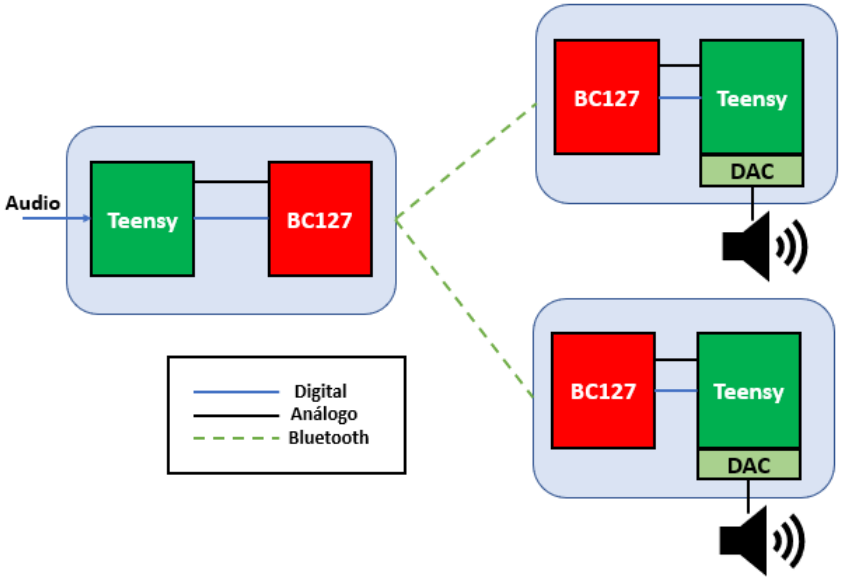

Figura 5.4: Modo Dual

Aunque se resuelva el problema de multiplexación esta configuración no es nueva, ya que tecnologías como Dolby 5.1 usan doble canal para generar sonido estéreo para teatros en casa. Se debe seguir explorando posibles soluciones.

5.3. Resultados Topología Cadena

Esta topología basada en Scatternet crea una comunicación serial de audio como muestra la figura 5.5, pero ahora con la función dual de audio nos permite empezar a derivar el audio en ramas y empezar a crear conexiones tipo árbol.

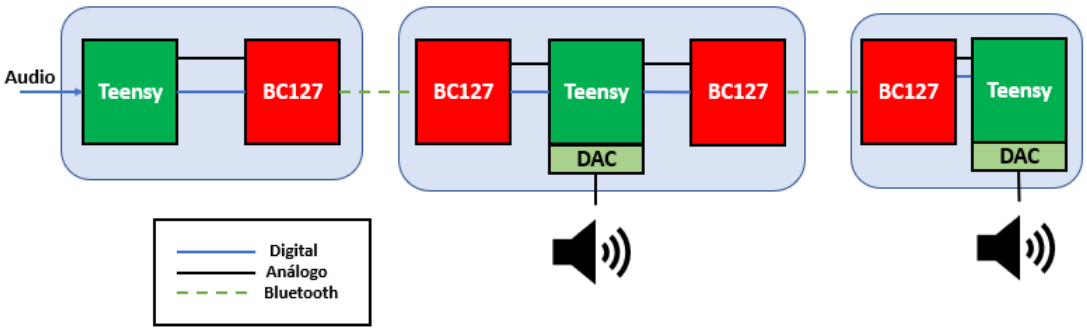

Figura 5.5: Implementación Cadena

En la figura 5.6 se muestra la señal del maestro con respecto al esclavo más alejado para observar el retardo aproximado que en este caso fue de 310 ms y aunque no cumple con el requisito de tiempo real si crea un antecedente importate a la hora de empezar a crear nuevas macro topologías.

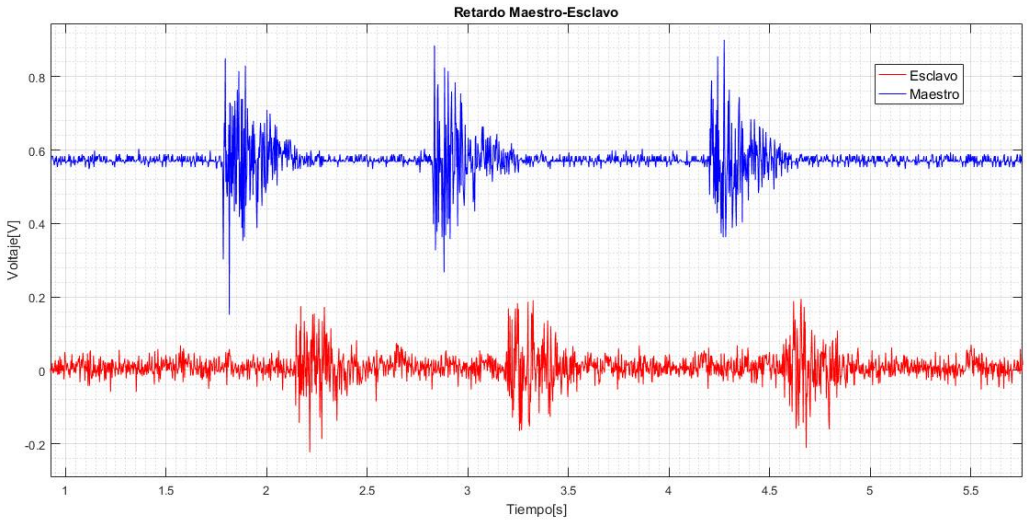

Figura 5.6: Retardo Maestro y segundo esclavo

Por su parte la sincronización entre ambos esclavos se muestra en la figura 5.7 esto muestra que igual sin usar algún protocolo de sincronización el retardo cambia constantemente y de ahí la necesidad de monitorear esta variable periodicamente.

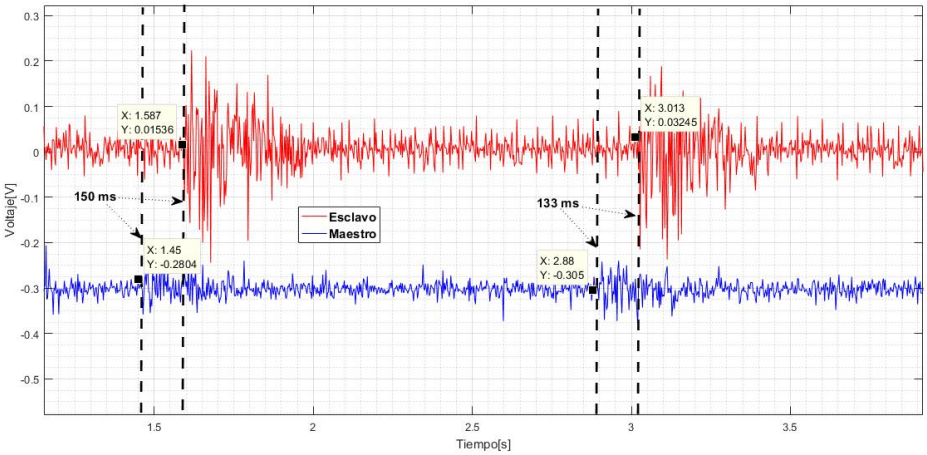

Figura 5.7: Sin Sincronía Esclavos

Curiosamente aparece un patrón y es que por cada conexión Bluetooth se suman entre 150 y 200 ms. Ya al ver visualmente el comportamiento del sistemas se procedió a implemetar un algoritmo que permitiera sincronizar el esclavo 2 con el 1 en la figura 5.8 se observa el diagrama de bloques del maestro y en la figura 5.9 el de los esclavos, el código de esclavos se debe repetir por cada nodo que se agregue. Si el nodo es el último no necesita implemetar todo el código ya que solo debe responder a una petición de retardo

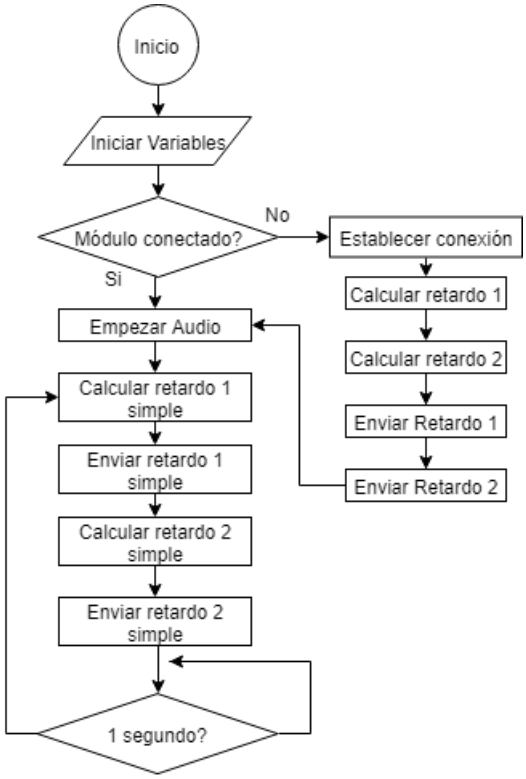

Figura 5.8: Diagrama Lógico Maestro

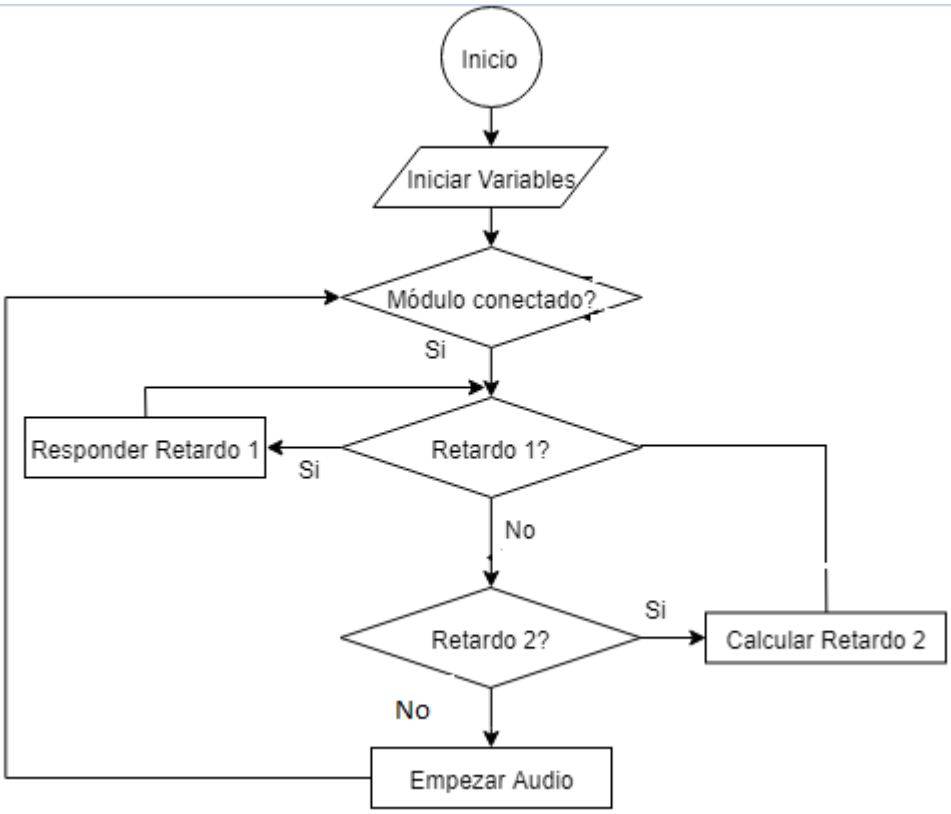

Figura 5.9: Diagrama Lógico Esclavo

El método por el cual se halla el retardo entre nodos tiene principios del procolo IEEE 1588 como se muestra en la figura 5.10 el maestro lleva el reloj de todo el sistema y los esclavos se encargan sólo de recibir órdenes. En un primer momento se establece la conexión y envía un caracter ID, si coincide con el del esclavo responde con un ACK, el maestro registra el momento que envía el ID y el momento en el que recibe un ACK guarda un medio de ese tiempo (suponemos camino simétrico) la primera vez que hace este proceso lo itera 50 veces.

Hace este proceso con el primer y segundo nodo, determina el retardo de cada uno e inicia el *Streaming* de audio. Gracias al comportamiento dual del sistema se puede enviar audio y a las vez revisar periódicamente el valor del retardo en el tiempo.

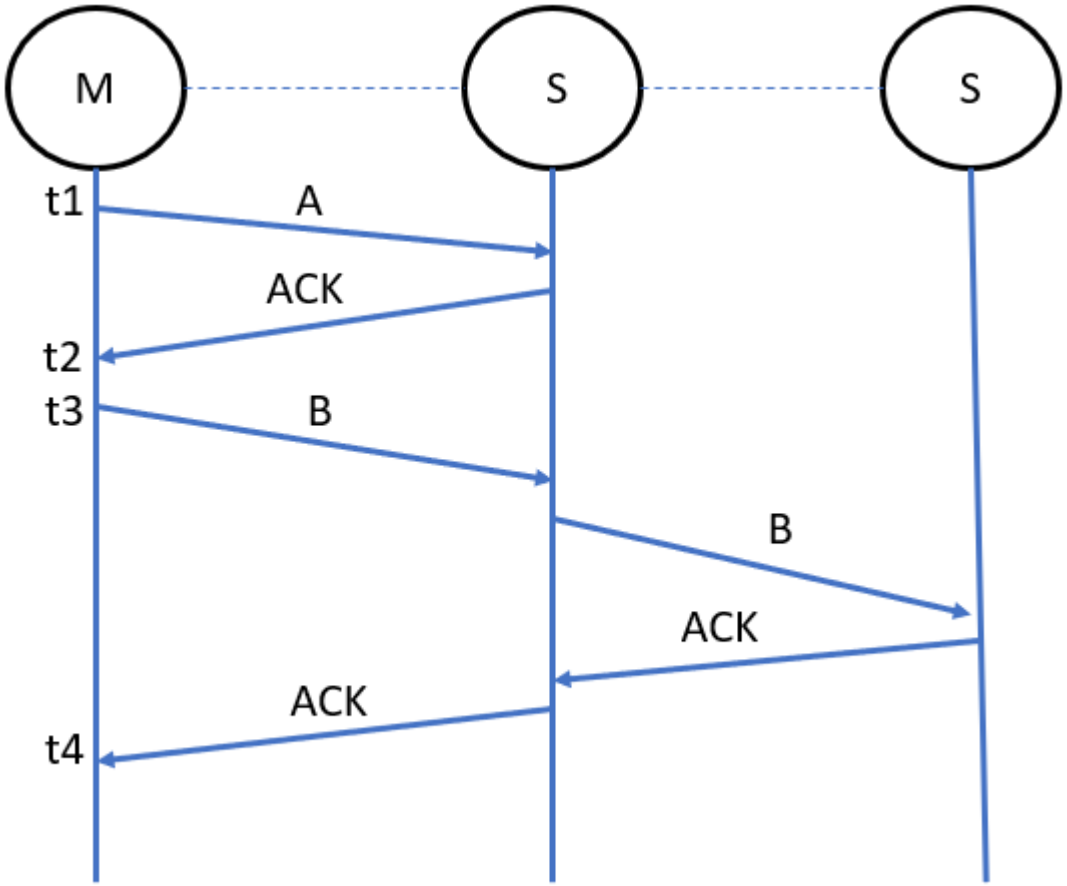

Figura 5.10: Cálculo Retardo

Con los diagramas definidos se corre el código y se tomán medidas de la sincronización, la figura 5.11 es una de las mejores medidas tomadas y aunque muchos intentos se encontraban dentro del rango de tiempo real, con una solo medida que se saliera mucho del rango disparaba la señal lejos de la sincronización. Esto evidencia que el método usado para el cáculo del retardo es poco robusto.

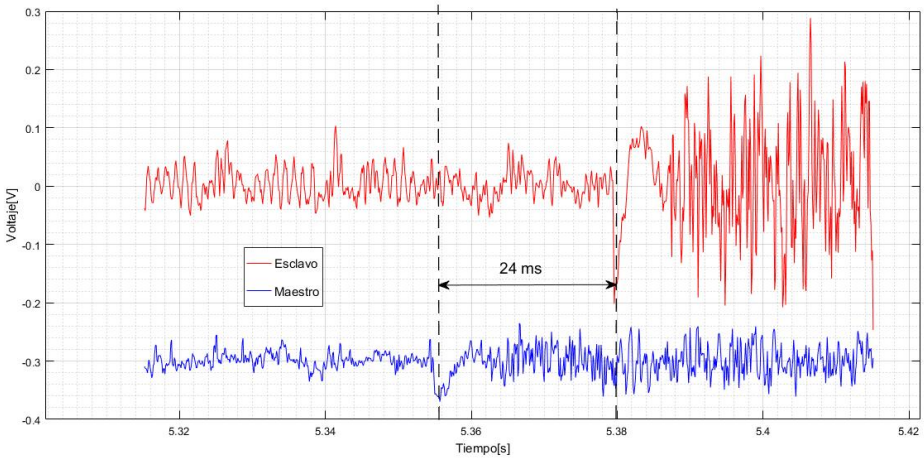

Figura 5.11: Sincronización Cadena

Las medidas de las sincronización y los códigos de maestro y esclavo se encuentran en el Anexo 7.2 para mejores referencias.

5.4. Resultados Topología Multi-transmisor

La figura 5.12 muestra la topología final que se basa en multiples transmisores. Esta topología tiene mucho potencial porque gracias a las características duales del audio del BC127 y la técnica de *Audio-Link* se puede crear una red mucho más grande (si lo que se quieren son cantidad de nodos) con retardos mínimos.

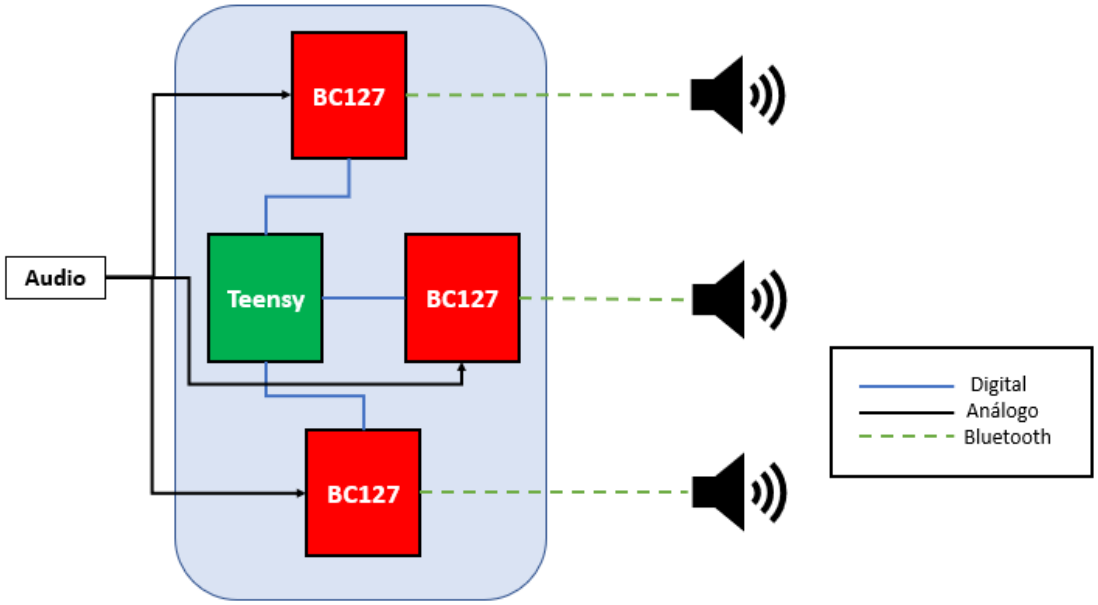

Figura 5.12: Implementación Multitransmisor

Lo que implica la dualidad y los mútiples transmisores es la capacidad de crear estructuras jerárquicas. En la figura 5.13 por cada maestro hay 2 esclavos que a su vez son maestros de otros dos esclavos significa que por cada delta de retardo la cantidad de nodos se duplica. Ahora, sino se tiene solo un maestro, sino 4 más compartiendo el audio simulatáneamente la cantidad de nodos aumenta cada vez más.

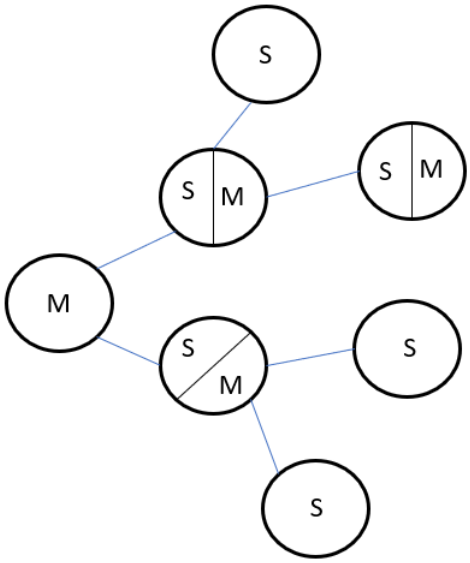

Figura 5.13: Conexión Multiple

5.5. Resultados Comparativos

De las topologías es evidente que la multiplexación no sirve para aplicaciones de tiempo real ya requiere de al menos 1 segundo de retardo, sin embargo se podría utilizar para otro tipo de aplicaciones donde hayan pocos datos y alta latencia. La topología en cadena tiene mucho menos latencia pero al igual que la multiplexión el retardo se acumula a medida que se agregan más nodos al red y tampoco cumple con los requisitos de retardo entre fuente y receptor. La topología multi-transmisor parece ser la más viable, ya que usando la propiedad de canal dual en conjunto con varios maestros se logra la comunicación hacia multiples nodos con el menor retardo posible. Cada una se puede usar en separado o en conjunto dependiendo de las necesidades que se planteen. El protocolo a pesar de ser relativamente sencillo cumplía con la especificaciones de sincronización, exceptuando momentos en los que un mala medida desviaba toda la señal. La tabla 5.1 muestra un resumen corto de lo hablado y permite analizar más visualemente la compración entre las topologías, esto permitiría escoger la más adecuada dependiendo de las necesidades del sistema.

Características	Unicast	Cadena	Multi-transmisor
Retardo al segundo esclavo (Aprox.)	660 ms	310 ms	N.A
Tiempo Agregado por nodo adicional	520 ms	160 ms	N.A
Capacidad estéreo	Si	Si	Si
Aplicación para tiempo real	No	No	Si
Cantidad de maestros	1	1	1 por cada 2 esclavos
Sensibilidad a cambios en el medio	Alta	Alta	Baja
Sincronización	Compleja	Media	Sencilla
Costo	Medio	Medio	Alto (depende de la cantidad de esclavos)

Cuadro 5.1: Comparativa

Capítulo 6

Conclusiones

Este trabajo de grado buscó dar una solución al problema de comunicación multicast en tiempo real con tecnología Bluetooth, usando los conceptos de *piconet* y *scatternet* para crear topologías que aprovecharan los distintos tipos de conexiones, de esta manera tener el mayor número de nodos con el menor retardo posible. Esto se logró, por medio del uso del módulo configurable BC127 y el controlador basado en C++ sobre la tarjeta Teensy.

El módulo Bluetooth BC127 es una herramienta que permite a los usuarios, a través de comandos AT modificar la estructura interna de Bluetoth. Sin embargo, para la aplicación que se quiere realizar en el presente trabajo no es suficiente. Aunque permite una variedad muy grande de configuraciones, en materia de latencia se queda corto, valdría la pena adquirir la licencia para el AptX *low latency* y determinar si se puede garantizar una latencia punto a punto menor a 100 ms de esta forma garantizar audio en tiempo real o posibles conexiones adicionales.

Con respecto a las topologías, debido al tipo de comunicación es evidente que a medida que el número de nodos incrementa también el retardo general del sistema, por ende la capacidad de los maestros debe incrementar igualmente, en palabras más simples, entre más nodos se tiene, menos se puede garantizar el tiempo real. La única topología que valdría la pena trabajar es la multi-tansmisor ya que en un principio no requiere de la sincronización de nodos y da libertad de generar otros tipos de redes.

En la realización de este trabajo de grado, se encontró que el cálculo del retraso entre nodos no es trivial, es decir requiere más investigación y de técnicas más eleboradas para filtrar estadísticamente datos equivocados, prueba de ello es que el protocolo diseñado en este trabajo de grado servía por momentos pero con un dato erróneo se desviaba completamente de la media sobre la que se encontraba.

Por último, a partir de los resultados, y debido a que no se puede modificar el *stack* de Bluetooth la opción más viable para garantizar la comunicación multicast de audio en tiempo real es la multi-trasnmisores, sin embargo por si sola no es la más eficiente tampoco, así que la mejor opción (también dependiendo de las necesidades) es implementar un red que tenga como base ésta topología y complementarla con otros tipos de conexión. Para futuros trabajos de grado, se recomienda usar módulos con licencias *AptX Low Latency* para mitigar al máximo el retardo y adicionalmente investigar sobre cómo se podrían modificar las capas LLC o HCI del Bluetooth para lograr una comunicación multicast o como una nueva tendencia está usar el BLE (*Bluetooth Low Energy*) para comunicar audio teniendo en cuenta que este si cuenta con método broadcast para información pero es limitado en capacidad de transmisión de datos.

Capítulo 7

Anexos

7.1. Mediciones e Interfaz LabView

https://drive.google.com/drive/folders/1eAA2K9RqG3jP91laUX2TaZLl0W1tZKwM

7.2. Archivo Teensy 3.5

https://drive.google.com/drive/folders/1xRehF2DVIeX4N4fI2E2qjtilWtVNcnbV

7.3. Códigos y Mediciones Protocolo

https://drive.google.com/drive/folders/1qdZMVMvlh7M-BpPnptoFZA_wde-heTYc

7.4. Diseño PCB para Teensy y Shield

https://drive.google.com/drive/folders/1iKN0B47aFaDbhCJcnoXgHI0P88ngD088

Bibliografía

[1] Bluetooth.com. Understand more about bluetooth technology | bluetooth technology website. [Online].Available: https://www.bluetooth.com/what-is-bluetooth-technology., Accessed:February 2018.

[2] Certsi. Analizando bluetooth. [Online]. Available: https://www.certsi.es/blog/analizando-bluetooth, Accessed:April 2018.

[3] CSR. Csr support. [Online]. Available: https://www.csrsupport.com/document.php?did=1487, Accessed:April 2018.

[4] J. He, Z. q. Huang, Y. b. Hou, and Y. m. Dong. Point-to-multipoint stereo audio transmitting system based on bluetooth. In *2010 International Conference on Communications and Mobile Computing*, volume 3, pages 323–328, April 2010.

[5] A. Majumda, D. G. Sachs, I. V. Kozintsev, K. Ramchandran, and M. M. Yeung. Multicast and unicast real-time video streaming over wireless lans. *IEEE Transactions on Circuits and Systems for Video Technology*, 12(6):524–534, Jun 2002.

[6] Jonny McClintock. Can bluetooth ever replace the wire? In *Audio Engineering Society Convention 140*, May 2016.

[7] Technical Committee on Sensor Technology (TC-9). *IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*. IEEE, July 2008.

[8] S. Pinkumphi and A. Phonphoem. Real-time audio multicasting on bluetooth network. In *2009 6th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology*, volume 02, pages 992–995, May 2009.

[9] K.M Rabii. Providing precision timing protocol (ptp) timing and clock synchronization for wireless multimedia devices, 2017.

[10] Sierra Wireless. *Melody Audio User Guide*, May 2017.

[11] Techopedia.com. "what is audio streaming? - definition from techopedia". [Online]. Available: https://www.techopedia.com/definition/6067/audio-streaming, Accessed:March 2018.