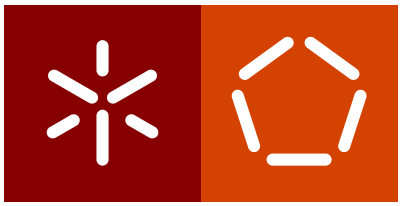
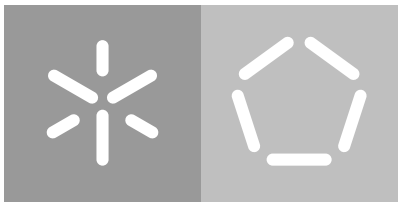


Universidade do Minho
Escola de Engenharia
Departamento de Informática

Oleksii Gylytskyy

Distribuição e Sincronização de Áudio Digital em Redes Sem Fios

June 2018



Universidade do Minho
Escola de Engenharia
Departamento de Informática

Oleksii Gylytskyy

Distribuição e Sincronização de Áudio Digital em Redes Sem Fios

Masters dissertation
Masters Degree in Informatics Engineering

Dissertation supervised by
Bruno Alexandre Fernandes Dias

June 2018

AGRADECIMENTOS

Em primeiro lugar, quero agradecer ao meu orientador Bruno Dias pelo apoio, paciência, atenção e disponibilidade em todas as etapas da dissertação.

Uma palavra de apreço especial para os meus amigos, Vasco Monteiro, André Sampaio e Ricardo Ramalho, que me ajudaram imenso em aspetos técnicos e, também, de escrita, especialmente ao Vasco Monteiro, que me ajudou a melhorar na língua Portuguesa.

Não podia de deixar de agradecer à Helena Dias pela sua simpatia, paciência e também pela ajuda em todo o meu percurso do Mestrado.

Finalmente, quero agradecer à minha família, cujo apoio não tem preço e que sem este nada disto tinha sido possível.

ABSTRACT

Nowadays, more and more, consumers look for practical and comfortable solutions for their everyday lives problems. Wireless technologies are, gradually, surpassing and replacing wired ones as they become faster, more efficient and reliable. The market, however, is full of solutions that, despite fulfilling most of users' needs, are proprietary versions and, therefore, need licensing, which holds back the evolution and development of these kind of systems. Another problem is that the majority of these solutions only address distribution of non-synchronized audio channels.

This document reports the investigation done on open-source, universal technologies that can ease the development of an alternative approach to audio distribution and synchronization. It discusses the application mechanisms developed for distribution and synchronization of digital audio channels on top of common wireless local area networks. It also presents the results of the tests made with the implemented prototype system.

Thus, the project's main goal was to prove that it is possible to achieve an optimal synchronization of independent distribution of individual audio channels over widely used IEEE 802.11 wireless local area networks.

The developed prototype achieved very good audio channel synchronization across all the system's speaker devices, even under stress tests on overloaded networks. This results in no perceptible deterioration (to human ear) of the audible experience for the end user as compared with a full cabled, digital or traditionally analog, audio system. This means the project's goal have been achieved and the algorithms could allow the development of new simpler and cheaper commercial solutions, with equal or improved quality when compared with the majority of paid solutions present on today's market.

RESUMO

Como em todas as áreas tecnológicas, também os utilizadores de sistemas áudio procuram soluções o mais práticas, convenientes e eficazes possível. As tecnologias sem fios têm vindo, gradualmente, a substituir as soluções com fios (ou cabladas) nos sistemas de distribuição de áudio, tanto , digital como analógico. O mercado está repleto de soluções que cumprem este propósito em variados graus de usabilidade e capacidades técnicas, sendo que o maior desafio é colocado pelo problema da sincronização na distribuição individual e independente de canais áudio no formato digital. As soluções atuais são quase exclusivamente versões comerciais, utilizando algoritmos e tecnologias proprietárias relativamente complexas e, como tal, não gratuitas. Este aspeto tende a atrasar o desenvolvimento de novos sistemas.

Esta dissertação documenta o projeto que teve como principal objetivo provar que é possível construir um sistema de distribuição independente e simultânea de canais áudio digitais, alcançando níveis ótimos de sincronização usando algoritmos e mecanismos alternativos sobre tecnologias bem conhecidas de rede local sem fios.

Assim, este relatório apresenta a investigação elaborada sobre tecnologias universais e gratuitas, que facilitem o desenvolvimento de um novo mecanismo de livre acesso para a distribuição digital de áudio sobre redes locais IEEE 802.11. Esse mecanismo e os algoritmos associados são discutidos em detalhe, sendo depois apresentados os passos da implementação e teste desse mecanismo num sistema protótipo.

O protótipo desenvolvido oferece um nível excelente de sincronização dos canais áudio em todos os dispositivos reprodutores do sistema, mesmo quando sujeito a condições de stress numa rede sobrecarregada. A eficácia do sistema foi medida analiticamente e não foi detetada nenhuma deterioração significativa na sincronização dos canais áudio nos sistemas reprodutores finais. Também nos testes empíricos de audição os utilizadores não conseguiram detetar qualquer nível de dessincronização.

Espera-se que a tecnologia desenvolvida, implementada e testada, permita o aparecimento futuro de novas soluções gratuitas, ou comercialmente mais competitivas, mas que poderão rivalizar com as soluções atuais em termos das capacidades técnicas de distribuição de áudio digital, sobre redes locais de utilização genérica e comum.

GLOSSÁRIO

Jitter Variação entre os tempos de Round Trip Time.

Linux Um sistema operativo baseada em UNIX.
Lossless Compressão sem perda de dados.

Master A entidade controladora do sistema.

Offset Diferença entre os tempos dos relógios dos dispositivos.
Open-source Código aberto, em português, é um modelo de desenvolvimento que promovo livre licenciamento para o design, arquitectura ou funcionalidades de um produto, neste caso de software, dando a possibilidade para que qualquer pessoa possa consultar e transformar podendo ajudar a evoluir esse mesmo software.

Point-to-Point Ponto a ponto.

Router Termo em inglês para encaminhador.

Slave Uma entidade secundária do sistema, que espera ordens do controlador para executar processos ou ações.
Smartphone Palavra que não carece de grande explicação, mas representa o termo adoptado para situações em que os portugueses se referem a "telefones inteligentes" que ainda não tem termo correspondente em português.
Streaming A transmissão contínua.

Tablet Embora não precise de tradução por ser o termo utilizado para estes dispositivos em Portugal, esta palavras refere a um dispositivo pessoal em formato de prancheta que pode ser usado para acesso à Internet, organização pessoal, visualização de fotos, vídeos, leitura de livros, jornais e revistas e para entretenimento com jogos e que apresenta uma tela sensível ao toque.
Time Skew Diferença entre a frequência de relógio de dois dispositivos.

Wi-fi Tecnologias sem fios.

ACRÓNIMOS

BMC	Best Master Clock.
DCCP	Datagram Congestion Control Protocol.
DLNA	Digital Living Network Alliance.
FTSP	Flooding Time Synchronization Protocol.
GPIO	General-purpose input/output.
HTTP	HyperText Transfer Protocol.
IEEE	Institute of Electrical and Electronics Engineers.
IP	Internet Protocol.
LAN	Local Area Network.
MAC	Media Access Control.
MP3	Um tipo de arquivo com compressão de áudio com perdas.
NTP	Network Time Protocol.
PHY	Physical layer.
POSIX	Portable operating system interface.
PPM	Parts Per Million.
PTP	Precision Time Protocol.
QoS	Quality Of Service.
RBS	Reference-Broadcast Synchronization.
RFC	Request for Comments.
RPi	RaspberryPi.

RTP	Real-time Transport Protocol.
RTT	Round Trip Time.
SCTP	Stream Control Transmission Protocol.
TCP	Transmission Control Protocol.
TPSN	Timing-sync Protocol for Sensor Networks.
UDP	User Datagram Protocol.
WISA	Wireless Speaker and Audio Association.
WLAN	Wireless Local Area Network.
WPAN	Wireless Personal Area Network.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

1

INTRODUÇÃO

Atualmente, cada vez mais compradores estão a dar preferência a equipamentos que, apesar de terem um custo elevado, sejam mais confortáveis e de fácil portabilidade. A maior procura verifica-se para equipamentos baseados em tecnologias sem fio. Com o aparecimento e evolução da tecnologia de redes locais sem fios, os custos de desenvolvimento e comercialização destas soluções foram reduzidos significativamente, o que resultou num aumento da procura e no maior investimento no desenvolvimento desta tecnologia. Hoje em dia podem ser encontrados pontos de acesso Wi-Fi IEEE 802.11a/b/c/g/n em quase todas as superfícies comerciais, edifícios e espaços públicos e residencais, proporcionando o acesso à Internet a telemóveis, computadores, televisores e outros dispositivos, assim como para o intercâmbio de informações digitais entre esses mesmos dispositivos.

As soluções sem fios têm vindo a substituir, gradualmente, as soluções com fios e poderá chegar a um ponto onde, no futuro, poderemos ser capazes de transmitir eletricidade remotamente, para todas as casas, usando a tecnologia point-to-point.

A transmissão de um sinal analógico usando tecnologias sem fios tem algumas limitações devido à interferência eletromagnética. Com o surgir de sistemas de armazenamento e processamento de dados digitais, a transmissão de dados passou a ser realizada quase exclusivamente em formato digital. A fim de acabar completamente com a necessidade de existirem ligações com fios entre equipamentos digitais, os fabricantes têm investido e impulsionado o desenvolvimento de tecnologias que operam sobre redes de comunicação sem fios.

Ainda que os sistemas de distribuição de canais áudio topo de gama, que reproduzem som de qualidade excecional, autêntico e sem perdas, utilizem, para transmitir o áudio, um sinal analógico a partir da fonte até ao dispositivo de reprodução, a indústria oferece atualmente uma ampla gama de sistemas de áudio digital sem fios que, na sua maioria, usam protocolos privados para a transferência de dados e algoritmos de sincronização entre os equipamentos finais, e portanto, usam uma frequência e algoritmos exclusivos para a transmissão e sincronização de canais de áudio; isto limita consideravelmente o desenvolvimento do mercado, não sendo possível utilizar equipamentos de rede genéricos com estes sistemas de áudio.

Assim surge a oportunidade de desenvolver um sistema que seja capaz de atingir os resultados dos sistemas digitais produzidos pela indústria mas usando protocolos abertos e tecnologias universais e abertas.

Com este projeto de dissertação espera-se desenvolver um algoritmo de sincronização de canais de áudio digital numa rede sem fios genérica e de uso comum, bem como o desenvolvimento e teste dum protótipo que permita provar que é possível investir no desenvolvimento deste tipo de sistemas com tecnologias de livre utilização e, como tal, rivalizar com as atuais alternativas pagas presentes no mercado.

1.1 OBJETIVOS

O principal objetivo deste trabalho é desenvolver um algoritmo que seja capaz de sincronizar a reprodução individual de vários canais de áudio digital em diferentes dispositivos.

A prossecução deste objetivo enfrenta dois problemas fundamentais: (i) como iniciar a reprodução síncrona de todos os canais em todos os dispositivos de reprodução áudio e (ii) como manter esta reprodução sincronizada ao longo de todo o tempo de reprodução do sinal áudio.

Genericamente, é necessário desenvolver um sistema que seja capaz de separar dois ou mais canais duma fonte áudio digital e depois transmiti-los da fonte para os sistemas reprodutores finais recorrendo a comuns redes locais sem fios Wi-Fi. Por fim, os sistemas reprodutores finais devem ser capazes de reproduzi-los de forma autónoma e síncrona durante um qualquer período de duração do sinal áudio.

Para cumprir estes objetivos é necessário:

- Investigar os requisitos mínimos relativos à sincronização dos canais de áudio considerando como padrão os limites da perceção auditiva humana;
- Perceber o que causa a dessincronização do áudio na altura da reprodução;
- Estudar os protocolos e algoritmos de sincronização de áudio, de transporte e de comunicação em redes sem fios, analisar as suas vantagens e desvantagens e perceber a sua viabilidade para eventual utilização no projeto;
- Fazer uma síntese alargada dos trabalhos existentes e relacionados na área;
- Identificar e justificar quais são os protocolos e mecanismos mais adequados para satisfazer os requisitos do presente projeto;
- Desenvolver um mecanismo de sincronização na reprodução de canais de áudio digital sobre redes locais sem fios; testar o mecanismo numa implementação protótipo;

- Completar a implementação dum protótipo completo com o desenvolvimento dum sistema cliente-servidor para a distribuição do áudio desde a fonte até aos dispositivos reprodutores finais.

1.2 ESTRUTURA DO DOCUMENTO

O documento é constituído por seis capítulos, organizados da seguinte forma:

- **Introdução**
  O primeiro capítulo apresenta a contextualização e motivações do projeto e os problemas previstos para a realização do mesmo. Este capítulo contém, também, uma descrição dos principais objetivos do trabalho e termina com uma breve descrição da estrutura do presente documento;

- **Estado da arte**
  No segundo capítulo são apresentados os principais resultados da investigação sobre tecnologias, protocolos e algoritmos que tornam este projeto possível, assim como de trabalhos relacionados que possam, de alguma forma, inspirar ou ajudar no processo de elaboração de alguns passos deste trabalho;

- **Solução Proposta**
  Este capítulo apresenta a solução proposta para o algoritmo de sincronização de áudio e a arquitetura para o sistema protótipo é introduzida; é discutida a escolha dos protocolos de transmissão e sincronização temporal que são a base do mecanismo proposto;

- **Implementação e Testes**
  Neste capítulo é detalhada a arquitetura dum sistema completo de distribuição e reprodução áudio e são discutidas as implementações do algoritmo e do protótipo desenvolvido com base nesse algoritmo, tendo em conta as escolhas para os protocolos justificadas no capítulo anterior; são também reportados os resultados dos testes realizados e apresentadas algumas conclusões sumárias sobre o desempenho do mecanismo proposto e do protótipo desenvolvido;

- **Conclusões e trabalho futuro**
  Por último serão apresentadas as conclusões deste trabalho, assim como possíveis melhorias e desenvolvimentos a considerar em trabalhos futuros.

2

ESTADO DA ARTE

Neste capítulo vão ser apresentados alguns sistemas de áudio sem fios. Vão ser também analisadas as vantagens dos protocolos de sincronização temporal existentes e vai ser efetuada uma comparação das tecnologias básicas das redes locais de comunicação sem fios, assim como apresentado um estudo sobre os protocolos de nível de transporte. Além disso, vão ser analisados os limites humanos no contexto da percepção de áudio.

2.1 AUDIÇÃO BINAURAL

A audição binaural [19] é a perceção de sons com a ajuda de ambos ouvidos ou partes simétricas (direita e esquerda) do sistema auditivo, que permite localizar a fonte de som no espaço devido à deteção de diferenças nas características principais dos sinais. As principais propriedades da audição a ter em conta são a localização espacial, efeito de precedência, sondagem binaural de loudness, fusão binaural de sons na determinação da altura, efeitos do ouvido "direito" e "esquerdo" na percepção da fala, de música, etc.

2.1.1 *Fusão binaural de sons*

O mundo é percebido por dois ouvidos. Os sons que chegam não são idênticos devido a diferenças temporais, de intensidade e de espectro. Embora possam ter características díspares, estes são fundidos num único som projetando uma única imagem. Esse processo é chamado de fusão binaural. O sistema auditivo faz a fusão binaural, ao longo do tempo em que é alimentado por sons, os quais possuem semelhanças, para ambos os ouvidos, sendo, no entanto, não fundíveis os sons que são completamente diferentes.

O mais importante para a fusão binaural são os sons com uma frequência inferior a 1500 Hz. Os sons de alta frequência, mas com frequências diferentes, que forem recebidos através dos auriculares do ouvido percebem-se como sinais separados, mas se esses sinais são modulados por um som de baixa frequência os sinais são fundidos numa única imagem auditiva. O mecanismo de fusão binaural de sons é descrito na forma de um modelo matemático que se baseia na procura pelo sistema nervoso auditivo central da correlação

cruzada entre sinais sonoros em ambos os ouvidos. Por outras palavras, os sons que entram nos ouvidos são tratados como eventos estatísticos e o mecanismo de fusão binaural usa uma procura comum entre eles. Este processo permite distinguir os componentes periódicos dos sinais do ruído, o que é importante para expandir o alcance dinâmico dos sinais de som percebidos durante a audição binaural.

Esta propriedade do sistema auditivo binaural é de grande importância para a avaliação da acústica da sala. O ouvinte percebe o som direto da fonte de sinal (cantor, músico, conferencista, etc.) e sons refletidos das paredes da sala. Os sons refletidos chegarão aos seus ouvidos mais tarde e terão uma direção diferente do que a do som direto. A fonte de som neste caso está localizada na direção do som direto e não refletida. Os sinais refletidos podem ser percebidos como sinais repetitivos distintos - um eco, ao passo que o nível de perceção consciente depende do tempo de atraso. A presença de sinais de eco na sala tem um impacto negativo sobre a qualidade do som da música e a inteligibilidade da fala. Se o intervalo entre dois sinais é muito curto (menos de 1 ms), o efeito de precedência não é notado, o que resulta num compromisso (média) na localização da origem sonora. Esse efeito é chamado de localização total. Se o intervalo é maior do que 5 ms entre pulsos, ou maior que 50 ms para a fala e música, o som direto e o eco podem ser reconhecidos separadamente, ou seja, o efeito de precedência já é evidente, sobretudo quando os sinais (o original e o refletido) têm espetros semelhantes.

2.1.2 *Efeito da precedência (efeito Haas)*

O efeito da precedência, também chamado efeito Haas, reside no facto de que, dentro dum determinado intervalo de tempo, o sinal áudio recebido anteriormente (a frente da onda sonora) domina a perceção auditiva sobre os sons recebidos mais tarde (eco).

Considere-se, por exemplo, uma situação em que dois altifalantes reproduzem o mesmo sinal no mesmo volume sonoro (Figura 2.1). Se o ouvinte estiver a uma certa distância na linha do meio, então, neste caso, o som vem de uma fonte imaginária localizada entre eles. No entanto, se inserir um atraso até 10 ms num dos altifalantes, o som começar-se-á a mover para o outro altifalante. Se aumentar o atraso para um intervalo de 10 a 30 ms, o som parecerá sair apenas do segundo altifalante. O som do altifalante atrasado será suprimido pelo cérebro, embora o sistema auditivo o continue a ouvir. No entanto, o som proveniente deste altifalante modifica a sensação de volume do som ouvido. Para que o ouvinte possa notar a fonte do sinal retardado (no intervalo de atrasos de 1-30 ms) seria necessário aumentar o nível em 10-12 dB no primeiro altifalante. Ou seja, o sinal atrasado deve ser duas vezes mais alto que o sinal no outro altifalante.

Quando o atraso começa a exceder 30 ms o ouvinte percebe gradualmente a presença de uma segunda imagem, cuja posição coincide com a posição do altifalante atrasado, mas o

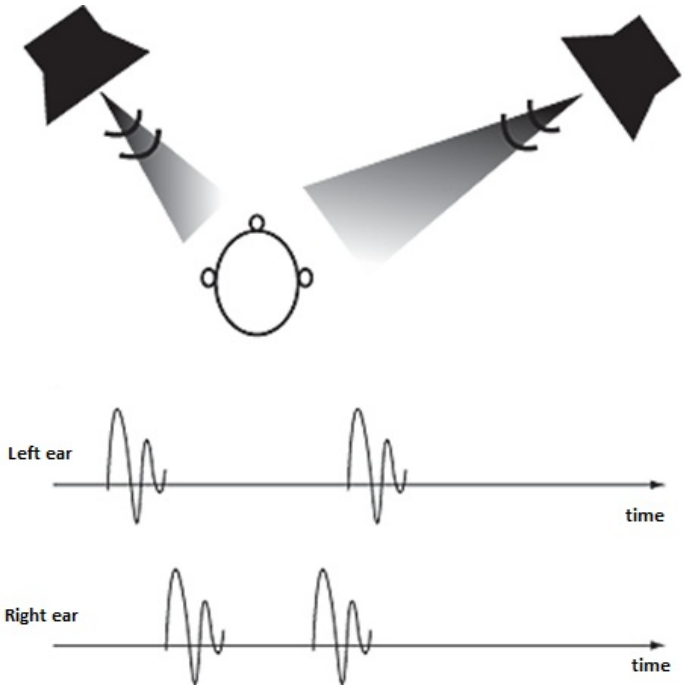


Figura 2.1: Efeito da precedência[19]

sinal do primeiro altifalante continua a dominar. Após 50 ms (dependendo da natureza do sinal - fala, música, etc.), o som do sistema atrasado é percebido como um eco.

Assim, o nosso aparelho auditivo usa diferentes mecanismos de processamento de sinais de áudio que nos permite determinar e localizar a posição da fonte de som no espaço tridimensional. É essa habilidade que é utilizada na criação de sistemas modernos de modelagem computacional de espaços de som tridimensionais e que também é usada em sistemas modernos de reprodução de som espacial.

2.2 SINCRONIZAÇÃO DE FLUXOS COM CONTEÚDO MÉDIA

A sincronização em tempo real de fluxos com conteúdo média é uma tarefa muito importante e que já existe desde os anos oitenta. Atualmente existem três tipos de sincronização: *Intra-Stream*, *Inter-Stream* e *Inter-Destination*.

A sincronização Intra-Stream consiste numa sincronização feita sobre um só fluxo de média, isto é, cada unidade que constitui o fluxo deve ser reproduzida numa dada ordem sem quaisquer lacunas e sobreposições. Como tal, o mesmo fluxo deve ser reconstruido do lado do reprodutor, mantendo a ordem e as relações temporais entre as unidades que o constituem. Esta sincronização pode ser quebrada se a distribuição do áudio digital for feita numa rede de comunicação que não garante que os pacotes cheguem por ordem e sem atraso.

A sincronização Inter-Stream descreve uma sincronização com base em dois fluxos diferentes mas logicamente relacionados. Por exemplo, o som de um vídeo ter que ser reproduzido de acordo, ou sincronizado, com a imagem.

O terceiro tipo, sincronização Inter-Destination, consiste num fluxo ter que ser reproduzido simultaneamente em vários destinos. Este terceiro conceito de sincronização é o desafio de maior complexidade. Os limites máximos possíveis de dessincronia são muito pequenos e pode ser díficil manter a sincronização na distribuição de áudio digital em comuns redes locais com simples protocolos de transporte...

O presente trabalho foca-se nos dois tipos de sincronização Intra-stream e Inter-Destination.

2.3 DESSINCRONIZAÇÃO TEMPORAL

Toda a tecnologia moderna, incluindo a tecnologia informática, para sincronizar vários processos, relógios eletrónicos, temporizadores e outros, usa um gerador de tempo. Ele gera impulsos elétricos (geralmente de forma retangular) de uma determinada frequência, que tipicamente é usada como uma frequência de referência - contando o número de pulsos é possível, por exemplo, medir intervalos de tempo. Para uma fonte de relógio do sistema operativo é necessária uma fonte de referência de precisão confiável por isso é usado um oscilador de quartzo, que é a melhor opção atualmente. No sistema clássico de um oscilador de cristal costuma ser é usado um ressonador de quartzo integrado para estabilizar a sua frequência. Os osciladores de quartzo geralmente operam numa única frequência, possuem um único sinal de saída ou um par diferencial complementar. No entanto, até o momento, a precisão que os ressonadores de quartzo fornecem não é suficiente para as tarefas de sincronização de áudio atuais [2].

2.3.1 *Oscilador de cristal, skew e offset*

A frequência é completamente determinada pelo número de vibrações por segundo, medida com algum erro. É pouco provável (ou mesmo impossível) que os mesmos osciladores de cristal em diferentes nós do sistema funcionem de forma sincronizada, mesmo que sejam completamente iguais. Nos nós do sistema existem dois tipos de erros temporais: a diferença nas frequências operacionais reais (ou o deslocamento do tempo) do relógio - *skew* - e a diferença de tempo entre os relógios locais em qualquer momento - *offset*. A variação do último depende diretamente do primeiro.

O skew depende da estabilidade e precisão do ressonador de quartzo. Por precisão de um ressonador de quartzo entende-se o grau de correspondência do divisor com sua frequência de saída nominal de quartzo, ou seja, a quantidade mínima de tempo que o

relógio conta. A estabilidade refere-se ao grau de constância da frequência do relógio de referência e pode ser de dois tipos: de curto prazo e de longo prazo. A instabilidade, ou a deriva, a curto prazo é mais frequentemente causada por mudanças nas condições ambientais (temperatura, pressão, humidade, mudanças mecânicas, etc.). A instabilidade a longo prazo está associada ao efeito de envelhecimento do ressonador de relógio de quartzo.

Devido a estes efeitos, os mesmos processos de software e hardware em sistemas informáticos aparentemente iguais podem demorar tempos diferentes. Em particular, a reprodução de áudio em dispositivos físicos aparentemente iguais, poderá resultar em perceções diferentes por causa da dessincronia temporal.

2.3.2 *Influência do oscilador de cristal na reprodução de áudio*

Na secção anterior foi notado que os ressonadores de quartzo têm imprecisão e influenciam o tempo de execução dos processos nos dispositivos informáticos. A distribuição de áudio digital e sua posterior reprodução em dispositivos transdutores diferentes, interligados por redes locais, pode ocorrer em momentos temporais diferentes e a um ritmo diferente, ainda que a reprodução seja iniciada no mesmo instante temporal no sistema controlador.

Assim, é essencial o reconhecimentos que estes fenómenos, no contexto do presente projeto, em que dois ou mais canais do mesmo áudio devem ser reproduzidos de forma sincronizada em dispositivos fisicamente remotos, utilizando apenas redes locais sem fios para a transmissão do áudio digital, são os principais responsáveis pela dessincronização não desejada.

Para o estudo e análise laboratorial do problema foi decidido usar uma plataforma modesta baseada em sistemas *RaspberryPi* (RPi), a que foram ligados altifalantes, como sistemas reprodutores finais. Esta decisão foi tomada tendo em conta não só o preço consideravelmente mais baixo, quando comparado com computadores convencionais, mas também para que os algoritmos do conceito proposto pudesse ser provado sem ter que se recorrer a sistemas de grande performace, ou seja, a eficiência dos mecanismos não deve depender relevantemente da performance dos sistemas onde é implementado.

Antes de mais, foi indispensável descobrir como o ressonador de quartzo do gerador de frequência destes dispositivos influenciaria a sincronia dos canais de áudio. Isso pode ser feito por uma medição das ondas sinusoidais geradas pelos dispositivos RPi através de um osciloscópio ou qualquer placa de som que tenha duas ou mais entradas analógicas. Para os testes foi utilizada uma placa de som "*EDIROL UA-5*" e o software "*Multi-Instrument*" e também foi desenvolvido um programa para Linux em linguagem C, cujo único propósito é a geração de uma onda sinusoidal na saída de áudio do dispositivo. Se os geradores de

frequência nestes dispositivos RPi fossem estáveis e pudessem permitir um funcionamento sincronizado, então a posição relativa das ondas não deveria mudar ao longo do tempo, ou seja, a diferença das fases das ondas manter-se-ia constante.

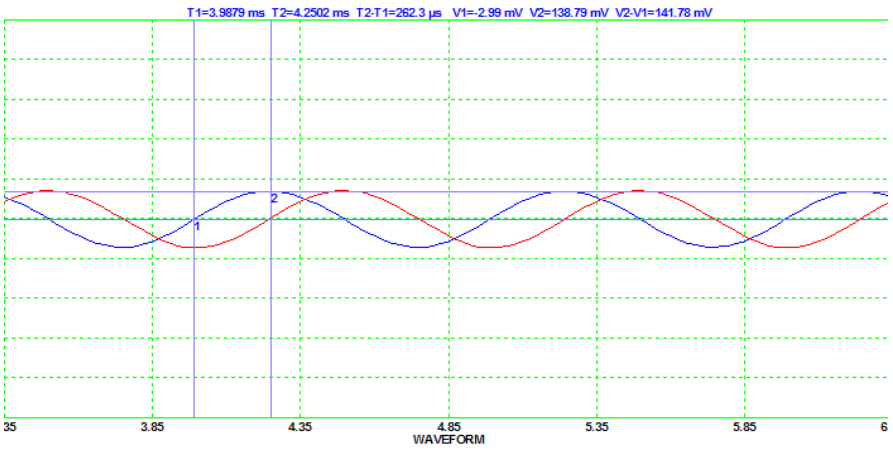


Figura 2.2: Dessincronia inicial

A Figura 2.2 é uma representação visual da interface do programa "*Multi-Instrument*" onde estão apresentadas as duas ondas sinusoidais dos dois dispositivos RPi usados para testes. É aparente uma dessincronia inicial, diferença ou deriva, das fases das ondas, que é igual a 262.3 μs. Depois dum tempo de espera de 120 segundos foi feita uma segunda medição, representada na Figura 2.3.

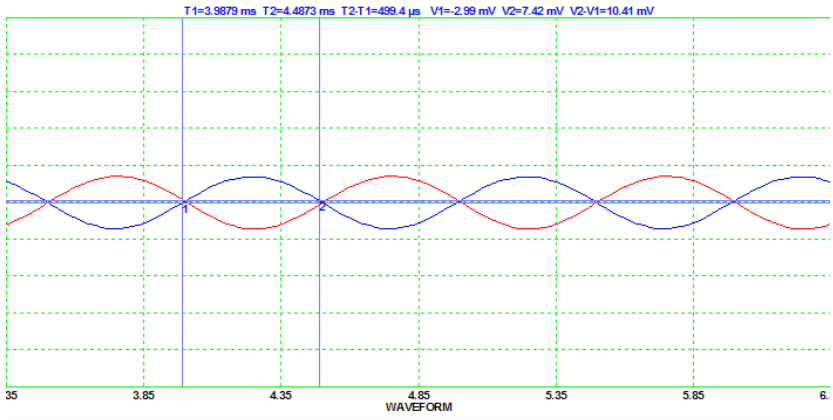


Figura 2.3: Dessincronia ao fim de 2 minutos

Pode ver-se pela Figura 2.3 que a diferença das fases das ondas aumentou para 449.4 μs. Neste cenário é fácil calcular o time skew entre os dois RPi:

$$499.4 - 262.3 = 237.1 \mu s / 2min \tag{1}$$

$$237.1/120 = 1.975\mu s/sec \tag{2}$$

Estes resultados confirmam que o áudio nos dispositivos RPi seria reproduzido a tempos diferentes, podendo originar uma dessincronia relevante, traduzindo-se, possivelmente, na perceção de eco ou mesmo em dois sons diferentes, ainda que fosse reproduzido o mesmo sinal em ambos os canais.

Sabendo que a frequência gerada também pode depender de parâmetros ambientais (temperatura, pressão atmosférica, humidade), foi necessário estudar o quanto esses fatores afetam a estabilidade da frequência de quartzo gerada. O cenário de testes foi semelhante ao anterior mas em que um dispositivo foi submetido a um arrefecimento (diminuição da temperatura ambiente) e o outro foi submetido a um aquecimento (aumento da temperatura ambiente).

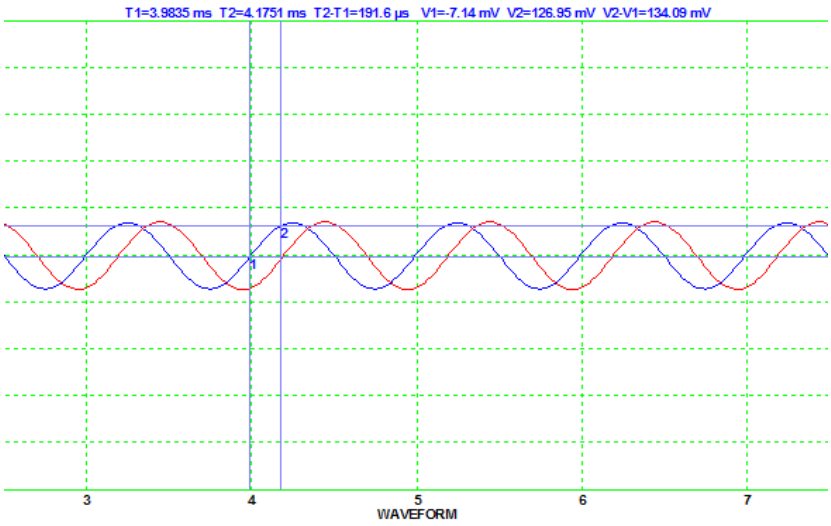


Figura 2.4: Dessincronia inicial com impacto da temperatura

As Figuras 2.4 e 2.5 ilustram os resultados obtidos com uma diferença de temperatura aproximada de 20 graus *Celsius* entre os dois sistemas. É claramente evidente que há um aumento muito relevante na diferença de fase devido à diferença de temperatura (mesmo tendo em consideração que a diferença de temperaturas provocada fosse relativamente grande):

$$(633.9 - 191.6)/60 = 7.371\mu s \tag{3}$$

Ou seja, apesar de se usarem dois dispositivos idênticos (mesmo fabricante, mesmo modelo, mesmas características, mesmas configurações de hardware e software), o áudio será reproduzido a tempos diferentes e a ritmos diferentes, ou seja, o processo de reprodução de um dispositivo não é síncrono em relação à reprodução do outro dispositivo. Mais ainda,

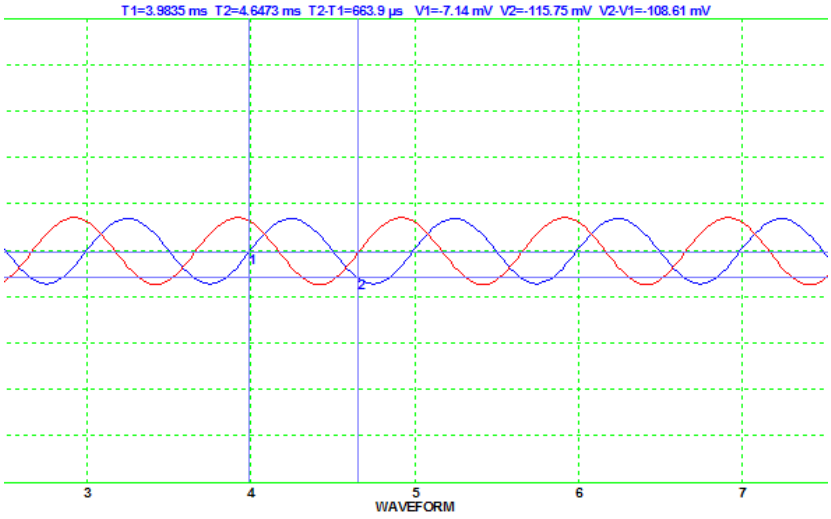


Figura 2.5: Dessincronia ao fim de 1 minuto com impacto da temperatura

o fenómeno de desssincronia terá níveis variáveis que dependerão da duração do fluxo de áudio e das diferenças nas condições ambientais a que cada sistema está sujeito.

Tendo em conta a existência desta dessincronia, e sabendo que a mesma não é estática, é fundamental descobrir uma forma de quantificar/medir o valor de dessincronia, medindo o declive/deriva temporal, para que depois este valor possa ser usado como parâmetro principal no mecanismo de compensação a definir e que sirva para manter a sincronização dos canais áudio, desde o início da reprodução e durante todo o tempo de duração do fluxo do sinal da fonte.

2.3.3 *Cálculo da dessincronização na reprodução contínua*

Para calcular quando o valor de dessincronização entre os sistemas reprodutores finais excede o limite máximo descrito na secção 2.1.2 foi realizado um conjunto de testes usando ondas sinusoidais com frequência pre-definida de período igual a 5ms. Ambos os canais foram alimentados na fonte com um sinal sinusoidal de 200 Hz. Se o valor de 50 ms na dessincronia for ultrapassado, esta tornar-se aparente para o ouvinte (tendo em conta sistemas de reprodução de elevada qualidade e ouvintes com excelentes capacidades auditivas). O teste prolongou-se até que uma onda aparecesse deslocada em 10 períodos em relação à outra. A tabela 2.1 mostra o resultado do teste que demorou cerca de 5 horas e 43 minutos até se exceder este limite máximo de dessincronia entre ondas.

Conforme pode ser confirmado pela 2.1, os canais estão completamente fora de sincronia ao fim de 20607 segundos. Se, genericamente, já existir uma dessincronia inicial, o limite máximo pode até exceder-se mais cedo.

Período	Dessincronização/ms	time/sec
1	5	2280
2	10	4545
3	15	6681
4	20	8671
5	25	10792
6	30	12681
7	35	14730
8	40	16722
9	45	18675
10	50	20607

Tabela 2.1: Dessincronia ao longo do tempo com ondas sinusoidais

Sabendo que cada pessoa tem características auditivas únicas, e pode começar ouvir o eco mais tarde do que outra, realizou-se uma experiência semelhante mas com um arquivo de audio real, em vez de se utilizar uma onda sinusoidal criada artificialmente, onde o autor deste documento tentou detetar a dessincronia verificada.

Nestes novos testes foi usado um arquivo de audio mono com som de batimento duma bateria. Como resultado dessa experiência pode constatar-se que, com algumas reproduções começou-se a ter a perceção da dessincronia (que o som tinha origem num dos dispositivos e acabava no outro) ao fim de menos de 3 horas de audição. Ao fim de 3 horas e meia já não existiam dúvidas acerca da existência de dessincronia mas só ao fim de 6 horas e meia é que houve a perceção de que se tratavam de dois sons diferentes. Os principais resultados do teste podem ser vistos na tabela 2.2.

De notar que os testes com som real foram feitos com som mono (exatamente o mesmo fluxo áudio em todos os canais) porque a dessincronia é um fenómeno que tem um efeito mais audível em fluxos de som mono do que em fluxos *stereo* para dois canais ou fluxos em que todos os canais têm fluxos áudio diferentes.

2.4 PROTOCOLOS E ALGORITMOS DE SINCRONIZAÇÃO TEMPORAL

Partindo dos problemas descritos acima é essencial que todos os processos de reprodução áudio em cada nó de todo o sistema distribuído sejam sincronizados inicialmente, isto é, que começem logo sincronizados, que começem no mesmo instante de tempo. Mesmo um pequeno desvio nos cálculos de relógio pode conduzir à dessincronia inicial desses processos, portanto, a sincronização temporal inicial temporal é uma tarefa fundamental para se garantir a sincronização desses mesmos processos ao longo de todo o tempo de duração da reprodução do fluxo de aúdio.

Tempo	Dessincronia/ms	Perceções dos ouvintes
0:38:40	4.852	Um som único
1:15:30	9.954	Um som único
1:50:30	14.761	Um som único
2:26:30	19.801	Um som único
2:48:40	23.060	Som único que começa numa coluna e acaba noutra
3:29:30	28.933	Som único que começa numa coluna e acaba noutra
4:04:30	34.057	Som único que começa numa coluna e acaba noutra
4:44:30	39.953	Som único que começa numa coluna e acaba noutra
5:14:30	44.919	Som único que começa numa coluna e acaba noutra
5:51:30	49.917	Um som de cada coluna
5:24:30	54.805	Um som de cada coluna
7:00:00	59.975	Um som de cada coluna

Tabela 2.2: Dessincronia com um arquivo áudio mono real

A forma mais óbvia de ter essa sincronia temporal no início da reprodução do fluxo áudio é garantir que, após a distribuição de parte do início do fluxo áudio do controlador para os sistemas reprodutores, estes estejam temporalmente o mais sincronizados possível por forma a que, pelo menos teoricamente, começem a reprodução áudio no mesmo instante de tempo absoluto.

Genericamente existem dois tipos principais de estratégias de sincronização temporal dos sistemas numa rede sem fios:

1. A primeira abordagem é a tentativa de sincronização dos relógios locais em relação aos outros relógios dos sistemas na mesma rede local, a espaços temporais predefinidos (ou quando um valor de deriva máximo é ultrapassado). Entre pontos de sincronização, cada sistema armazena informações sobre a sua deriva de acordo com os nós vizinhos. A maioria dos protocolos de sincronização usa esta estratégia.

2. Outro método é o de sincronização global ou absoluta, quando os sistemas da rede têm que estar constantemente sincronizados a uma escala de tempo global, absoluta e constante. Esta será a forma mais rigorosa de sincronização, mas é também a mais difícil de implementar. Poucos algoritmos de sincronização usam esta estratégia, sobretudo porque este tipo de sincronização é muito exigente em termos de recursos. Só costuma ser usado quando a utilização dos sistemas tem requisitos de tempo real em aplicações críticas.

Existem inúmeros protocolos de sincronização temporal. Nesta secção irão analisar-se os principais que são utilizados para a sincronização de dispositivos em redes de sensores sem fios - *Reference-Broadcast Synchronization*, *Timing-sync Protocol for Sensor Networks* e *Flooding*

Time Synchronization Protocol - e os protocolos que são usados para sincronização de sistemas em redes locais (com ou sem fios): *Network Time Protocol* e *Precision Time Protocol*.

Além disso, irão ser analisados algoritmos específicos de sincronização como o *Cristian's Algorithm* e o *Berkeley's Agorithm*.

2.4.1 *Reference-Broadcast Synchronization*

O Reference-Broadcast Synchronization (RBS) [9][12] é um protocolo de *broadcast* para sincronização dos sistemas recetores dos anúncios. A forma mais simples de RBS precisa dum sistema servidor de *broadcast* e dois sistemas recetores. O pacote de dados de sincronização será transmitido do servidor para os dois recetores que registam o tempo de de chegada do pacote. Depois, os dois recetores trocam essa informação temporal e calculam o seu deslocamento ou deriva. A principal vantagem deste protocolo é que elimina a incerteza no tempo de envio e de acesso 2.4.8. A principal desvantagem é que o seu comportamento deixa de ser eficaz em redes de maior escala com transições múltiplas.

2.4.2 *Timing-sync Protocol for Sensor Networks*

O Timing-sync Protocol for Sensor Networks (TPSN) [10][12] é um protocolo que utiliza uma árvore que representa a topologia da rede. O processo de sincronização é dividido em duas fases: a fase de deteção e a fase de sincronização. Na primeira fase cria-se uma topologia hierárquica de rede em que cada nó é atribuído a uma determinada camada ou nível. Apenas um nó está no nível zero, o nó raiz. Na fase de sincronização todos os nós do nível i serão sincronizados com nós do nível i-1. Isso sincronizará todos os nós dum determinado nível com o nó raiz. Tal como a fase de deteção, a fase de sincronização começa a partir do nó raiz e espalha-se pela rede. A principal vantagem do TPSN é a possibilidade de se obter uma grande precisão na sincronização nos relógios de todos os sensores. A principal desvantagem é que não é aplicável em redes de sensores com alterações dinâmicas ou frequentes na topologia.

2.4.3 *Flooding Time Synchronization Protocol*

O Flooding Time Synchronization Protocol (FTSP) [11][12] é uma versão melhorada do protocolo TPSN, permitindo a utilização em redes com topologia dinâmica. O nó raiz transmite informações de sincronização de tempo para todos os recetores participantes. A mensagem contém a marca de tempo do servidor de tempo global. Os destinatários marcam o seu tempo local quando a mensagem é recebida. Tendo o tempo de transmissão do remetente e o tempo de receção, o recetor pode estimar o deslocamento do seu relógio.

Para se obter um elevado valor de compensação da deriva do relógio, o FTSP usa regressão linear. O servidor temporal, ou nó da raiz, é selecionado dinamicamente e periodicamente, e também é responsável por manter o tempo de rede global. Os nós de receção são organizados numa estrutura de rede com topologia do tipo malha, em vez da topologia do tipo árvore, e também trocam entre si as informações temporais. O FTSP é confiável na medida em que usa *flooding messages* para resolver os problemas associados a eventuais falhas momentâneas de comunicação entre os nós, algo comum em redes de sensores, ou mudanças dinâmicas na topologia da rede. Como o TPSN, o FTSP também fornece carimbos temporais ao nível MAC, o que melhora significativamente a precisão da sincronia e reduz a sua variação.

2.4.4 *Network Time Protocol*

O Network Time Protocol (NTP) [8] utiliza um sistema hierárquico, multi-nível, de fontes de tempo. Cada nível da hierarquia é uma camada e a cada camada é atribuído um número, a começar do 0 (zero). Este número determina a distância a partir do relógio de referência e existe para evitar dependências circulares na hierarquia. É importante referir que o identificador de camada não é um indicador certo da qualidade ou fiabilidade da sincronia temporal a partir da camada 2. Ou seja, por exemplo, um sistema na camada 3 pode ter uma sincronização mais precisa e fornecer um sinal de melhor qualidade do que um sistema da camada 2. De um modo geral, as camadas servem, sobretudo, para distribuir a carga e proporcionar uma maior área de cobertura.

A camada 0 é, geralmente, um sistema especial que serve de instrumento de elevada precisão e será visto como referência ou padrão temporal, como, por exemplo, um relógio atómico (*quantum molecular*), um rádio-relógio ou instrumentos com precisão análoga. Normalmente, estes dispositivos não estão diretamente conectados à rede mas estão ligados a um sistema computacional da rede local através duma interface específica, geralmente uma porta série.

Na camada 1 temos os servidores NTP que estão diretamente ligados ao relógio de referência na camada 0. Atuam como servidores de hora da rede e respondem aos pedidos enviados pelos sistemas na camada 2. Normalmente só existe um ou dois servidores NTP por cada rede local.

A camada 2 inclui qualquer sistema ou equipamento computational que receba os dados de sincronização temporal a partir dos servidores da camada 1, utilizando o protocolo NTP. Estes sistemas podem comparar os seus dados temporais com os doutros sistemas da mesma camada de maneira a conseguirem-se valores de deriva estáveis e consistentes para toda a camada.

Os sistemas da camada 2, por sua vez, podem atuar como servidores para os sistemas da camada 3, e assim por diante.

2.4.5 *Precision Time Protocol*

O Precision Time Protocol (PTP) [14] está definido na norma IEEE 1588, cuja primeira versão foi lançada em 2002. Em 2008 o protocolo foi revisto para PTPv2. Este protocolo é extremamente flexível e pode ser usado em diversos campos de aplicação que requeiram sincronização temporal, assegurando uma precisão teórica de 10 ns em condições de operação ótimas. O PTP opera sobre o princípio de *master-slave*, onde o master fornece o tempo a um número indeterminado de slaves na rede para que estes sincronizem o seu relógio com o do master. O protocolo inclui um mecanismo que permite a configuração e segmentação da topologia de rede automaticamente: cada nó usa um algoritmo para determinar o melhor relógio num segmento. Quando o segmento não tem master disponível, ou este perde a precisão, um slave pode passar a master. Todos os masters PTP transmitem dados de sincronização para os outros nós através duma mensagem *Sync*. Com base nestas mensagens, os outros nós podem atualizar os dados temporais e, caso se justifique, proceder à sincronização ou ajuste do seu relógio. Devido à comportamento cíclico do algoritmo, os nós podem ser conectados ou removidos da rede dinamicamente.

O protocolo prevê uma sincronização precisa no modo de software puro, mas, para se alcançar a precisão máxima, é necessário usar também o mode de hardware. A sincronização com elevada precisão é conseguida através da fixação das etiquetas das mensagens de sincronização nas interfaces Ethernet, ao nível do hardware, dos controladores de interface de rede, incluindo routers e switches.

2.4.6 *Cristian's algorithm*

Este algoritmo [13] define uma abordagem simples para o processo de sincronização temporal dum sistema em relação a um servidor temporal.

Em primeiro lugar, o sistema solicita ao servidor o tempo, medindo o intervalo de tempo entre o momento de envio do pedido (T0) e o momento em que a resposta for recebida (T1). O valor do relógio pode ser atualizado com base no tempo devolvido pelo servidor e no tempo (T1-T0) / 2, assumindo que o atraso total é simétrico (pode ser medido nos dois sentidos).

O sistema vai repetindo o passo anterior até obter um nível de sincronização apropriado, tendo em consideração o tempo de convergência necessário e a máxima precisão obtível por este método. A estratégia tem os mesmos problemas que afetam os algoritmos de servidor único: o servidor pode falhar e o processo de sincronização não estará disponível.

2.4.7 *Berkeley's Algorithm*

O algoritmo de Berkeley [13] não assume que qualquer máquina tenha uma fonte de tempo exata para servir de padrão no processo de sincronização. Em vez disso, o tempo de referência é definido como o tempo médio com base no tempo dos participantes e o processo de sincronização tenta sincronizar todas os sistemas na rede com esse tempo padrão. O servidor de tempo solicita periodicamente o tempo a cada sistema da rede. Com os valores recolhidos o servidor calcula o tempo médio (incluindo o seu próprio tempo no cálculo) que passa a ser o tempo padrão. Em seguida, envia o tempo de deslocamento necessário para cada sistema fique mais sincronizado com o valor padrão. O processo é repetido ciclicamente a intervalos pre-definidos.

2.4.8 *Variância de tempos em redes sem fios*

Em todas as redes sem fios o principal problema para os protocolos de sincronização temporal é a variância elevada nos tempos de acesso, de envio, de propagação e de receção.

1. O tempo de acesso é o atraso que ocorre quando o MAC espera o acesso ao canal de transmissão. O MAC concorrente deve aguardar até que o canal fique livre antes da transmissão.
2. O tempo de envio é o tempo necessário para criar uma mensagem e colocá-la pronta no buffer de emissão.
3. Tempo de propagação é o tempo que a mensagem demora a ser enviada do buffer de saída pelo canal de comunicação.
4. Tempo de receção: é tempo que vai desde a receção da trama no interface físico até à mensagem estar disponível no buffer de entrada no sistema recetor.

A eliminação de, ou capacidade de prever com precisão, qualquer um destes tempos, aumenta significativamente a eficiência do protocolo de sincronização a definir.

2.5 PROTOCOLOS DE TRANSPORTE

Os dois protocolos de transporte mais conhecidos e utilizados no mundo para a transmissão de dados entre sistemas numa rede local são o Transmission Control Protocol (TCP) [3] e o User Datagram Protocol (UDP) [4].

2.5.1 *Transmission Control Protocol*

O TCP é um protocolo seguro com garantia de entrega dos dados. Tem um mecanismo que permite uma transferência de dados com uma conexão pré-estabelecida, e realiza o segundo pedido em caso de perda ou corrupção destes mesmos dados, elimina a duplicação na preparação das duas cópias do mesmo pacote, garantindo assim a integridade dos dados e notifica sobre os resultados de transmissão. O TCP é um protocolo complexo e com tempo de processamento relativamente grande devido ao seu mecanismo de estabelecimento de conexão e garantia de entrega dos pacotes ao destino, evitando a necessidade de implementar essa funcionalidade ao nível das aplicações.

2.5.2 *User Datagram Protocol*

O UDP é o protocolo de transporte mais rápido uma vez que ele é definido com um formato mínimo mas com tudo o que é necessário para a transmissão de dados. Naturalmente, devido a este formato reduzido, tem algumas desvantagens, entre as quais:

1. Os pacotes chegam desordenados, isto é, o primeiro pacote pode chegar mais tarde que o segundo, por exemplo;
2. A entrega de mensagens no UDP não está garantida, a mensagem pode ser perdida e podem ser obtidas duas cópias da mesma mensagem. O último caso ocorre quando são enviadas mensagens para um endereço usando duas rotas diferentes;
3. O UDP não requer estabelecimento de uma ligação, e os dados podem ser enviados imediatamente, logo que eles sejam preparados;
4. O UDP não envia mensagens de confirmação e, por isso, os dados podem ser recebidos ou perdidos. Usando o protocolo UDP, se for preciso uma transmissão confiável dos dados, a funcionalidade de reenvio dos pacotes perdidos deve ser implementada a nível aplicacional;

No final, a propagação de um protocolo depende, em grande parte, do equilíbrio entre a facilidade da sua implementação e conjunto das suas funcionalidades.

Além destes dois protocolos, existem outras alternativas que são menos conhecidas e, portanto, menos utilizadas:

2.5.3 *Datagram Congestion Control Protocol*

Datagram Congestion Control Protocol [5] é um protocolo, da camada de transporte do modelo OSI.

A sua principal função é fazer streaming multimédia em tempo real e, portanto, é um candidato interessante, embora não seja muito utilizado.

O DCCP tem mecanismos de controlo de congestão ECN [RFC3168] e REC Nonce [RFC3540] evitando a necessidade de os criar a nível aplicacional, tem também concordância garantida dos parâmetros na organização das ligações. O protocolo não garante a entrega atempada dos pacotes nem a sua ordenação, mas tem mecanismos adicionais para assegurar, com alta confiabilidade, a informação sobre a entrega dos pacotes.

2.5.4 *Stream Control Transmission Protocol*

O Stream Transmission Control Protocol [6] é um protocolo de transporte confiável, que fornece uma transmissão estável e ordenada, preservando a reordenação dos pacotes, dos dados entre dois pontos finais, como o TCP. Além disso, o protocolo garante preservação das fronteiras de mensagens individuais, como o UDP. No entanto, ao contrário do TCP e do UDP, o SCTP tem vantagens adicionais, tal como o suporte de *multihoming* (a conexão síncrona entre dois hosts em dois ou mais canais físicos independentes) e *multi-streaming* (a capacidade de transmitir dados, simultaneamente, em vários fluxos independentes que evita o fenómeno *Head-of line Blocking*), bem como proteção contra ataques DDoS por causa do 4-*way handshake*. Porém, também tem as suas lacunas como, por exemplo, a quantidade relativamente grande de encaminhamento de tráfego que é bastante superior à dos protocolos TCP / UDP. Note-se que este protocolo não é adequado para streaming de áudio.

2.5.5 *Real-time Transport Protocol*

O RTP [7] ao contrário dos protocolos UDP, TCP, SCTP e DCCP não é um protocolo da camada de transporte. Ele funciona no nível aplicacional e, portanto, utiliza os protocolos de nível de transporte para transmissão de dados, onde UDP se usa por definição, mas também pode ser executado sobre o TCP. O RTP funciona em colaboração com o RTCP(Real-Time Control Protocol), que fornece estatísticas e mecanismos de gestão, uma vez que o RTP é apenas para o transporte dos dados.

Esta família de protocolos foi projetada com o intuito de transmitir conteúdo de áudio e vídeo em tempo real. O RTP fornece informações sobre o tipo de dados transmitidos e faz a numeração e marcação de tempo de cada pacote. Deve-se ter em mente que o próprio RTP não permite entrega pontual e não oferece quaisquer garantias de do nível de qualidade de serviço QoS. Este protocolo não dá garantias sobre a ordem correta de entrega dos dados, esta ordem pode ser recuperada com a ajuda de números sequenciais nos pacotes no lado do receptor. O carimbo de tempo, de cada pacote, é usado para sincronizar os fluxos de vídeo e de áudio, e pode também servir para reordenar os pacotes no lado do receptor. É de

notar que como o RTP não garante a entrega atempada dos dados e não tem mecanismos de sincronização entre os intervenientes da troca de informação, não é indicado para o trabalho que se propõe neste documento.

2.5.6 *Tecnologias sem fios*

Até à data conhecem-se duas tecnologias para a transmissão, sem fios, de dados digitais: Wireless Personal Area Networks e Wireless Local Area Networks.

Segundo o artigo [1], onde foi feita uma comparação detalhada das capacidades de ambas as tecnologias, não é possível, por causa dos limites da velocidade e também do raio da cobertura, utilizar a tecnologia Bluetooth para o streaming de áudio estéreo de alta e média qualidade sem aplicar algoritmos de compressão dos dados.

Tendo em conta que este artigo foi publicado em 2005, e que desde então a evolução destas tecnologias não estagnou, é necessário obter uma visão atualizada das mesmas. Em 2009 foi anunciada a versão 3.0 do Bluetooth que, através da adição de AMP (Alternate MAC/PHY) ao protocolo 802.11 como conexão de alta velocidade, podia obter teoricamente velocidades máximas até 24 Mbit por segundo mas com maior consumo energético, contudo esta versão também consegue funcionar com a velocidade habitual de 1 Mbit por segundo. Na versão 4.0, no modo de alto consumo energético, foi possível atingir o dobro da largura de banda e, mais tarde, na versão 5.0 verifica-se também um novo crescimento nesta largura da banda.

Em relação à tecnologia WiFi, o padrão IEEE 802.11ac que tem a faixa de frequência de 5GHz é capaz de fornecer uma velocidade superior de 433 Mbit/sec só com uma antena.

No contexto deste projeto, pretende-se transmitir não só som estéreo, mas também som surround no formato 5.1 e 7.1, de alta-fidelidade, sem ter a necessidade de aplicar algoritmos de compressão ou se houver necessidade, utilizar um algoritmo sem perda de qualidade (*Lossless*). Tendo em consideração a velocidade máxima de transmissão dos dados e o raio da cobertura, uma possível solução passa pelo uso da tecnologia WiFi.

2.6 OUTRAS SOLUÇÕES RELACIONADAS

Como foi previamente mencionado, nesta parte do capítulo, vai ser feita um apanhado, algo detalhado, de trabalhos relacionados existentes.

2.6.1 *DLNA*

Não é uma tecnologia de áudio sem fios, mas é um conjunto de padrões [15] de rede que permite, para dispositivos compatíveis, transmitir e receber o conteúdo de dados média

(imagens, música, vídeo) através de redes sem fios, e exibi-lo em tempo real. Ou seja, a tecnologia para conectar computadores, telemóveis, eletrodomésticos ou outros dispositivos a uma rede digital. Todos os dispositivos DLNA comunicam uns com os outros através de níveis de tráfego da rede TCP / IPv4 HTTP para transferir o conteúdo média e, posto isto, ele pode trabalhar também nas redes com e sem fios. Isto facilita a compatibilidade com outros equipamentos, uma vez que o protocolo IPv4 já existe há bastante tempo.

A partir de 2011 foi adicionado a este padrão o suporte para Bluetooth.

Para deteção e auto-configuração dos dispositivos no DLNA, este utiliza o protocolo UPnP. No entanto, este protocolo só é capaz de reproduzir áudio ou vídeo, em simultâneo, num dispositivo e, por isso, é inconveniente para reproduzir música por toda a casa, por exemplo.

2.6.2 *AirPlay*

Este é um protocolo [16] desenvolvido pela Apple, cuja utilização é fornecida pelos meios de comunicação sem fio de streaming (áudio, vídeo, imagens) entre os dispositivos, ou seja, é uma tecnologia que permite transmitir áudio e vídeo a partir de dispositivos iOS para outro sistema de áudio, por exemplo, MAC ou Apple TV. AirPlay substituiu a tecnologia AirTunes da Apple que permitia enviar fluxos de áudio para a sua reprodução remota via WiFi. Ao contrário do AirTunes, o AirPlay permite a transmissão para os dispositivos compatíveis com essa tecnologia e não apenas música mas, também, fotos, vídeos e metadados (títulos de músicas, nomes de artistas, nomes de álbuns, o tempo restante do arquivo e a capa do álbum). Uma diferença importante entre o AirPlay e o AirTunes é o facto da Apple permitir a incorporação da sua tecnologia nos produtos de outros fabricantes e, assim, alcançar a interoperabilidade dos produtos desses fabricantes com produtos Apple. Como protocolo de transferência de dados de áudio, o AirPlay usa RTSP (Real Time Streaming Protocol) sobre o já bem conhecido UDP. A transferência é realizada sem qualquer perda de dados em 16 bits e 44,1 kHz, utilizando um algoritmo de compressão sem perdas Lossless. Contudo, o Airplay não suporta mais de dois canais e, como tal, não é capaz reproduzir som stereo em duas colunas diferentes, separando os canais um para cada coluna.

2.6.3 *WISA*

WISA [17] é uma associação que desenvolve o protocolo para transmissão áudio da fonte para o amplificador em redes sem fios. O protocolo prevê a interoperabilidade entre dispositivos de diferentes marcas que suportam esta especificação. A WISA utiliza o protocolo DFS (Selecção Dinâmica de Frequência) para procurar frequências livres no intervalo de 5.2Ghz a 5.8GHz. O protocolo não usa algoritmos de compressão de dados e é

capaz de transmitir áudio de 24 bits a 32, 44.1, 48 ou 96KHz. O protocolo dispõe de um sistema de detecção antecipada e recuperação de erros, bem como um máximo de 5.2 ms de dessincronização temporal.

2.6.4 *SONOS*

Mais uma empresa [18] que desenvolve hardware/software para sistemas áudio sem fios e assim como outras companhias, a SONOS não disponibiliza o seu protocolo para uso público. Para além disso, o sistema de áudio sem fios deles só trabalha com os equipamentos da SONOS. Não há nenhuma possibilidade de encontrar mais detalhes em relação ao protocolo que possa ser útil para este projeto.

2.6.5 *PulseAudio*

Outro projeto que merece atenção é o servidor de som PulseAudio de plataforma cruzada que funciona em plataformas POSIX, como Linux, Solaris e FreeBSD e distribuído sob software livre. Uma das características interessantes do servidor de áudio Pulseaudio é a transmissão do fluxo de áudio pela rede para dispositivos de áudio remotos. O PulseAudio pode receber som de uma ou mais fontes (processos ou dispositivos) e enviá-lo para um ou mais servidores de PulseAudio remotos. O Pulseaudio não possui funções como:

1. A separação de canais de áudio.
2. Não sincroniza os canais de áudio durante a reprodução nos dispositivos remotos.

2.7 OUTROS CASOS DE ESTUDO

Há também outros autores [20] [21] [22] que tentaram encontrar soluções para o problema do presente relatório.

Observando o artigo [20], vemos que os autores chegaram à conclusão de utilizar o protocolo NTP para sincronizar a hora do sistema pelas seguintes razões:

1. A precisão que o NTP oferece é suficiente para a realização da tarefa;
2. Como os autores pensaram em cenários em que os equipamentos pudessem estar a trabalhar em redes diferentes e como o PTP funciona apenas no nível L2, o NTP resolve esse problema por ser global;
3. A plataforma Windows .NET, da Microsoft, que tinham a intenção de usar, tem muito pouco suporte do PTPv2;

No que diz respeito ao protocolo para transmissão de dados, a opinião deles é que o protocolo DCCP seria o melhor candidato, devido à sua simplicidade e os mecanismos de controle de congestão, mas infelizmente eles tiveram que abandonar o uso do mesmo por causa do fraco apoio no Windows. Os autores, surpreendentemente, acabaram por escolher usar o protocolo RTP / RTCP, pelo simples facto de que o próprio protocolo pode organizar timestamps nos pacotes do fluxo de áudio, o que ajuda a garantir a reprodução ordenada/organizada.Para a sincronização, os autores optaram por uma uma abordagem, baseada em duplicação e remoção das amostras do buffer, através da utilização da framework GStreamer. Os testes de sincronização em redes sem carga mostraram bons resultados, abaixo do limite. No entanto, numa rede sobrecarregada a qualidade da sincronização baixou devido aos atrasos assimétricos.

No artigo [21], o trabalho teve como principal objetivo a obtenção dos melhores resultados, em termos de precisão, na sincronização temporal através de redes sem fios. Os autores queriam saber a precisão máxima com que pode ser sincronizada a hora do sistema através do wireless usando o protocolo PTP. Usando apenas a parte aplicacional do protocolo PTP, concluíram que o relógio pode ser sincronizado com uma precisão até 50 microssegundos, argumentando que esta precisão é suficiente para a sincronização *tightly coupled audio*.

Na dissertação [22], da autoria do estudante da Universidade do Minho Paulo Alves, foi estudado o mesmo problema, onde o autor provou que é possível, utilizando tecnologias open-source, alcançar uma dessincronia, menor que um milissegundo, na reprodução de som estereofónico em dois dispositivos distintos. Após inúmeros testes, o autor chegou, também, à conclusão que para resolver o primeiro problema, não há necessidade de sincronizar o tempo nos dispositivos, sendo suficiente saber a diferença do tempo de offset em todos os dispositivos em relação ao controlador. Conhecendo essa diferença, no início da reprodução, reduz os gastos em termos de recursos do sistema e tempo de execução. Na opinião do autor, a melhor solução será o uso do protocolo PTP, um protocolo de alta precisão, sendo este suficiente para efetuar a sincronização temporal na rede local e também capaz de fornecer os valores do offset. Para solucionar o segundo problema o autor propôs calcular o declive entre dispositivos e, uma vez conhecendo esse declive, remover ou adicionar amostras ao fluxo de áudio. Além disso a sincronização sistemática do tempo nos dispositivos exclui completamente a possibilidade de calcular o time skew. Ele provou ainda que, conhecendo apenas este time skew, é possível sincronizar a reprodução de dois canais e, também, que estas operações não afetam a qualidade da música reproduzida, porque uma amostra de 44.100Hz tem uma duração de apenas 22.6 μs.

3

SOLUÇÃO PROPOSTA

Com base nos estudos realizados e que foram sumariamente descritos no capítulo anterior, foi possível ter uma perceção inicial de quais os protocolos e tecnologias que poderiam contribuir de melhor forma para a obtenção dos objetivos do presente projeto. Este capítulo explica as escolhas feitas em relação às tecnologias de comunicação usadas e os os algoritmos de sincronização temporal necessários para se tentar garantir valores de dessincronia entre dispositivos reprodutores adequadamente pequenos e estáveis, por forma a que o efeito auditivo da dessincronia não seja percetível durante toda a reprodução dos fluxos de áudio, independentemente da sua duração.

A maior parte dos sistemas de distribuição de áudio em redes sem fios descritos no capítulo anterior, pressupõem soluções que implementam a distribuição de áudio através de protocolos e tecnologias próprios que, ou não estão disponíveis para utilização por terceiros ou carecem de processos de licenciamento não gratuitos. Além disso, os mecanismos de sincronização para fluxos de áudio para reprodução de canais individuais e independentes ou não existem (estão incluídos apenas mecanismos de distribuição simultânea de fluxos agregados) ou não foi possível conhecer os algoritmos e estratégias utilizados na sua implementação.

Como já foi mencionado, o principal objetivo deste projeto é provar que é possível desenvolver uma solução em que o som multi-canal de uma fonte pode ser distribuído e sincronizado por sistemas reprodutores remotos, um para cada canal (ou grupos de canais), utilizando apenas algoritmos relativamente simples (que exijam poucos recursos para a sua implementação) e tecnologias e protocolos de comunicação comuns e muito económicos, na maior parte dos casos, pre-existentes ao sistema de distribuição áudio. Ou seja, a solução proposta deverá ser facilmente implementada em redes locais IP sobre tecnologia de rede sem fios Wi-Fi. Na realidade, a solução deve ser suportada em qualquer tecnologia comum de redes locais, com ou sem fios, uma vez que a solução proposta deverá ser independente da tecnologia de rede de nível físico ou de ligação.

3.1 ESCOLHA DO PROTOCOLO DE SINCRONIZAÇÃO TEMPORAL

Provou-se na secção 2.3.2 que o tempo, mesmo em dois dispositivos completamente idênticos em termos de hardware e software, flui de maneira diferente, o que influencia a reprodução do áudio nos sistemas reprodutores.

Assim, os principais problemas a resolver foram:

- Como iniciar a reprodução sincronizada de dois, ou mais, canais áudio em diferentes dispositivos reprodutores, fisicamente remotos e independentes, ligados a uma mesma rede sobre uma conexão Wi-Fi;
- Como manter essa sincronização, durante todo o tempo de reprodução do fluxo de áudio nos sistemas reprodutores independentes.

Como foi sugerido em [22], foi decidido tentar solucionar o primeiro problema sem se proceder à sincronização dos relógios de todos os dispositivos do sistema. A vantagem imediata desta abordagem, diferente das soluções anteriores que suportam a sincronia do início da reprodução na sincronia dos relógios de todos os equipamentos do sistema, é evitar-se a adoção obrigatória de protocolos/serviços de sincronização temporal (como o NTP ou o PTP) e evitar-se a execução do processo contínuo destes mecanismos de sincronização, poupando-se recursos locais em cada equipamento.

Na estratégia clássica, depois dos relógios de todos os equipamentos do sistema de distribuição de áudio estarem sincronizados duma forma estável (isto é, com um valor estável de dessincronia adequadamente pequena para o objetivo em causa), o inicio do fluxo de áudio pode ser agendado pelo equipamento controlador para um momento temporal igual em todos os dispositivos reprodutores. No entanto, esse momento temporal tem que ser definido tendo em conta o tempo necessário para a prévia distribuição duma parte suficiente do fluxo de áudio (no sentido de permitir eventuais alterações dinâmicas na largura de banda da rede local e eventuais diferenças no tempo de execução no processo de preparação dos dados de áudio no buffer de reprodução nos equipamentos reprodutores) para os buffers de áudio em cada dispositivo reprodutor. Se os dispositivos reprodutores estiverem com os relógios sincronizados, garante-se assim um início sincronizado da reprodução dos fluxos de áudio em todos eles. Parte-se do princípio que o processo interno de início da reprodução dos dados que já estão preparados num buffer próprio para reprodução áudio tem diferenças negligenciáveis na duração da sua execução. Não esquecer que, nesta estratégia, a persecução da sincronia dos relógios é um processo contínuo, executado em paralelo com todos os outros mecanismos necessários para a efetiva distribuição, preparação e reprodução dos fluxos de áudio em todos os dispositivos do sistema.

Na abordagem escolhida para a solução agora proposta, o processo de sincronização dos relógios de todos os dispositivos do sistema é substituído por um mecanismo mais simples de cálculo do offset, desvio ou deriva, temporal de cada dispositivo reprodutor em relação ao equipamento controlador. Calcular este valor para cada dispositivo reprodutor é suficiente para se garantir o início sincronizado dos fluxos de áudio com precisão similar à atingida através do mecanismo clássico de sincronização de relógios. No entanto, o processo, sendo mais simples, utiliza menos recursos nos dispositivos e pode ser facilmente implementado sem se recorrer a quaisquer protocolos/serviços adicionais. Sabendo-se a deriva temporal de cada um em relação ao equipamento controlador, cada um dos dispositivos reprodutores só tem de adaptar o tempo de início da reprodução dos dados já prontos no buffer de reprodução áudio tendo em consideração esta deriva (a adaptação é uma simples adição ou subtração do valor da deriva ao tempo de início definido pelo equipamento controlador). Outra vantagem é que o valor de deriva temporal calculado servirá também como base ao algoritmo que garante a sincronização durante a reprodução dos fluxos de áudio.

Para resolver o segundo requisito fundamental de manter a sincronização estável durante a reprodução de todo o fluxo de áudio, é essencial medir as diferenças na evolução das derivas dos diferentes dispositivos reprodutores, ou *time skew*. Ou seja, as diferenças entre os *jitters* da deriva temporal de cada dispositivo vai ser utilizado no mecanismo de sincronização durante a reprodução do áudio. Mais uma vez, este mecanismo de sincronização não precisará de ajuda de protocolos adicionais de sincronização temporal.

3.2 Escolha do protocolo de transmissão

Relativamente à escolha do protocolo de transporte, e por algumas das razões já apresentadas no capítulo anterior, os dois candidatos mais adequados seriam o UDP e o RTP. Contudo, considerando que o RTP não implementa diretamente qualquer mecanismo de sincronização entre fluxos remotos de áudio, a sua utilização, sendo um protocolo mais complexo e com mais exigências em termos de recursos, foi descartada (o RTP só suporta diretamente a sincronização entre fluxos de vídeo e áudio no mesmo dispositivo reprodutor).

Refira-se que a não utilização do TCP deve-se ao facto de que a sua implementação no contexto da distribuição de fluxos áudio em redes locais IP não traz vantagens relevantes e acrescenta uma latência indesejável, quando comparada com a utilização do UDP.

No que diz respeito ao UDP, as principais vantagens na sua utilização são a sua simplicidade, baixo *overhead* e menor latência. O UDP possibilita menores atrasos no tempo de envio, no tempo de acesso, no tempo de propagação e no tempo de receção em redes locais IP sobre Wi-Fi. O facto de não suportar mecanismos de controlo de perda de pacotes, de

controlo da congestão e de controlo da ordem de chegada dos pacotes não é significativo para os requisitos dum simples protocolo de distribuição de dados áudio sem compressão em redes locais Wi-Fi. Por outro lado, o mecanismo de cálculo da deriva temporal é tanto mais preciso quanto menos latência existir na comunicação entre os dispositivos, confirmando o UDP o protocolo de transporte mais adequado neste contexto.

3.3 ARQUITETURA PROPOSTA

A arquitetura utilizada no desenvolvimento do sistema de distribuição de áudio deste projeto está representada na figura 3.1 e inclui um sistema *master* que funciona como controlador, contém os dados áudio originais e calcula o time skew de todos *slaves* (dispositivos reprodutores, neste caso sistemas Raspberry-PI). Periodicamente, o controlador determina os offsets para cada reprodutor final e com base nestes valores calcula o time skew associado.

Para a sincronização *Intra-Stream* foi escolhida a metodologia de início deferido com períodos de espera adaptados ao valor de deriva para cada dispositivo reprodutor, suportado com a utilização dum *buffer* de preparação dos dados a serem utilizados diretamente no processo final de reprodução de áudio controlado pelo sistema operativo. Além disso, este buffer permite manter uma reprodução fluida, absorvendo eventuais diferenças dinâmicas na largura de banda disponível, comum nas redes Wi-Fi.

Para a sincronização *Inter-Destination* (sincronia nos fluxos de áudio entre dispositivos reprodutores) foi escolhida a metodologia de adição e remoção de amostras [22]. Cada reprodutor final, com base no valor do time skew enviado pelo controlador, calcula a quantidade e frequência das amostras para remoção ou adição no *buffer* de dados para a reprodução áudio. A adição de amostras atrasa a reprodução e a remoção de amostras acelera a reprodução do áudio. É necessário efetuar um cálculo do número de amostras que é preciso remover a um dos dispositivos reprodutores e, ao mesmo tempo, acrescentar ao outro, com o objetivo de atingir o mesmo ritmo de reprodução do áudio em todos os dispositivos reprodutores. Como tal, de forma a atingir este balanceamento o número de amostras que tem de ser retirado de um dispositivo tem de ser igual ao número de amostras a acrescentar ao outro, idealmente fazendo a diferença da reprodução do áudio em ambos os dispositivos não seja percetível para o ouvido humano.

3.4 CÁLCULO DA DERIVA TEMPORAL

A deriva temporal (ou offset) corresponde à diferença entre os relógios dos sistemas operativos dos dispositivos reprodutores (slaves) e o dispositivo controlador (master).

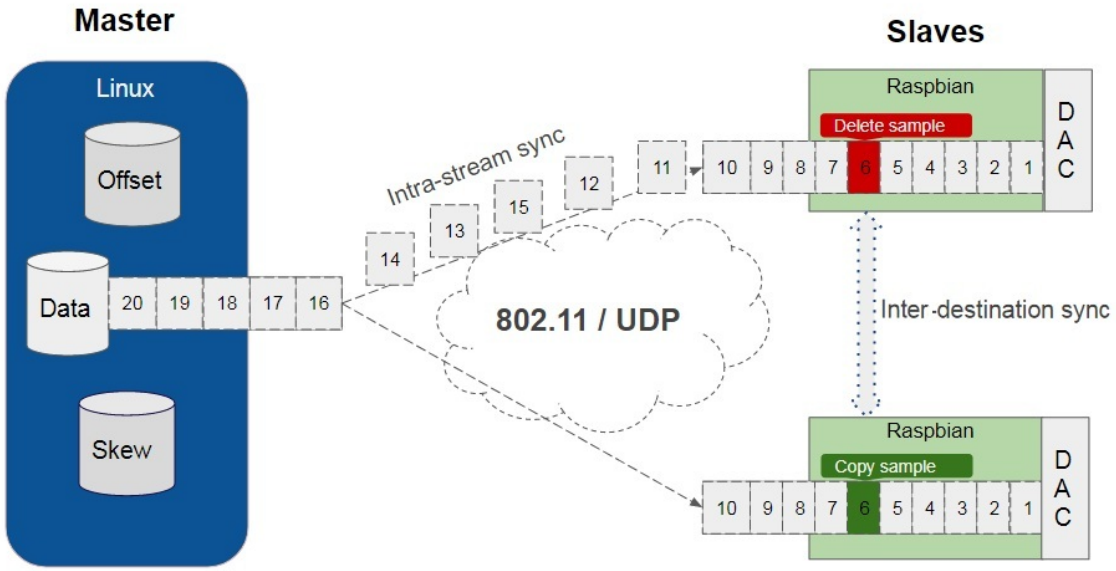


Figura 3.1: Arquitetura proposta

A Figura 3.2 tenta apresentar visualmente este conceito. t1 é um instante do tempo do relógio do sistema operativo no controlador e t2 é um instante do tempo do relógio do sistema operativo num dispositivo reprodutor. A diferença entre t1 e t2, a haver, representa o offset.

O primeiro passo no cálculo do offset é o master enviar uma mensagem para o slave com a indicação de t1. Se o tempo total de comunicação (tempo de preparação e envio no master, mais o tempo de propagação na rede, mais o tempo de receção e processamento da mensagem no slave) não fosse relevante, então o offset seria apenas a diferença entre t2 e t1.

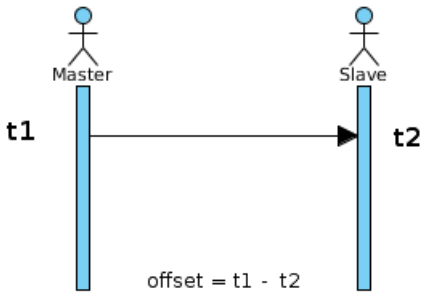


Figura 3.2: Cálculo do offset

Sabe-se, no entanto, que cada pacote na rede tem um tempo de entrega que inclui um atraso descrito e, como tal, quando o slave recebe a mensagem esse atraso tem que ser considerado. Ou seja, ao calcular o offset é necessário fazer uma correção e subtrair o atraso total (*delay*) ao tempo t2 (ver Figura 3.3).

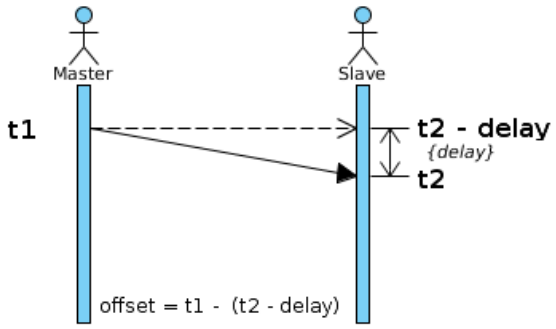


Figura 3.3: Cálculo do offset considerando o delay

O slave só poderia calcular o valor do atraso se o tempo fosse exatamente igual nos dois dispositivos mas como os relógios não estão sincronizados o slave não consegue estimar a duração da transmissão do pacote. Por este motivo, a correção deve ser feita do lado do master onde é possível estimar o *Round Trip Time* (RTT). Partindo do princípio de que o tempo de ida para o slave (master -> slave) e o tempo de volta para o master (slave -> master) é aproximadamente o mesmo, então o valor de correção pode ser estimado a partir da Fórmula 4, visualmente representado na Figura 3.4.

No entanto, esta assunção é consideravelmente perigosa e passível de erro uma vez que o delay nos dois sentidos, em redes Wi-Fi, pode não ser o mesmo. Este erro é considerado na Fórmula 4 pela introdução dum ΔT, valor que representa um intervalo para o erro expectável na diferença de estimativa do delay nos dois sentidos.

$$CorrectionValue = (t3 - t1)/2 \pm \Delta T \tag{4}$$

Partindo destas ideias base, o cálculo do offset pode ser feito em 4 passos:

1. **Master**: Determina tempo t1 no momento do envio de uma mensagem de solicitação de tempo do sistema ao slave;
2. **Slave**: Ao receber o pedido determina tempo t2 e responde ao master;
3. **Master**: Ao receber a resposta do slave determina tempo t3;
4. **Master**: Calcula o offset, pela Fórmula 5.

$$offset = t1 - (t2 - ((t3 - t1)/2 \pm \Delta T)) \tag{5}$$

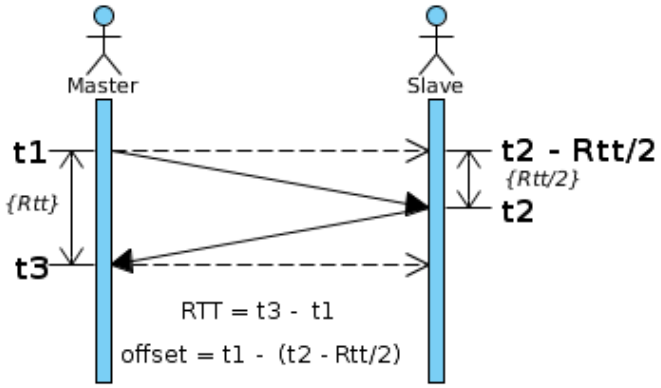


Figura 3.4: Cálculo do offset com base no valor do RTT

Apesar de tudo, esta abordagem do cálculo do offset pode não conseguir obter valores precisos por causa de variações consideráveis no ΔT, devido à variação do RTT, ou jitter. Além disso, valores imprecisos de offset podem, por sua vez, levar a erros consideráveis nos valores do time skew.

O valor de jitter na rede pode ser verificado através duma simples execução do comando ping, conforme exemplo da Figura 3.5.

```
64 bytes from 192.168.1.200: icmp_seq=8 ttl=64 time=10.7 ms
64 bytes from 192.168.1.200: icmp_seq=9 ttl=64 time=66.7 ms
64 bytes from 192.168.1.200: icmp_seq=10 ttl=64 time=8.57 ms
64 bytes from 192.168.1.200: icmp_seq=11 ttl=64 time=8.67 ms
64 bytes from 192.168.1.200: icmp_seq=12 ttl=64 time=12.5 ms
64 bytes from 192.168.1.200: icmp_seq=13 ttl=64 time=10.3 ms
64 bytes from 192.168.1.200: icmp_seq=14 ttl=64 time=9.05 ms
64 bytes from 192.168.1.200: icmp_seq=15 ttl=64 time=111 ms
64 bytes from 192.168.1.200: icmp_seq=16 ttl=64 time=9.05 ms
64 bytes from 192.168.1.200: icmp_seq=17 ttl=64 time=43.2 ms
64 bytes from 192.168.1.200: icmp_seq=18 ttl=64 time=9.00 ms
64 bytes from 192.168.1.200: icmp_seq=19 ttl=64 time=8.95 ms
64 bytes from 192.168.1.200: icmp_seq=20 ttl=64 time=6.49 ms
64 bytes from 192.168.1.200: icmp_seq=21 ttl=64 time=13.0 ms
64 bytes from 192.168.1.200: icmp_seq=22 ttl=64 time=53.7 ms
64 bytes from 192.168.1.200: icmp_seq=23 ttl=64 time=12.3 ms
```

Figura 3.5: O jitter na rede

Este teste mostra que o RTT varia muito e também não se sabe onde ocorreu a maior diferença no atraso, se na ida ou na volta. Para contornar este problema é necessário fazer uma filtragem dos valores do RTT, descartando todos os pacotes cujo atraso não pertença a um determinado intervalo de confiança, ou seja, pacotes que apresentem atrasos demasiadamente grandes ou pequenos.

3.4.1 *Intervalo de confiança*

Como foi referido anteriormente, é necessário determinar um intervalo de confiança para os valores de RTT. Para isso, é essencial construir um gráfico de densidades de probabilidade de possíveis valores para o RTT, para perceber a variância obtida e também se é possível distinguir, entre esses valores, este intervalo. Com o objetivo de recolher dados foi escrito um programa em C que usa o protocolo UDP para os pedidos de eco.

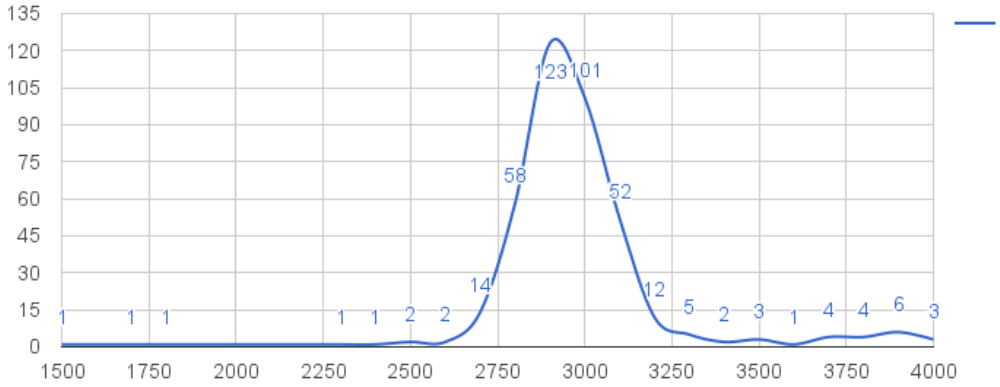


Figura 3.6: Histograma com os valores do RTT

Como se pode ver na Figura 3.6, que é o resultado de 500 iterações, o gráfico obedece à lei da distribuição de Haus, onde o centro da distribuição coincide com o seu máximo e será o verdadeiro valor da quantidade medida, por isso, partindo da regra empírica, que é apresentada na Figura 3.7, 98% dos valores situam-se na faixa da regra empírica, 68% dos valores estão no intervalo de mais ou menos um sigma, do topo do gráfico. Como se pode ver na Figura 3.6, com base em 500 iterações, 44,4% dos valores RTT situam-se no intervalo de 2900 a 3099 μs e aproximadamente 66,8% de 2800 a 3199 μs.

Assim, é necessário encontrar um equilíbrio entre a probabilidade de pertencer ao intervalo e a largura do mesmo, portanto, e com base nos testes realizados em redes Wi-Fi, o intervalo de 2800 a 3199, com uma largura de apenas 400 μs, pode ser considerado como o intervalo de confiança.

3.5 Cálculo do time skew

Sabendo que o tempo nos dois dispositivos flui de forma diferente, os valores do offset irão variar continuamente.

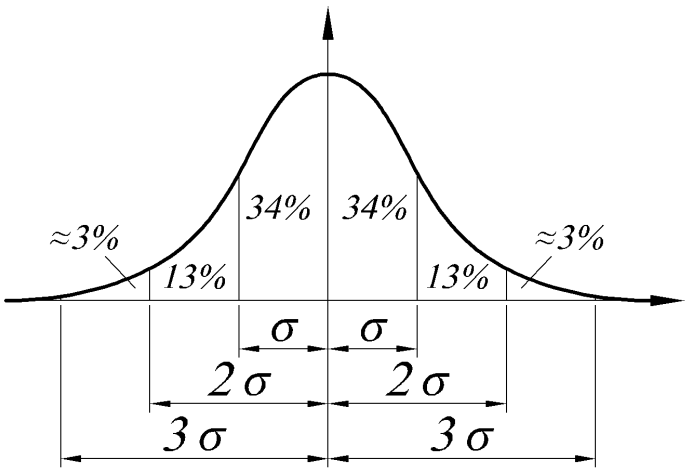


Figura 3.7: A regra empírica

Na Figura 3.8, é apresentado um gráfico que mostra uma seleção dos valores de offsets observados durante 20 minutos entre um dispositivo controlador e um dispositivo reprodutor com um intervalo de 10 segundos entre medições.

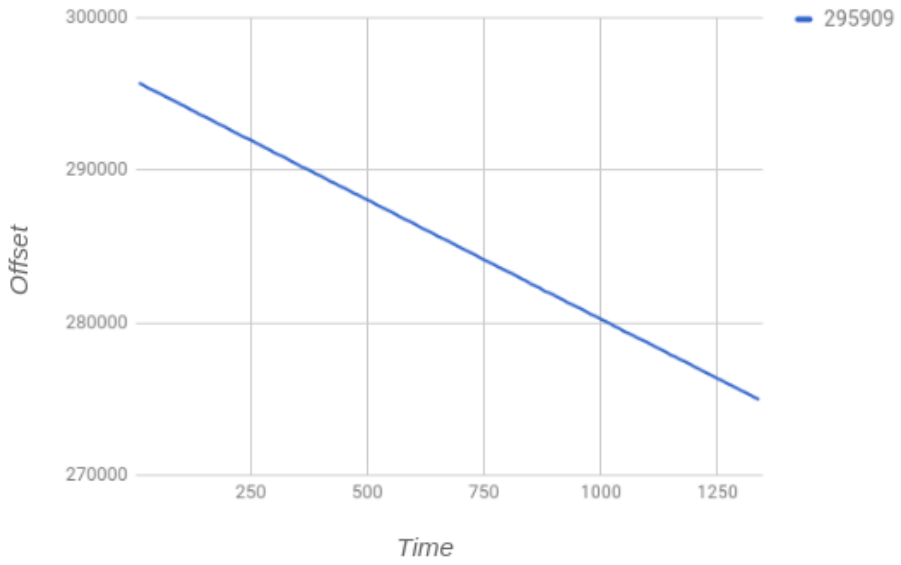


Figura 3.8: Valores de offset

Como se pode ver, neste caso particular, o valor do offset diminui constantemente (se o gráfico fosse incluir valores de offset por bastante mais tempo, o seu valor diminuir até chegar a zero e depois diminuiria para valores cada vez mais negativos a um ritmo mais ou menos constante).

O cálculo do time skew pode então ser feito usando a Fórmula 6, onde **n** e **i** representam os valores selecionados do offset, ou através da Fórmula 7, se se pretender uma variação menor na sua estimativa (função mais 'suave'). Neste caso **n** representa a quantidade de offsets tidos em consideração.

$$skew = \frac{X_i - X_n}{T_n - T_i} \tag{6}$$

$$skew = \frac{1}{n}\sum_{i=1}^{n}\frac{X_i - X_n}{T_n - T_i} \tag{7}$$

Uma outra abordagem ao cálculo do time skew utiliza regressão linear e pode ser feito com base na Fórmula 8.

$$\bar{y} = \beta + \alpha\bar{x} \tag{8}$$

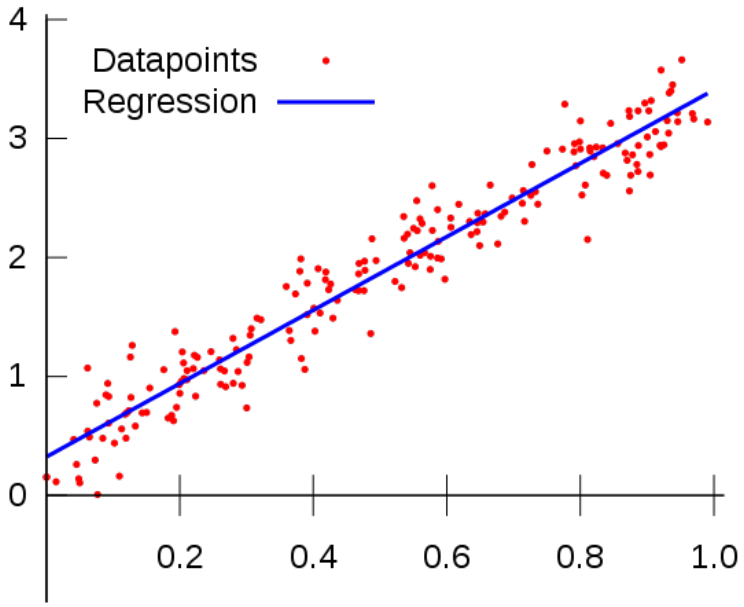


Figura 3.9: Um exemplo da estimação por regressão linear

O valor α é uma constante que influencia a inclinação da linha:

$$\alpha = \frac{\overline{xy} - \bar{x}\bar{y}}{\bar{x^2} - (\bar{x})^2} \tag{9}$$

Neste caso, **y** representa o valor do offset, **x** representa o tempo e o α representa o valor estimado para o time skew.

A abordagem escolhida dependeu de variados testes realizados com o intuito de escolher o método em que se conseguisse valores de time skew mais corretos. Mais detalhes serão explicados no capítulo seguinte dedicado a pormenores de implementação.

4

IMPLEMENTAÇÃO E TESTES

Com base nas estratégias concetuais apresentadas no capítulo anterior desenvolveu-se um programa experimental o bom funcionamento do algoritmo desenvolvido e que incluiu o cálculo do offset e do time skew. A Figura 4.1 mostra a estrutura principal do protótipo.

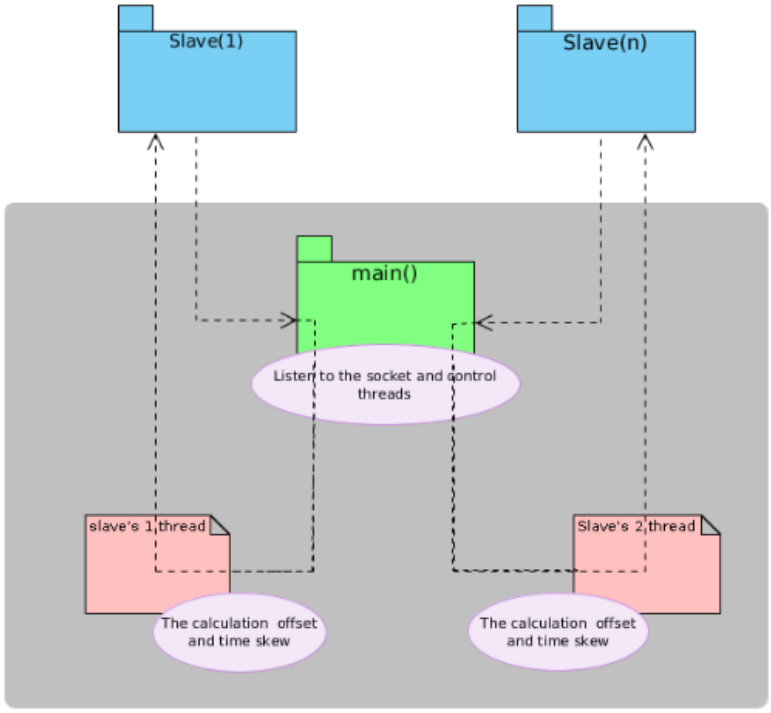


Figura 4.1: A estrutura principal do protótipo

De forma a minimizar o atraso nos cálculos dos valores desejados (offset e time skew), decidiu-se desenvolver e estabelecer todo o processo implementando processos paralelos, fazendo assim com que estes cálculos aconteçam em diferentes *threads* no master, melhorando a performance do sistema.

A principal tarefa da função *main()* consiste em criar uma nova thread associada a cada slave bem como uma conexão virtual entre o master e cada um desses slaves. A motivação para o cálculo do offset e do time skew decorrer em threads distintas prende-se, mais uma

	1		2		3		4	
	Range/ms	Percent	Range/ms	Percent	Range/ms	Percent	Range/ms	Percent
1	1700-9600	100%	1824-4803	84%	2193-3289	68%	2452-2917	46%
2	1700-8900	100%	1868-4507	83%	2292–3157	68%	2511-2915	40%
3	1800-9400	100%	1832-4578	84%	2181-3118	66%	2442-2917	44%
4	1500-10000	100%	1896-4315	81%	2332-3042	60%	2584-2932	41%
5	1600-9300	100%	2019-5026	80%	2358-3299	64%	2590-2995	46%
6	2000-10000	100%	2247-5830	84%	2440-4182	66%	2599-3179	54%
7	1500-9700	100%	1825-5202	83%	2291-3238	63%	2587-2967	48%
8	1700-9900	100%	1750-5630	84%	2073-3721	70%	2414-2949	47%
9	1500-9700	100%	1694-4835	82%	2169-3063	65%	2460-2933	40%
10	1200-10000	100%	1939-5332	77%	2443-3790	60%	2630-3203	44%
11	1600-9700	100%	2009-4715	88%	2618-3244	77%	2787-2993	54%
12	1700-9500	100%	2239-4434	85%	2700-3178	73%	2857-3002	49%

Tabela 4.1: Os possíveis intervalos de confiança

vez, com o objetivo de reduzir o *overhead* temporal destes cálculos, obtendo-se assim maior rapidez na obtenção dos resultados pretendidos.

4.1 INTERVALO DE CONFIANÇA

Assim que o master regista um novo slave é criado um processo dedicado para iniciar a determinação de um intervalo de confiança para o RTT. Depois de estabelecida a ligação, o master começa por enviar solicitações ao slave, incluindo o número de sequência de pacote. De seguida, o master entra em estado de espera pela resposta. Se o tempo limite de espera ultrapassar o valor máximo permitido, o master envia uma nova solicitação para o slave, desta vez com o número de sequência incrementado (+ 1). Se a resposta for recebida com o número de sequência esperado, o master calcula o RTT obtido e armazena-o em memória e aumenta a contagem das respostas bem-sucedidas. Caso contrário, o master ignora o pacote e entra em modo de espera pela resposta com o número de sequência esperado. Quando o limite de solicitações é atingido, o master, com base nos dados coletados e no intervalo *min* e *max*, calcula a esperança matemática, o desvio padrão e determina os novos limites do intervalo com base nos valores obtidos. Simultaneamente, considera qual a percentagem de RTT do número total que está incluído neste novo intervalo. Se a percentagem de entrada ultrapassar 60% da amostra, o master repete o processo, caso contrário, devolve o intervalo de confiança.

O resultado deste algoritmo pode ser visto na Tabela 4.1 e a sua representação gráfica na Figura 4.2.

Como se pode ver, o intervalo diminui com cada nova iteração e a probabilidade do RTT obtido estar neste intervalo, ou seja, de vir a ser considerado, diminui também. Na

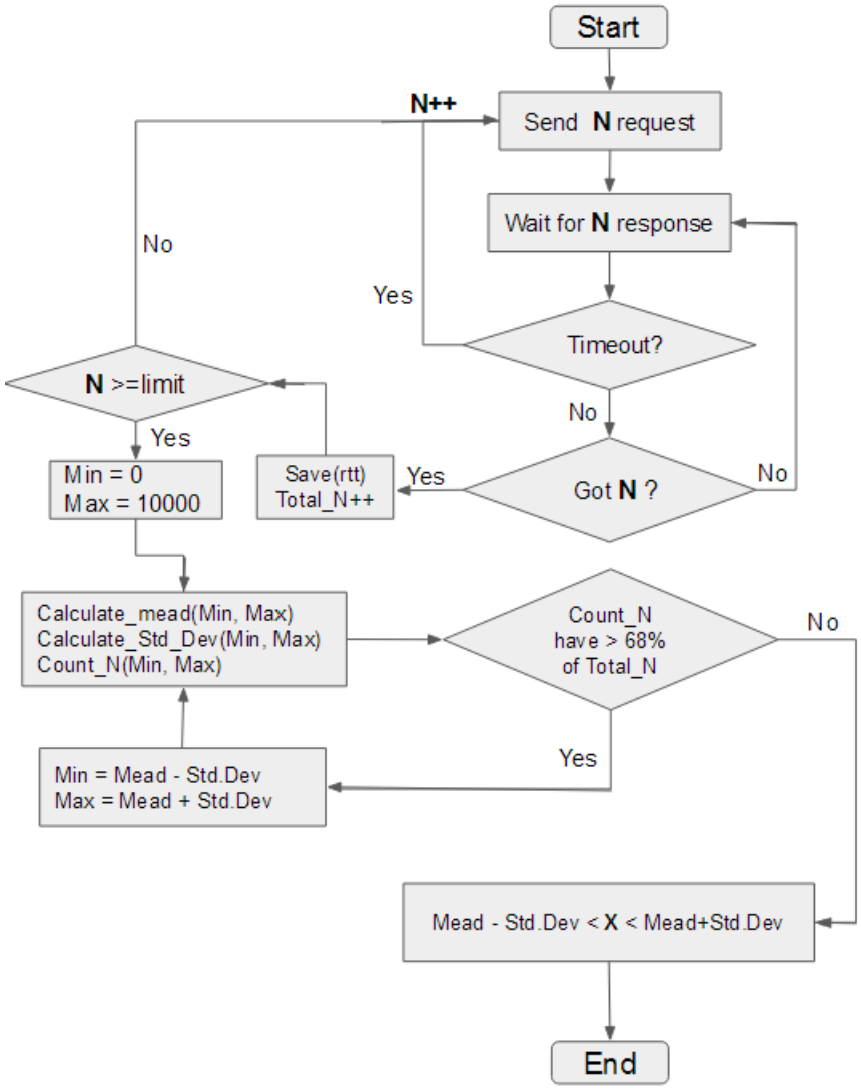


Figura 4.2: O algoritmo da determinação do intervalo de confiança

última iteração do último teste, representado na Tabela 4.1, o intervalo é de 2857-3002 microssegundos e tem uma probabilidade de 49%. O intervalo apresentado garante que o RTT é inferior ou igual a 145 ms, entenda-se, muito pequeno. Como consequência, tem-se um ΔT muito pequeno também, uma vez que o intervalo apresentado não permite grandes variações. O facto de o intervalo apresentado ser pequeno, permite ainda concluir que as 'idas' e 'voltas' das mensagens, quando comparadas entre si, nos vários pacotes de teste, são muito parecidas em termos temporais. Neste contexto, ou seja, tendo em conta os tempos semelhantes de 'ida' e 'volta', tem-se que a influência do ΔT nos cálculos efetuados é consideravelmente pequena/baixa.

4.2 OFFSET E TIME SKEW

O processo para determinar o offset é semelhante ao processo de determinação do intervalo de confiança, no entanto, com duas pequenas diferenças: *i)* o slave envia o seu tempo atual juntamente com a resposta; *ii)* todos os pacotes recebidos com um RTT que não se enquadram no intervalo de confiança são descartados.

Com base nos valores do offset, o master, descartando os valores que não se encontram no intervalo de confiança, calcula a média do offset. Este processo é repetido a cada 10 segundos, intervalo que pode, facilmente, ser alterado. Com o aumento do intervalo considerado aumentaria, também, a precisão do time skew quantificado. Tem-se que a precisão do time skew também melhora com o acréscimo do número de medições efetuadas. No entanto, este conceito envolve uma noção de equilíbrio entre o intervalo considerado e o tempo que é necessário para estabelecer um time skew mais preciso. A escolha do do valor de 10 segundos para o intervalo, obtido pelos testes realizados, permite recolher uma amostra de valores do offset que possibilita o cálculo de um skew suficientemente preciso, no menor intervalo de tempo possível. O algoritmo do cálculo está representado na Figura 4.3.

Depois de ter, pelo menos, dois valores do offset de um slave, o master está em condições de calcular o valor do time skew, de acordo com a Fórmula 7, ou usando a Fórmula 9, anteriormente apresentadas neste documento. A Tabela 4.2 mostra os valores de time skew calculados pela fórmulas apresentadas.

A tabela 4.2 contém a comparação dos resultados de das duas abordagens para o cálculo do skew, abordadas anteriormente. Observando os valores verifica-se que as fórmulas obtêm resultados muito semelhantes e, como tal, não é possível deduzir que uma seja melhor que a outra.

A precisão dos cálculos do offset e do time skew depende da dimensão do jitter na rede, por isso decidiu-se repetir os testes considerando um cenário que contempla uma rede Wi-Fi congestionada. Neste contexto, tornaram-se necessários pelo menos mais dois dispositivos adicionais e um programa utilitário, para se poder, através deste, controlar a congestão na rede. Como dispositivos adicionais usou-se um telemóvel e um computador (Desktop). Para controlar a carga da rede foi usado um utilitário chamado *iperf* que funciona em várias plataformas. Em primeiro lugar, fez-se um teste para descobrir a capacidade máxima da rede. Depois, com base nesta capacidade máxima, realizaram-se os testes com o cenário onde o telemóvel e computador fixo geram tráfego suficiente para congestionar a rede, enquanto o master e os slaves executam os mecanismos de cálculo dos offsets e dos time skews.

Note-se que todos os tempos de offeset e time skew apresentados têm como unidades μs.

Time	Skew (Fórmula 7)	Skew (Fórmula 9)
0	14.013	14.224
10	14.073	14.129
20	14.048	14.054
30	14.023	13.983
40	14.034	14.046
50	14.039	14.068
60	14.040	14.072
70	14.029	14.023
80	14.016	13.979
90	14.017	13.994
100	14.022	14.021
110	14.016	13.999
120	14.017	14.009
130	14.011	13.988
140	14.013	14.001
150	14.011	13.996
160	14.010	13.993
170	14.009	13.994
180	14.006	13.982
190	14.004	13.976
200	14.004	13.983
210	14.005	13.990

Tabela 4.2: Os resultados do algoritmo para cálculo do time skew (μs)

25%		50%		75%		100%	
Skew (7)	Skew (9)	Skew (7)	Skew (9)	Skew (7)	Skew (9)	Skew (7)	Skew (9)
13.167	13.167	14.536	14.872	14.335	15.101	14.224	15.052
13.106	13.049	14.553	14.904	14.345	15.079	14.237	15.041
13.899	15.233	14.587	15.018	14.361	15.097	14.247	15.016
14.583	16.662	14.612	15.082	14.372	15.085	14.271	15.073
15.005	17.106	14.656	15.247	14.398	15.160	14.281	15.055
15.303	16.736	14.680	15.292	14.422	15.221	14.284	15.005
15.300	16.138	14.703	15.330	14.437	15.228	14.296	15.005
15.426	16.212	14.724	15.367	14.445	15.204	14.309	15.012
15.547	16.344	14.729	15.314	14.880	15.227	14.321	15.018
15.559	16.131	14.724	15.209	15.354	15.264	13.863	15.075
15.554	15.931	14.712	15.086	16.348	15.240	16.136	15.182
15.574	15.892	14.708	15.015	13.639	15.260	16.385	15.150
15.592	15.871	14.721	15.046	16.034	15.280	15.138	15.053
15.572	15.709	14.720	15.005	14.758	15.238	15.082	15.064
15.564	15.629	14.719	14.964	15.520	15.299	14.945	15.073
15.541	15.501	14.725	14.971	15.506	15.257	17.262	15.050
15.578	15.659	14.734	14.988	13.867	15.261	14.073	14.996
15.580	15.644	14.757	15.081	14.405	15.264	15.174	15.025
15.581	15.628	14.763	15.083	14.380	15.300	14.500	15.025
15.586	15.634	14.761	15.041	14.619	15.351	16.451	15.065
15.601	15.689	14.768	15.051	14.390	15.407	13.552	15.000
15.596	15.643	14.772	15.044	14.670	15.366	14.669	15.045
15.594	15.619	14.775	15.037	16.315	15.465	14.335	15.046
15.594	15.614	14.772	15.000	15.532	15.442	15.582	15.102
15.594	15.604	14.780	15.017	16.495	15.406	14.219	15.082
15.612	15.687	14.781	15.005	14.767	15.356	15.615	15.158
15.606	15.640	15.057	14.996	14.914	15.366	16.696	15.121
15.609	15.649	14.598	14.993	17.425	15.403	15.463	15.093
15.611	15.651	15.268	15.014	16.069	15.297	13.747	15.019
15.604	15.604	14.913	15.010	14.996	15.303	15.117	15.008
Standard Deviation							
0,019301	**0,105250**	**0,143423**	**0,035366**	**0,979354**	**0,070159**	**1,010254**	**0,053175**

Tabela 4.3: Cálculo de time skew (μs) variando a carga na rede

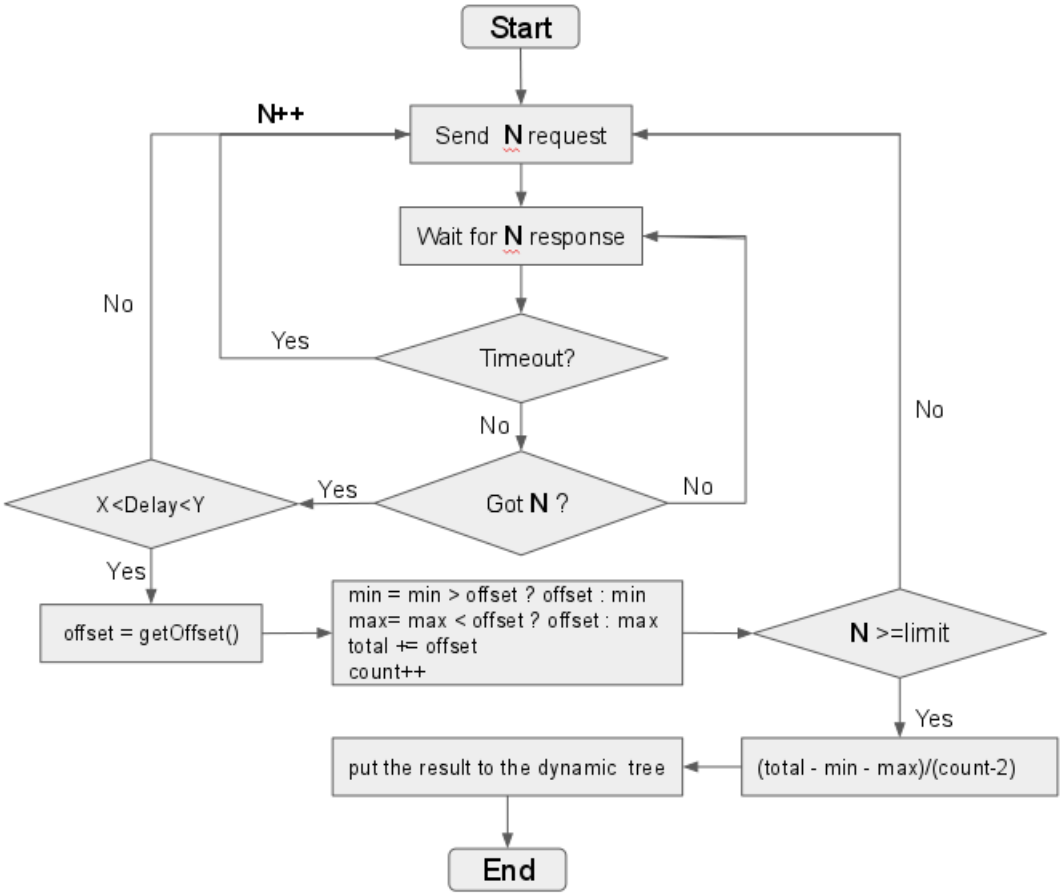


Figura 4.3: O algoritmo da determinação do offset

Como se pode ver na Tabela 4.3, o algoritmo com ajuda de regressão linear funciona melhor quando comparado com um algoritmo que contempla uma simples média dos valores obtidos. Partindo deste princípio, os testes seguintes foram baseados no algoritmo com regressão linear.

Assim, efetuaram-se uma série de testes para comparar o valor de diferentes time skews medidos, em instantes diferentes durante um dia, pelo osciloscópio com os calculados pelo protótipo desenvolvido. Por forma a obter uma maior precisão, os testes foram realizados em paralelo, isto é, em cada teste foram observados os valores do osciloscópio e os valores calculados pelo algoritmo, em simultâneo. Estes testes foram realizados com recurso a:

- Dois dispositivos RaspberryPi;
- Placa de som genérica, utilizada como osciloscópio;
- Algoritmo desenvolvido;
- Aplicação Multi-Instrument.

Na Tabela 4.4 mostram-se os resultados obtidos com a realização dos testes mencionados. Pode concluir-se que o os algoritmos implementados no código desenvolvido determinam o valor do time skew com enorme precisão. Assim, prova-se que o algoritmo funciona como esperado e que o resultado do seu uso pode ser usado para a resolução dos problemas de sincronização de fluxos canais de áudio.

N	Skew (Osciloscope)	Skew (Algorithm)
1	2.257	2.135
2	2.236	2.148
3	2.591	2.56
4	2.706	2.667
5	2.661	2.615
6	2.572	2.481
7	2.185	2.031
8	2.575	2.572
9	2.565	2.673

Tabela 4.4: Medição e cálculo do time skew (μs)

4.3 SINCRONIZAÇÃO INICIAL

O próximo passo do mecanismo de sincronização passa por, com um simples *trigger*, tentar iniciar a reprodução síncrona de fluxos áudio nos dispositivos reprodutores. O master, com base nos valores do offset, obtidos pelo software desenvolvido, gera o tempo atual nos dispositivos e envia o comando de reprodução, marcado com o tempo de início, com uma margem de K segundos (para acomodar o tempo de transmissão, receção e preparação duma quantidade suficiente de dados num buffer para reprodução áudio). Do lado dos dispositivos reprodutores, após a receção do pedido de reprodução e dos dados do fluxo de áudio, prepara-se a placa de som e o buffer específico para reprodução de áudio local e inicia-se a reprodução no momento definido pelo master. Estes procedimentos estão resumidos no esquema da Figura 4.4.

Foram efetuados uma série de testes e cujos resultados de dessincronização inicial se podem observar na Tabela 4.5. Pode observar-se que existiu uma dessincronização inicial que variou entre 0.249 e 10.770 ms. Mesmo no caso mais grave, de 10.77 ms, não será percetível qualquer eco ou diferença nos sons vindos das diferentes fontes, porque esta dessincronia é ainda inferior 30 ms. No entanto, tendo em conta o facto de que, após o início da reprodução de áudio, a dessincronia entre os canais será cada vez maior, idealmente deve tentar ter-se uma dessincronização inicial com valores mais próximos de zero.

Relembra-se que, nos cálculos do offset pode haver um erro no ΔT porque, tal como descrito anteriormente, não é possível saber exatamente quanto tempo demora a solicitação

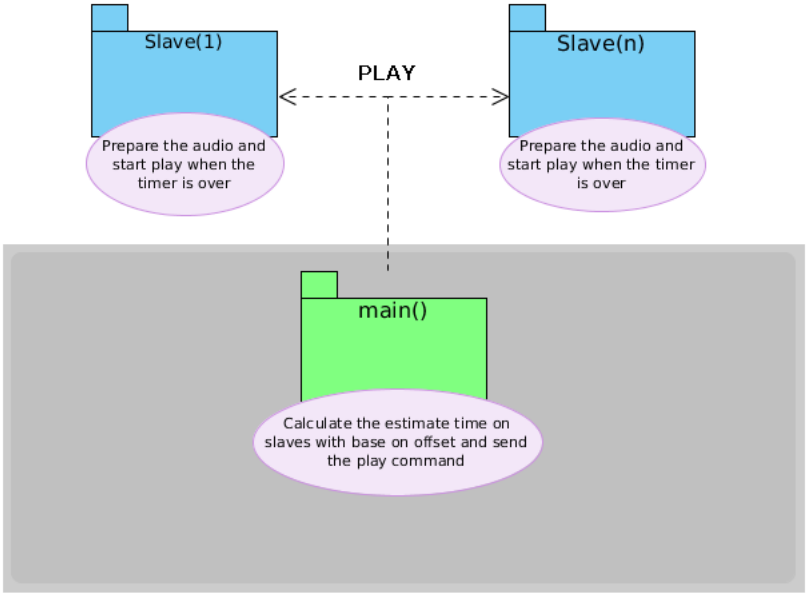


Figura 4.4: A sincronização no início da reprodução áudio

do master para o slave, mas é este valor que representa a dessincronização inicial, isto é, o ΔT representa genericamente o valor inicial de dessincronia.

Os resultados dos testes sugerem também a influência de outros fatores nos valores de dessincronia no início da reprodução, uma vez que, a dessincronização máxima possível com um intervalo de 2600-3000 μs não deve exceder 3 ms.

Considere-se um exemplo onde, no primeiro dispositivo, a 'ida' é instantânea, ou seja, demora 0 ms e a 'volta' 3 ms, sendo que, no segundo dispositivo, é ao contrário. Este exemplo é representado na Figura 4.5.

N	Dessincronização/ms	N	Dessincronização/ms
1	8.684	11	0.907
2	4.716	12	10.770
3	8.593	13	0.566
4	4.013	14	0.362
5	1.451	15	10.158
6	8.389	16	10.385
7	8.253	17	9.931
8	2.607	18	0.317
9	8.026	19	0.249
10	7.074	20	9.727

Tabela 4.5: Os resultados da dessincronia inicial

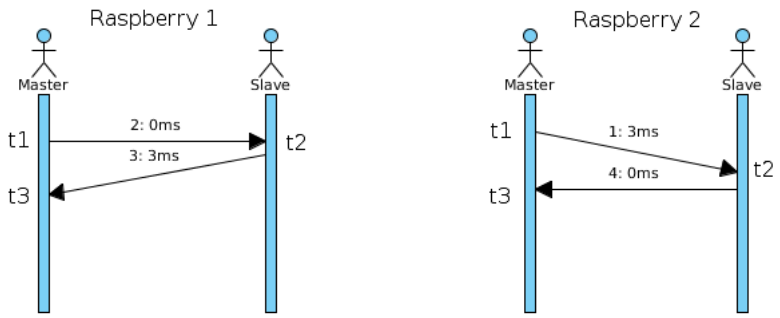


Figura 4.5: Um exemplo para prova de conceito ΔT_i= 3 ms

Em ambos os casos, o RTT é o mesmo, mas o erro do ΔT é diferente. Obviamente este não é um caso real, mas permite verificar claramente, que o ΔT não poderia ultrapassar 3 ms.

No caso do primeiro dispositivo, o envio do pacote foi imediato, ou seja, o atraso entre envio e receção foi de 0 ms, daí os tempos t1 e t2 acontecerem no mesmo instante, ou seja, serem iguais. Com este offset, o master sabe exatamente qual o tempo no dispositivo 1. No entanto, nos cálculos do offset retira-se a metade do valor do RTT ao valor de t2, tentando aproximar o tempo t2 à medida temporal do tempo t1 mas, como o valor de t2 foi obtido sem atraso, ou seja, obtido como tempo real, obtém-se um cenário onde, no master, resulta uma representação errada do tempo no dispositivo 1, com um erro de -1.5 ms, ou seja, t1 está com um atraso de 1.5 ms.

Relativamente ao exemplo do segundo dispositivo, a situação é análoga mas inversa, ou seja, t1 tem um avanço de 1.5 ms em relação ao verdadeiro valor de t2. Quando o master enviar um comando para reproduzir o fluxo dum canal áudio, o primeiro dispositivo irá iniciar a reprodução com 1.5 ms de avanço em relação ao master e o segundo dispositivo, por sua vez, irá iniciar a reprodução com o mesmo valor, mas de atraso, em relação ao master, resultando numa diferença global de 3 ms em relação aos dois dispositivos.

Torna-se claro que a razão para esta dessincronização é o atraso entre o envio dos dados para o buffer de reprodução e o início da reprodução propriamente dita. Portanto, torna-se necessário excluir este atraso nos estudos sem usar uma placa de som, para perceber se a dessincronização inicial mudará.

Para resolver esse problema, foi testado o uso das postas de GPIO(General-purpose input/output) no dispositivo, em vez de reproduzir ondas sinusoidais, por forma a ser possível aplicar corrente nas postas de GPIO por software, as portas de GPIO que por sua vez, podem ser ligados diretamente a um osciloscópio.

Foi efetuada mais uma série de testes de que resultaram os valores que estão apresentados na Tabela 4.6. Pode facilmente confirmar-se que foram obtidos excelentes resultados

N	Dessincronização/μs	N	Dessincronização/μs
1	385,5	16	294,8
2	113,4	17	158,7
3	68	18	272,1
4	136,1	19	45,3
5	90.7	20	90,7
6	68	21	158,7
7	181,4	22	340,1
8	158,7	23	317,5
9	226,8	24	90,7
10	113,4	25	22,7
11	136,1	26	272,1
12	317,5	27	0.1
13	362,8	28	45,4
14	158,7	29	317,5
15	45,4	30	45,4

Tabela 4.6: Os resultados da dissincronia inicial usando portas de GPIO.

globais, com uma dessincronização inicial ao nível do nano-segundo. Por outro lado, isto sugere que as funções de reprodução áudio da biblioteca Alsa, utilizada no contexto do projeto, introduzem atrasos que influenciam os valores de dessincronização inicial.

Assim, foi decidido realizar uma série de experiências para estudar o atraso entre o pedido de reprodução e o início da mesma. A Figura 4.6 representa um diagrama de blocos do procedimento para a experiência.

Nos testes que decorreram, após o 'envio' dos dados do buffer de reprodução áudio para a placa de som, através de um comando é ativado uma porta de GPIO. A dessincronização resultante pode assim ser medida, provando não só a presença de um atraso introduzido pelas funções de reprodução local de áudio, como também quantifica a sua dimensão, excluindo a hipótese de outros fatores adicionais poderem afetar relevantemente os valores de dessincronia.

Na Figura 4.7 tem-se uma representação do atraso entre a aparência de corrente na porta de GPIO, representada pela linha vermelha, e o início da reprodução do fluxo de áudio (onda sinusoidal), na curva azul. Na Tabela 8 estão apresentados a variância, a variação da latência entre a chegada de corrente e início da onda sinusoidal. Pode verificar-se que a latência mínima e máxima nos dispositivos reprodutores é aproximadamente igual a 2.9 ms e 12.65 ms, respetivamente, o que justifica a dessincronização máxima da Tabela 4.5, que, ainda assim, é muito boa, impossível de ser detetável pelo ouvido humano.

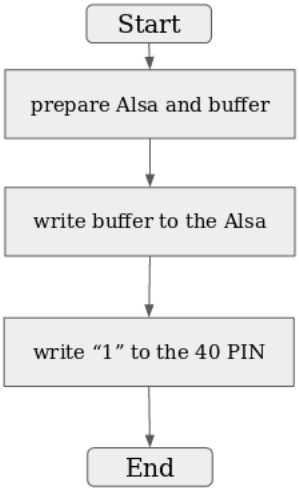


Figura 4.6: Diagrama de blocos para determinação do atraso introduzido pelo módulo Alsa

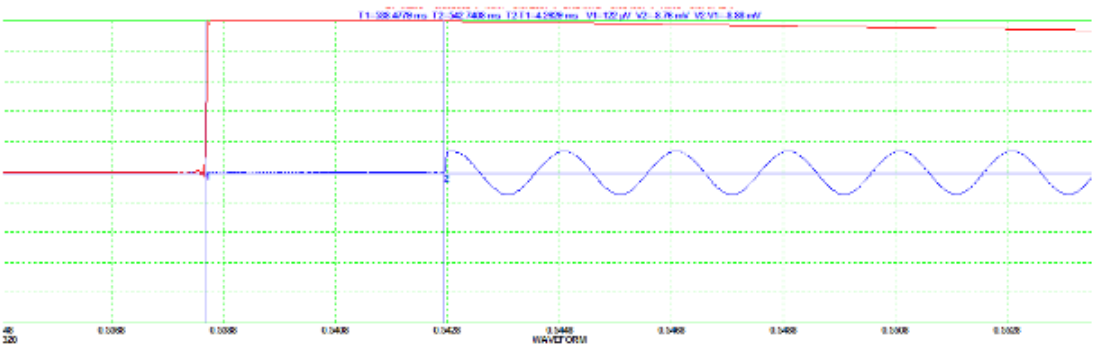


Figura 4.7: Atraso introduzido pelo módulo Alsa

4.4 Sincronização durante a reprodução de áudio

Para o próximo passo foi fundamental conseguir manter a dessincronização tão pequena quanto possível e o mais estável possível durante todo o tempo de reprodução do fluxo áudio (tempo esse que, por exemplo, em distribuição de áudio de fontes contínuas de sinal - como canais de rádio ou de *podcasts* pode ser muito grande, na ordem de várias horas, dias, ou até ser 'non-stop').

Para o conseguir atingir este objetivo utilizou-se a estratégia de eliminação ou adição de amostras individuais ao *buffer* fluxo de áudio presente em cada um dos dispositivos reprodutores. Com base no valor do time skew e na frequência de amostragem do fluxo de áudio, pode calcular-se a quantidade de amostras que devem ser excluídas ou adicionadas para ir continuamente compensando o valor de time skew. Assim, o master, além do tempo inicial de reprodução (já adaptado com o offset respetivo), enviará ao dispositivos reprodutores o valor do time skew.

Dispositivo reprodutor 1			Dispositivo reprodutor 2		
7,482	12,562	11,927	9,274	7,482	9,931
8,208	5,101	7,301	7,505	3,333	12,176
5,525	4,603	6,847	8,933	6,371	6,87
10,521	8,095	8,457	3,945	4,58	11,677
5,646	3,287	4,262	9,387	9,138	12,335
9,5	9,5	6,281	12,312	11,654	7,346
9,75	7,324	9,002	11,859	8,185	12,38
10,679	11,904	7,528	6,553	5,442	9,954
3,242	10,43	12,63	11,813	2,925	11,904
8,661	5,759	7,868	6,87	5,532	12,652
Desvio Padrão					
2,644497592			3,01307254		

Tabela 4.7: A variância do atraso (ms) introduzido pelo módulo Alsa

O algoritmo implementado para cálculo do número de amostras a adicionar ou remover foi apresentado em [22].

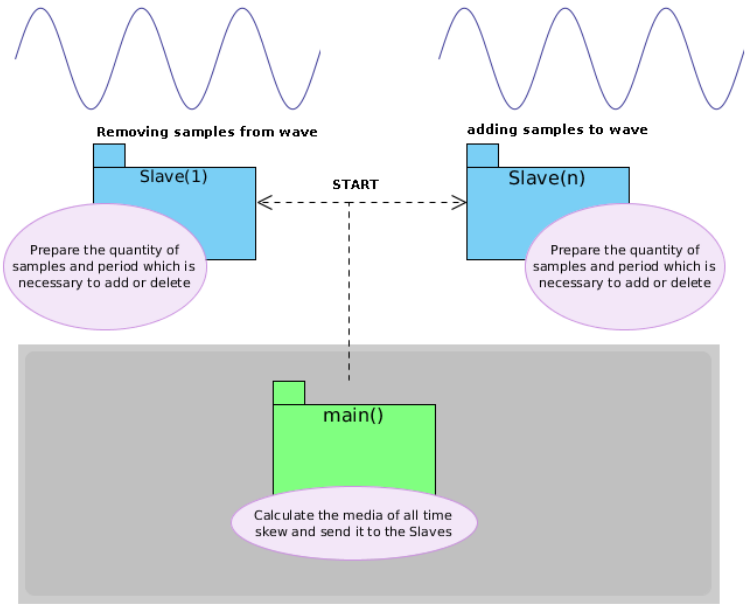


Figura 4.8: A sincronização ao longo da reprodução

O protótipo foi sujeito a novos testes, desta vez usando o algoritmo de manutenção de sincronização de canais ao longo do tempo de reprodução. Como se pode ver na Tabela 4.8, ao fim de 7 horas de reprodução ininterrupta os dispositivos reprodutores apresentavam uma dessincronia de 9.5 ms. Apesar deste valor ainda ser excelente, nota-se claramente um ritmo crescente e constante deste valor ao longo do tempo, o que é indesejado.

Duração	Dessincronização
0h	1.337
1h	2.038
2h	3.083
3h	4.262
4h	5.566
5h	7.018
6h	8.215
7h	9.500

Tabela 4.8: Valores de dessincronia (ms) ao longo do tempo, com algoritmo desenvolvido

Time	Dessincronization
0h	3.404
1h	3.204
2h	3.112
3h	3.179
4h	3.125
5h	3.077
6h	3.086
7h	3.027
8h	3.044
9h	2.964
10h	2.919
19h	2.500

Tabela 4.9: Dessincronia (ms) ao longo do tempo, com algoritmo adaptado com time skew dinâmico

Os testes realizados utilizaram código implementando o algoritmo tal como descrito em [22], prevendo a utilização dum único valor do time skew que foi enviado para os dispositivos no início da reprodução.

Posteriormente, o algoritmo foi melhorado a fim de que os dispositivos fossem capazes de atualizar, periodicamente, o valor do time skew conforme ia sendo calculado pelo master a uma frequência pre-definida. Depois da alteração foram realizados novos testes. Como se pode verificar pela observação da tabela 4.9 a dessincronização resultante manteve-se sempre na ordem dos 3 ms durante 19 horas seguidas. Ou seja, o resultado obtido melhorou significativamente em relação à iteração anterior e preenche completamente os objetivos propostos inicialmente.

5

APLICAÇÃO FINAL

Um dos objetivos do projeto previa o desenvolvimento dum sistema aplicacional com interface para utilizadores, com uma arquitetura centralizada num dispositivo controlador da distribuição do sinal áudio da fonte (coleção de músicas, por exemplo) para os equipamentos remotos de reprodução de áudio digital com um ou mais canais. Este sistema integraria o código protótipo desenvolvido para implementar os algoritmos e mecanismos definidos e testados durante o projeto.

O correto funcionamento deste sistema completo seria utilizado para testar e provar que os conceitos, algoritmos e estratégias escolhidos definem uma solução válida para resolver os requisitos enunciados utilizando apenas recursos modestos e suportada por tecnologias de comunicação de redes locais o mais comuns possíveis.

Assim, o sistema protótipo base foi implementado em ambiente Linux, utilizando versões/distribuições convencionais, sem quaisquer configurações especiais. A solução proposta suporta a distribuição de fluxos de áudio com qualquer número de canais (mono, estéreo, 5.1, 7.1, etc.) em sistemas finais reprodutores, individualmente ou em grupos de canais, com ou sem cópia (é possível distribuir o mesmo canal áudio por vários sistemas reprodutores finais, o que, na realidade, é o caso de utilização mais exigente em termos de sincronia, ou melhor, em que a dessincronia pode ser notada com mais facilidade).

Mais uma vez, nos testes realizados, o sistema aplicacional completo incluiu dois dispositivos finais reprodutores e um controlador e distribuidor dos canais áudio de sinais estéreo ou mono, conforme esquema da Figura 5.1:

- Aplicação cliente controlador Android
- Aplicação servidor musical Linux.
- Dispositivos reprodutores com ligação a sistema de altifalantes baseados em sistemas Raspberry-Pi Linux.

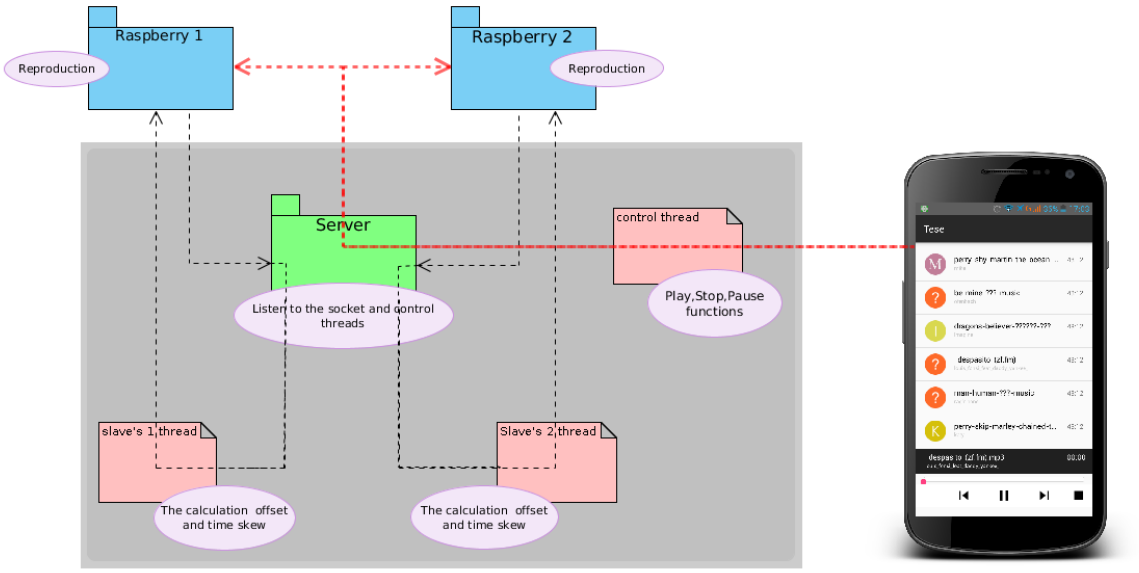


Figura 5.1: Arquitetura do produto final

5.1 Cliente controlador de áudio

As funcionalidades do cliente controlador são a visualização da lista de arquivos de áudio e o controlo da reprodução. Para maior mobilidade e conforto da aplicação foi implementada num equipamento móvel com sistema operativo Android. Foi desenvolvido um aplicativo simples sobre um protocolo de comunicação cliente-servidor com base no protocolo UDP. O interface pretendeu ser intuitivo e simples. O cliente pode executar funções como:

- **Get music list**: A lista de arquivos é solicitada ao servidor em intervalos regulares.
- **Play**: Inicia a reprodução do arquivo de áudio selecionado;
- **Pausa**: Pausar a reprodução;
- **Volume**: Controlar o volume da reprodução.

5.2 Dispositivos reprodutores de áudio

Estes dispositivos têm como função principal a reprodução final e a sincronização de áudio de canais individuais. São os slaves no sistema de distribuição de áudio. Além das capacidades próprias dos slaves conforme descritas no capítulo anterior, estes dispositivos também têm as funcionalidades para:

- **Play**: (Re)começar a reprodução local;

- **Stop**: Termina a reprodução local e liberta os recursos;
- **Pause**: Pausa a reprodução;
- **Volume**: Altera o volume de reprodução.

De notar que todo o fluxo de áudio é dividido por blocos numerados. Quando estes dispositivos recebem os blocos de dados processam-nos, preparam o áudio e gravam-no adequadamente no buffer específico para reprodução áudio no sistema operativo local. Se algum pacote for perdido o cliente fará a solicitação do mesmo novamente.

No que diz respeito ao volume, o ALSA usa uma escala de volume exponencial, ou seja, é necessário utilizar uma função de conversão de uma escala linear, de 0 a 100 por exemplo, para saber que valor passar ao ALSA. Para aumentar ou diminuir o som foi usada a Fórmula 10.

$$y = \beta * ln(\frac{x}{\alpha}) \tag{10}$$

Onde o "y" é o valor da escala do ALSA e o "x" é o valor da escala linear. O α e o β são as constantes. Para calcular α e β foram usadas Fórmula 11 e Fórmula 12

$$\alpha = \frac{y_{max} * ln(x_{min}) - y_{min} * ln(x_{max})}{y_{max} - y_{min}} \tag{11}$$

$$\beta = y_{max} / ln(\frac{x_{max}}{\alpha}) \tag{12}$$

5.3 SERVIDOR MUSICAL

O servidor, ou master, é um centro multi-funcional de todo o sistema e é executado no modo *daemon*. Suporta e/ou executa funções como:

- O estabelecimento da comunicação com os dispositivos reprodutores;
- O estabelecimento da comunicação com as aplicações clientes/controladoras;
- A organização da comunicação virtual da aplicação cliente com os dispositivos reprodutores;
- Calcula todos os valores de offset e de time skew;
- Gere o arquivo musical;
- Procede à codificação e descodificação de alguns formatos de armazenamento de ficheiros de música mais comuns.

Alguns dos processos internos do servidor são realizados em threads individuais. A hierarquia das threads pode ser vista na Figura 5.2.

A comunicação entre o servidor e os dispositivos reprodutores e entre o servidor e os clientes controladores é feita por mensagens encapsuladas no protocolo UDP. Em cada mensagem, o primeiro byte é responsável por identificar a ação ou processo que é necessário executar. A thread principal, que é representada a verde, é o core do sistema.

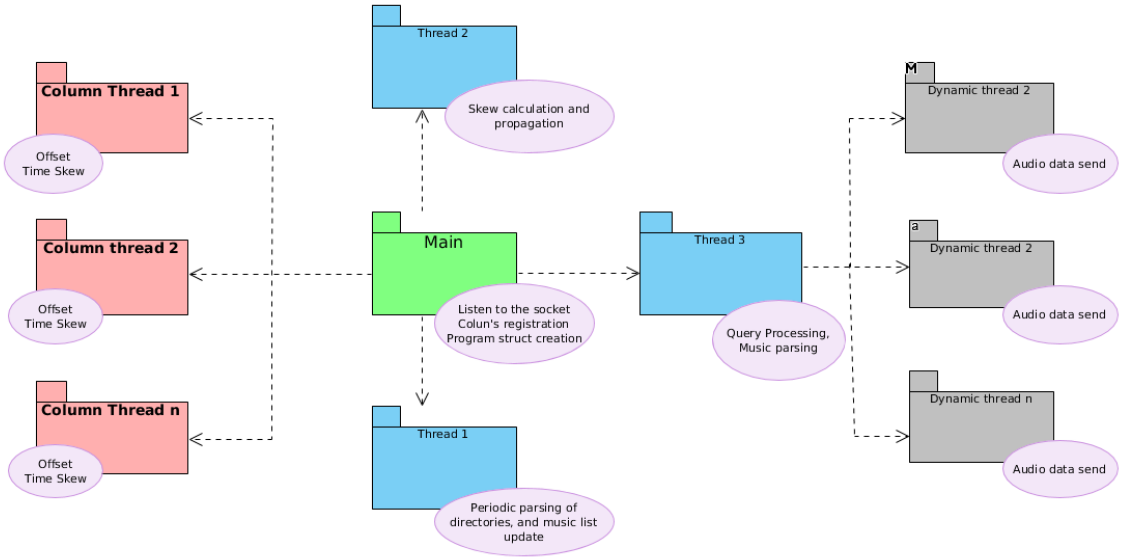


Figura 5.2: A hierarquia das threads do servidor musical

Em cada uma das threads, pintadas a vermelho, os processos são realizados para determinar o offset, o time skew e o intervalo de confiança, ou seja, executar todos os algoritmos descritos nas secções 4.1 e 4.2. Uma das threads auxiliar, representada a azul, envia periodicamente um pacote de broadcast para descobrir os dispositivos reprodutores na rede. Foi escolhida esta estratégia porque assim estes dispositivos não precisam de saber o endereço IP e a porta UDP do servidor. Este pacote também serve como HeartBeat, para determinar se a coluna está online ou offline. Esta thread atualiza periodicamente a lista dos arquivos áudio com base num ficheiro de configuração. Outra thread auxiliar, representada a azul, faz o processo do cálculo e da distribuição do time skew para cada um dispositivos reprodutores. A Thread representada a cor-de-laranja é ocupada a processar/redirecionar os pedidos dos clientes para os dispositivos reprodutores.

Neste momento, o servidor suporta apenas os formatos MP3 e o WAV. O MP3 é um dos formatos mais comuns e populares para codificação digital de informações de áudio com perdas. Para processar arquivos MP3 foi utilizada a biblioteca mpg123. Esta biblioteca oferece uma ampla gama de funções para analisar metadados e descodificar dados para o formato PCM, que é o formato interno que o servidor e os dispositivos reprodutores usam.

A estrutura do arquivo WAV é mostrada na Figura 5.3. Os primeiros 44 bytes são metadados e o resto são os dados em formato PCM.

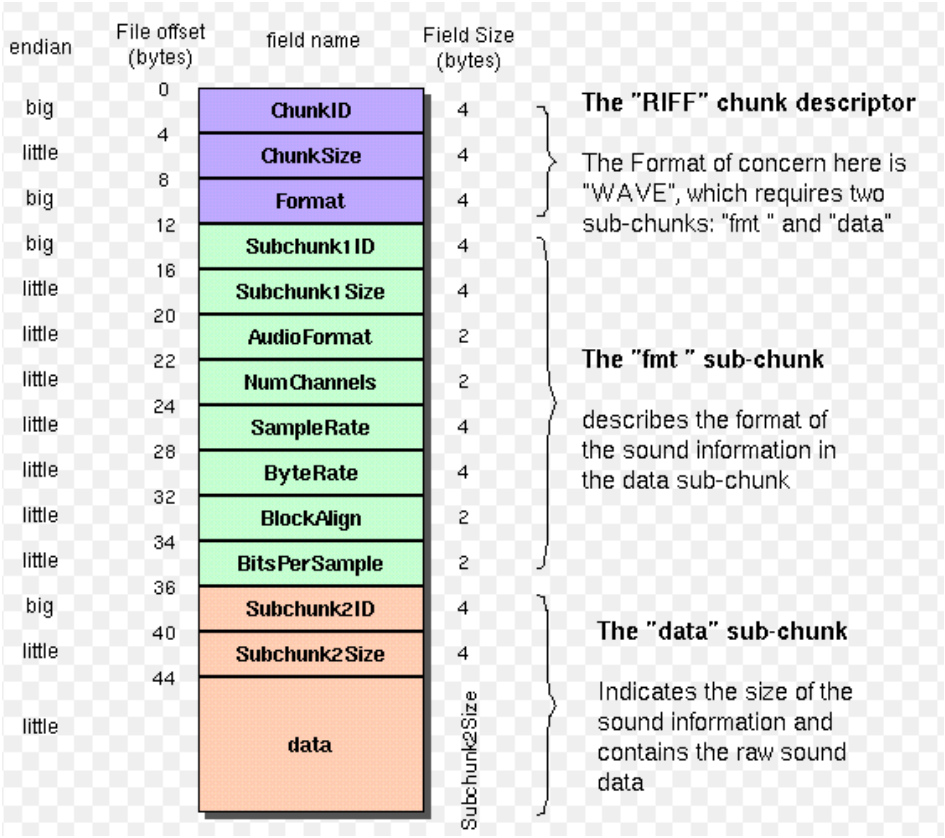


Figura 5.3: A estrutura do arquivo WAV

No caso de um arquivo WAV, o processo de análise começa com a leitura dos primeiros 44 bytes. Com base nesses dados, o servidor cria repositórios para cada canal. Após a extração desses dados o servidor cria uma thread para tratar a distribuição e sincronização em cada dispositivo reprodutor.

Os dispositivos reprodutores preparam o que precisam para receber e reproduzir os dados áudio. Quando estão prontos para a receção, o servidor começa a transferência de dados do canal correspondente. Todo o fluxo de dados é dividido em blocos, ou pacotes. O processo de transferência dos dados é apresentado na figura 5.4, que também inclui o controlo da perda de blocos.

- **Passo 1**: A solicitação da lista de blocos;
- **Passo 2**: A geração da lista e envie a para o servidor;
- **Passo 3**: O envie os dados da lista.

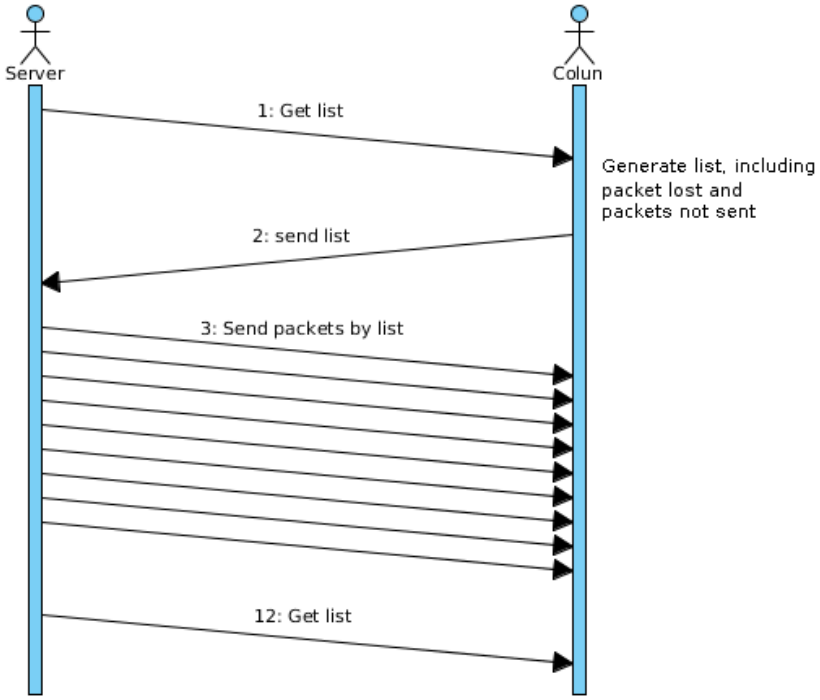


Figura 5.4: O algoritmo de transferência de dados com o controlo da perda de blocos

O servidor repete esses passos até que receba uma lista nula. Quando cada dispositivo reprodutor tiver o buffer com dados suficiente para não haver interrupções na reprodução, o servidor envia o comando para iniciar a reprodução. No final da transferência de dados, as threads vão ser terminadas e os recursos vão ser libertados.

No que diz respeito às outras funções da aplicação controlador, como parar, pausar ou alterar o volume, o servidor simplesmente redireciona os pedidos do cliente para os dispositivos reprodutores.

6

CONCLUSÕES E TRABALHO FUTURO

No contexto deste projeto de dissertação, foi abordado com sucesso o problema da sincronização de áudio em sistemas de distribuição de canais áudio por dispositivos remotos e independentes, ligados entre si por uma rede Wi-Fi comum.

O objetivo principal do trabalho apresentado consistia na definição e implementação e teste de mecanismos pelos quais pudesse ser alcançada a reprodução síncrona de canais de áudio individuais utilizando apenas como suporte algoritmos e tecnologias de uso comum, simples e que não exigissem recursos - hardware software, ou configurações - especiais.

Para resolver os problemas que foram surgindo, estudaram-se os seguintes assuntos:

1. Causas de dessincronização dos relógios nos dispositivos duma rede local;
2. Forma como essa dessincronização afeta o processo de reprodução de áudio;
3. Protocolos de sincronização temporal e de que forma podem ajudar a eliminar ou diminuir relevantemente essa dessincronia.

Foram investigadas as soluções já existentes e os trabalhos relacionados. Descobriu-se que o tempo (e, consequentemente, os processos), em diferentes dispositivos, flui diferentemente, ainda que o hardware e o software seja idêntico. Estudou-se a forma como a deriva temporal dos sistemas influencia a sincronia na reprodução de áudio distribuído por vários dispositivos numa rede local sem fios.

Através de testes empíricos foi possível confirmar que acima dos 25 ms de dessincronia na reprodução de fluxos de áudio mono o seu efeito começa a ser notado pelos ouvintes com maior capacidade auditiva e que, portanto, era necessário atingir e manter valores de dessincronia significativamente abaixo deste patamar (no caso de fluxos de som não-mono, a exigência não é tão apertada).

Foram investigados os protocolos e algoritmos existentes para implementar sincronização temporal de sistemas em redes locais, ainda que não diretamente ligados à sincronia de fluxos áudio. Foram também estudados outros projetos anteriores que estudaram e apresentaram resultados e estratégias úteis para a resolução, pelo menos parcial, dos mesmos

problemas de sincronização de áudio. Alguns dos resultados desses trabalhos influenciaram diretamente muitas das soluções integradas na arquitetura final aqui proposta, bem como os algoritmos e mecanismos desenvolvidos, implementados e testados.

A utilização do protocolo UDP como único protocolo de transporte de todo o sistema, a não utilização de qualquer outro protocolo aplicacional de sincronização de temporal já existente, como o NTP ou o PTP, e a definição duma solução independente das tecnologias de rede local utilizadas foram decisões conscientes no sentido de simplificar ao máximo o design da solução e minimizar os recursos necessários.

A implementação e teste dum protótipo sobre redes locais Wi-Fi deve-se às limitações adicionais que este tipo de rede local apresenta, quando comparada com redes locais cabladas. Se a solução funcionar neste tipo de tecnologia estrá pronta para funcionar em tecnologias menos restritivas (em termos de largura de banda, latência e jitter). Além disso, as redes Wi-Fi são, hoje em dia, a tecnologia mais comum para interligação de dispositivos em redes locais.

É importante realçar que a maior parte das soluções comerciais atuais omitem quaisquer informações técnicas detalhadas acerca de que protocolos e mecanismos são usados na sincronização do áudio em canais individuais, quando existe.

Outro problema de tecnologias mais difundidas para distribuição de conteúdos multimédia têm o problema de necessitarem de licenciamentos não gratuitos e de não fornecerem, até à data, mecanismos de sincronização de fluxos de áudio por canais individuais reproduzidos em dispositivos remotos.

A primeira fase de desenvolvimento foi a definição e implementação dos algoritmos para cálculo dos valores da deriva temporal (offset) e da sua variação ao longo do tempo (time skew), uma vez que foi concluído por experimentação laboratorial que a obtenção de valores o mais precisos possíveis é essencial para se implementar com sucesso os mecanismos de sincronização inicial e de manutenção estável dessa sincronização durante todo o período de reprodução do fluxo de áudio.

Utilizando os resultados dos testes exaustivos realizados às implementações construídas, podemos concluir que os mecanismos e algoritmos propostos atingem patamares de sincronização excelentes, tanto a inicial como a mantida ao longo do tempo de reprodução. Os valores de dessincronia obtidos para a versão final do protótipo garantem, inequivocamente, que é possível, com a tecnologia desenvolvida, manter a sincronia musical mesmo em condições adversas, como numa rede Wi-Fi congestionada, com dispositivos reprodutores com poder computacional muito modesto e de baixo custo e com sistemas operativos gratuitos sem quaisquer configurações especiais.

Além disso, ainda no âmbito do projeto, foi possível desenvolver um sistema completo de distribuição e sincronização de áudio com todos os componentes necessários, incluindo uma aplicação controladora para dispositivos móveis.

Em suma, é seguro concluir que todos os objetivos principais do projeto foram alcançados com sucesso.

Como trabalho futuro seria essencial apresentar os resultados deste trabalho em formato artigo científico e tentar a sua publicação e apresentação numa conferência interessante.

Além disso, com mais tempo disponível deveria ser possível continuar o desenvolvimento e integração de funcionalidades novas e melhorar a performance de algumas já existentes em todos os componentes do sistema. Nomeadamente o suporte de mais formatos de áudio, incluindo formatos de compressão sem perdas de dados como o FLAC, ou o suporte à distribuição de fluxo de áudios multi-canal 5.1 e 7.1, já que, nos algoritmos e mecanismos utilizados, não existem restrições ao número de canais áudio ou dos dispositivos reprodutores na rede local.

Também seria importante investigar outros métodos de filtragem dos valores obtidos para o offset por forma a verificar se é razoável esperar ainda melhoraria na precisão do seu cálculo.

BIBLIOGRAFIA

[1] E. Ferro and F. Potorti, *Bluetooth and Wi-Fi wireless protocols: A survey and a comparison*, Wireless Communications, IEEE, no. February, pp. 12–26, 2005.

[2] Hui Zhou, Charles Nicholls , Thomas Kunz, Howard Schwartz *Frequency Accuracy and Stability Dependencies of Crystal Oscillators*, Carleton University, Systems and Computer Engineering, Technical Report SCE-08-12, November 2008

[3] Wilson Boulevard *RFC:793, 1122, 3168, 6093, 6528 - TRANSMISSION CONTROL PROTOCOL, PROTOCOL SPECIFICATION* September 1981

[4] J. Postel *RFC 768 - User Datagram Protocol, PROTOCOL SPECIFICATION*28 August 1980

[5] E. Kohler, M. Handley, S. Floyd *RFC 4340 5595, 5596, 6335, 6773 - Datagram Congestion Control Protocol, PROTOCOL SPECIFICATION*March 2006

[6] R. Stewart, Q. Xie, K. Morneault, C. Sharp, H. Schwarzbauer, T. Taylor, I. Rytina, M. Kalla, L. Zhang, V. Paxson *RFC 2960, 3309 - Stream Control Transmission Protocol,PROTOCOL SPECIFICATION* October 2000

[7] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson *RFC 3550, 5506, 5761, 6051, 6222, 7022, 7160, 7164 - Transport Protocol for Real-Time Applications PROTOCOL SPECIFICATION*July 2003

[8] David L. Mills *RFC 1305 - Network Time Protocol , PROTOCOL SPECIFICATION*March 1992

[9] Jeremy Elson, Lewis Girod, Deborah Estrin *Fine-Grained Network Time Synchronization using Reference Broadcast*Department of Computer Science, University of California,Los Angeles

[10] Saurabh Ganeriwal, Ram Kumar, Mani B. Srivastava *Timing-sync Protocol for Sensor Networks*Networked and Embedded Systems Lab, University of California, Los Angeles

[11] Miklós Maróti, Branislav Kusy, Gyula Simon, Ákos Lédeczi *The Flooding Time Synchronization Protocol*Institute for Software Integrated Systems, Vanderbilt University 2015

[12] Wenli Liu *Time Synchronization in Wirelees Sensor Networks* School of Engineering University of Applied Sciences May 2011

[13] Paul Kryzanowski *Clock Synchronization: Distributed System*Rutgers University 2016

[14] *IEEE 1588 Precise Time Protocol: The New Standard in Time Synchronization* Time. Symmetricom, Inc., pp. 1–5,2005.

[15] *"DLNA." [Online]. Available: https://www.dlna.org/*

[16] *"AirPlay." [Online]. Available:https://developer.apple.com/airplay/*

[17] *"Wisa." [Online].Available:http://www.wisaassociation.org.*

[18] *"Sonos." [Online]. Available: http:/www.sonos.com/en-eu.*

[19] Irina Aldoshina *Basics of Psychoacoustics*

[20] Christoffer Lauri, Johan Malmgren *Synchronization of streamed audio between multiple playback devices over an unmanaged IP network*Department of Electrical and Information Technology Lund University, September 2015

[21] Yonghwan Bang, Jongpil Hant, Kyusang Lee, Jongwon Yoon, Jinoo Joung, Sungbo Yang, and June-Koo Kevin Rhee *Wireless Network Synchronization for Multichannel Multimedia Services*Computer Science., Sangmyung University, Seoul, S.Korea. Feb.15-18,2009

[22] Paulo Ricardo Ferreira Alves *Sincronização áudio em sistemas de alta fidelidade sem fios*Escola de Engenharia da Universidade do Minho, 2014

[23] S. Bhattacharya, A.Dutta, R. Dasgupta *A Group Synchronization Algorithm for VoIP conferencing* Dept. of I.T. NIT Durgapur India

[24] Dr. Stephen Brown, Jorge Oliver *Inter-channel Synchronization for transmiting live audio streams over digital radio links* Computer Science, Maynooth University

[25] Fernando Boronat, Jaime Lloret, Miguel Garcıa *Multimedia group and inter-stream synchronization techniques: A comparative study* IGIC Institute, Valencia, Spain

[26] Sue B. Moon, Paul Skelly, Don Towsley *Estimation and Removal of Clock Skew from Network Delay Measurements* Departament of Computer Science, University of Massachusetts

[27] Dominique Fober, Yann Orlarey, Stephane Letz *Real Time Clock Skew Estimation over Network Delays* Computer Music Research Laboratory, Lyon, 2005

[28] *"ALSA" [Online]. Available: https://www.alsa-project.org*