# Investigating Low-Cost Platforms for Ultra-Low-Latency Audio Streaming in Networked Music Performances

Leigh Murray

Master's programme in Music, Communication and Technology

Department of Music
Norwegian University of
Science and Technology

Department of Musicology
University
of Oslo

May 2022

# Abstract

The use of Network Music Performance (NMP) technologies has grown considerably over the last three years, due largely to the mandatory social distancing and lockdown rules imposed during the COVID19 pandemic.  While the development of software based NMP platforms is steadily progressing, the vast majority rely on general purpose operating systems which are not optimised for ultra-low-latency high quality audio. Additionally, the hardware requirements can easily result in NMP systems costing thousands of dollars. This thesis finds that low-cost platforms for ultra-low-latency audio streaming in Network Music Performances are not only possible but are capable of providing lower-latency audio at a higher quality than existing high end NMP platforms.

## Acknowledgments

Thanks to Alena Clim and Stefano Fasciani for their feedback and supervision and Christian Werner for kindly lending me the UNISON/RemJams prototypes.

## Contents

# 1 Introduction

Musicians like to play together but due to distance, cost, or pandemics meeting in person is not always a viable option. This has led many musicians and researchers to look to the internet to provide a solution that allows musicians to play together when they can't be physically present. That solution is provided by Network Music Performance (NMP) platforms which enable high-quality low-latency audio streaming between two or more participants in realtime. Many NMP software platforms are free and open source but require high-performance or specialised hardware in order to achieve sufficiently low latencies with high quality audio resulting in high costs for the user. Even with high-performance and specialised hardware, performance bottlenecks are often encountered due to general purpose operating systems like Windows not being suited to ultra-low-latency audio applications. This thesis sets out to investigate low-cost platforms for ultra-low-latency audio streaming in network music performances.

This thesis starts with an investigation into the current literature of NMP platforms to establish the requirements for such a platform to be considered state of the art. These requirements are then used to develop a low-cost prototype which is evaluated based on its ability to reach or exceed these requirement. Finally, recommendations are given for future investigation and development of low-cost platforms for ultra-low-latency audio streaming in NMP.

## 1.1 Personal Motivation

While studying Music, Communication and Technology at the University of Oslo (UiO) I have had the opportunity to study and experiment with a wide range of Network Music Performance technologies. Through collaboration with students at UiO in Oslo and the Norwegian University of Science and Technology (NTNU) in Trondheim, I have tested, analysed and live-streamed network music performances using a range of free open-source consumer solutions as well as expensive high-end professional solutions typically only available to academic institutions.

Coming from a professional background in software development, I grew frustrated with the existing solutions; the software solutions were hindered by the underlying operating systems and hardware; the technologies were overly complicated for the end user; the hardware platforms were expensive, and the professional solutions were prohibitively expensive for the average consumer and even required highly provisioned WANs.

While undertaking a semester abroad at Aalborg University in Copenhagen I prototyped a microcontroller-based development board for digital music instruments. Microcontrollers don't have an operating system and are relatively cheap and powerful. I wanted to investigate the current consumer and state of the art devices for network music performance and determine the feasibility of a low-cost platform meeting the requirements set by these existing technologies.

## 1.2 Research Aims and Objects

### 1.2.1 Aim

This thesis aims to investigate low-cost platforms for ultra-low-latency audio streaming in Network Music Performances (NMP).

### 1.2.2 Objectives and Thesis Structure

#### 1.2.2.1 Literature Review

The first objective involves the review of the relevant literature regarding Network Music Performances and the technologies used to facilitate them. Through analysis of the existing platforms and their limitations, a set of requirements for a low-cost NMP platform are developed.

#### 1.2.2.2 Platform Selection and Development

The second objective involves the design of a low-cost audio streaming platform and evaluating its suitability as a network music performance device against the requirements obtained from existing state of the art NMP platforms in the literature review.

#### 1.2.2.3 Platform Evaluation

The third object is to evaluate the selected platform's suitability as a low-cost audio streaming device for NMP based on the various requirements formed by the literature review.

#### 1.2.2.4 Recommendations

The final objective is to provide a recommendation for future development of a low-cost audio streaming platform based on the findings of the evaluated platform.

# 2 Literature Review

Chapter 2.1 investigates the needs and core concepts of NMP followed by chapter 2.2 which investigates the current state of the art technologies that are used to provide a high quality NMP experience.

## 2.1 Network Music Performance

When musicians perform together, they typically do so in the presence of one another in the same physical location. Students and teachers meet physically for lessons and masterclasses, band members travel to a rehearsal room, collaborators meet in studios, and musicians and audiences attend the same venues for performances. Considerable time, cost and effort can be associated with facilitating these interactions, which come in the form of commuting for students, flights and accommodation for touring musicians, transporting of equipment. This, along with the pressure to have a reduced carbon footprint has resulted in many researchers, musicians and organisations investigating cheaper, more convenient and less environmentally harmful alternatives (Davies, 2015). The social distancing laws introduced during the COVID19 pandemic prohibited many musicians from meeting physically which left online collaboration as the only alternative (Bosi, Servetti, Chafe, & Rottondi, 2021).

Network music performance (NMP) is the attempt to bring real-time musical collaboration online, allowing musicians in two or more geographically disparate locations to perform together as if they were in the same room. This is done by capturing audio and video streams of the musicians at each location and transmitting these streams between the musicians, enabling them to see and hear each other perform in real-time. The experience is similar to that provided by video conferencing platforms such as Zoom or Skype, and indeed these technologies are still present in many NMP setups (Cook, 2015), however the nature of real-time music interaction necessitates specific hardware and purpose-built technologies to meet the differing latency and quality requirements of NMP, especially in regards to audio.

Network music performance is not an attempt to replace regular music performance, but a complementary solution that greatly increases opportunities for rehearsal, performance, education and experimentation.

The following chapters review the current literature detailing the concepts and technologies involved in creating a successful NMP, beginning with the key contributors to a high quality NMP experience, which are audio quality and low latency (Rottondi, 2016).

### 2.1.1 Latency

The single most important requirement for NMP is low latency audio (Rottondi, 2016) (Rofe & Reuben, 2017) (Tsioutas & Xylomenos, On the Impact of Audio Characteristics to the Quality of Musicians' Experience in Network Music Performance, 2021), which is typically measured as the time it takes for the audio created by one performer to be heard by the other(s), and has been referred to as the Over-all One-way Source-to-Ear delay (OOSE) (Rottondi, 2016) or Mouth to Ear (M2E) delay (Tsioutas, Xylomenos, & Doumanis, 2019). This is illustrated in Figure 2.1.1.

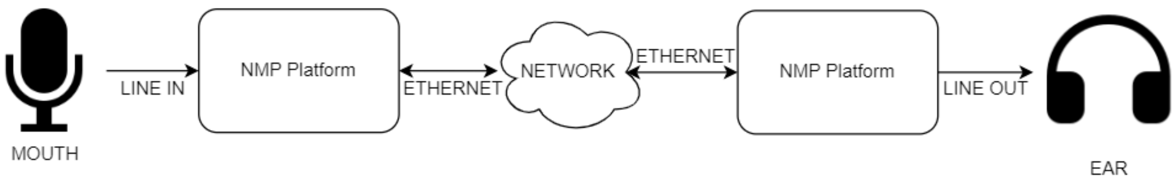



*Figure 2.1.1 Audio/Data Flow for Mouth to Ear (M2E) Delay*

The requirement for low latency audio is due to the necessity for musicians to synchronise their performances such that they are able to play in time with each other and maintain a steady tempo. In order to achieve this synchronicity, OOSE ideally needs to be between 10 and 25 milliseconds, referred to as the Ensemble Performance Threshold (EPT) (Carôt, Kramer, & Schuller, 2006) or Latency Tolerance Threshold (Rottondi, 2016). Although particular genres, slower tempos and more experience can enable musicians to maintain steady tempo at higher latencies (Bouillot, 2007), 25 milliseconds is generally the target as it is the maximum value where the delay is not perceived by the performers (Carôt, 2009) (Carôt, Kramer, & Schuller, 2006) (Rottondi, 2016).

A similar method of latency measurement is known as My Mouth to My Ear (MM2ME) and has been presented as a more appropriate measurement for NMP as it aims to replicate the latency between a musician creating a sound and then hearing a remote musician's response to that sound (Tsioutas, Xylomenos, & Doumanis, 2019). This is illustrated in Figure 2.1.2.

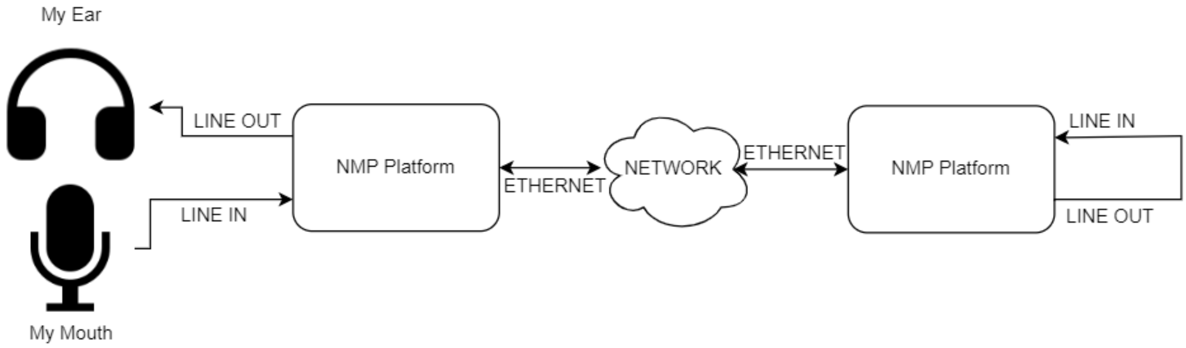



*Figure 2.1.2 Audio/Data Flow for My Mouth to My Ear (MM2ME)*

In a MM2ME setup, a sound is captured on the LINE IN of the local NMP platform and sent to the remote NMP platform. The remote NMP platform then loops the audio through its LINE OUT and LINE IN before sending the audio back to the local NMP platform, which finally sends the audio to its LINE OUT. The delay between generating a signal on the local LINE IN and "hearing" the signal back on the local LINE OUT is the MM2ME latency. In an ideal setup, MM2ME latency will simply be double the M2E latency, and vice versa. The main advantage of measuring MM2ME is based on real world NMP setups, where the local and remote devices are potentially several thousand kilometres apart, making it very difficult to accurately measure the delay between generating a sound locally and hearing the sound remotely as is required by M2E.

When performing together physically, the largest contributor to audio delay is the speed of sound in air as sound waves traverse the air between performers. When performing virtually, there are many more factors that contribute to latency. These are illustrated in Figure 2.1.3.

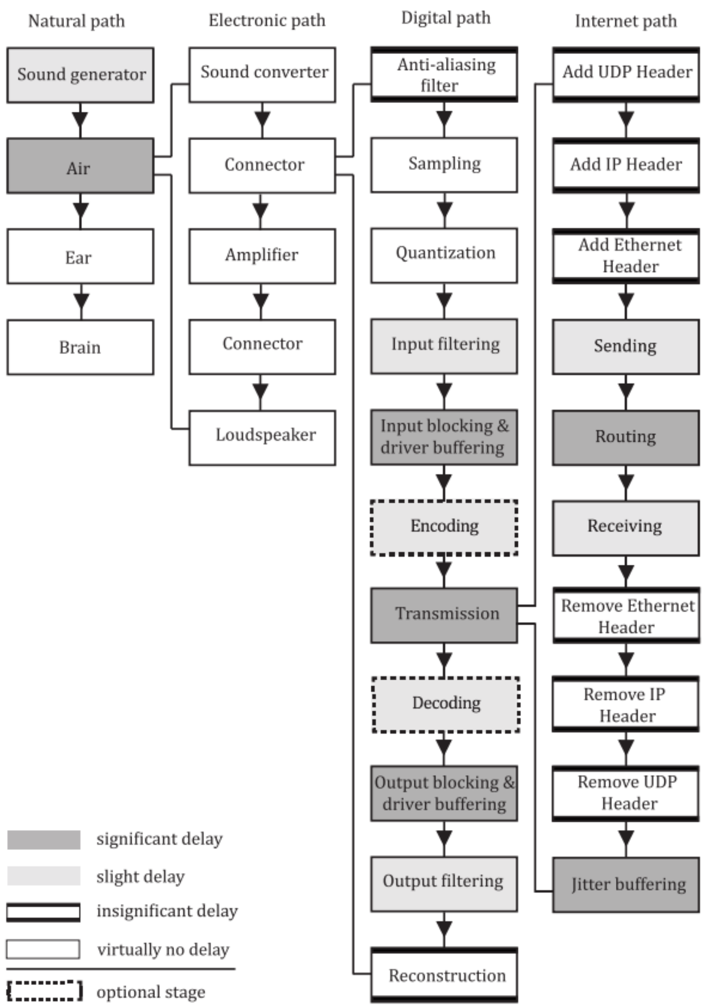



*Figure 2.1.3 Total signal path of an audio signal – with respect to the final performance latency threshold of approximately 25 ms, the stages are marked with significant delay, slight delay, insignificant delay or virtually non-existent delay. Depending on the applied hardware and the actual physical distance, significant delays reside in the domain of several milliseconds. Slight delays typically range between approximately 100 µs and a maximum of 1 ms. Values below 100 µs are considered insignificant (Carôt, Hoene, Busse, & Kuhr, 2020)*

As shown in Figure 2.1.3, the slight and significate contributors to NMP delay are:

#### 2.1.1.1 Air

The speed of sound in air is approximately 343 metres per second, resulting in 2.9 milliseconds of audio delay per meter travelled. The 25ms EPS is equivalent to two musicians performing together with a physical separation of 8.6 metres. In NMP setups, the delay introduced by the sound travelling through air can be minimised by placing the microphone close to the performer or having a direct input from an electronic instrument, and having the performers wear headphones, which also contributes to the quality of the audio stream by preventing feedback loops.

#### 2.1.1.2 Input filtering

When audio is converted from analogue to digital is often undergoes filtering.  The most common filtering on input is a low pass filter to remove frequencies outside the range of human hearing, however audio interfaces can also implement digital filters and effects that can increase delay (Rottondi, 2016).

#### 2.1.1.3 Input blocking & driver buffering

When digital samples are transferred between an audio codec/interface and other hardware/software, they are sent in batches, known as audio buffers. The audio interface must wait until it has captured enough samples to fill a buffer before it can send it to the external hardware/software, and this in turn results in a delay in the audio stream ranging from less than 100 microseconds to over 3 milliseconds, depending on the sample rate and buffer size. This is explained more in chapter 2.1.3.

#### 2.1.1.4 Transmission

Transmission involves the processes of Sending, Routing, Receiving and Jitter Buffering. Sending, Receiving and Jitter Buffering are dependent on the hardware and software of the NMP, however Routing on WANs is dependent on external internet service providers and the greater internet infrastructure which end users typically have no control over, and is the biggest contributor to latency and jitter over long distances.

##### 2.1.1.4.1 Sending

Sending audio streams in NMP platforms typically involves the creation of a data packet containing a sample buffer, and possibly configuration or control data, and sending this data packet to the ethernet hardware. Some NMP platforms can also include previously sent audio buffers along with the current buffer for redundancy, which requires more bandwidth and increases transmission time. The input sample buffer can also be split into multiple data packets to support low bandwidth networks (Bouillot & Cooperstock, 2009).

##### 2.1.1.4.2 Routing

The time it takes for data to make a return journey from a source to a destination is referred to as the round-trip-time (RTT). Drioli & Buso (2013) stated that the RTT on Local Area Networks (LAN) is less than 1ms and approximately 1ms per 100 kilometres on WANs, though it is not clear whether they were referring to the highly provisioned academic WANs available at their university or regular WANs that general internet users have access to. Wireless LANs are considered unsuitable for NMP as they introduce too much jitter (Bosi, Servetti, Chafe, & Rottondi, 2021).

##### 2.1.1.4.3 Receiving

The received data is placed into a buffer until the entire packet it received, at which point it is available to the host application for further processing.

##### 2.1.1.4.4 Jitter Buffering

WANs are highly susceptible to jitter, meaning packets are not always received at the consistent intervals at which they are sent and can arrive in a different order to which they were sent. Smaller packets, which are associated with smaller buffer sizes and lower latency audio, are most susceptible to this phenomenon. NMP platforms can combat this by creating a jitter buffer, which contains two or more sample buffers, the exact amount of which can be tuned depending on the level of jitter experienced. This provides the possibility of re-ordering buffers that arrive out of sequence and attempts to create a buffer of audio samples large enough that the generated audio stream is unaffected by packet delays. This of course results in more latency so a trade-off is often made where sufficiently delayed packets are discarded at the expense of experiencing audio dropouts, in order to keep audio latency to a minimum.

##### 2.1.1.5 Output blocking & driver buffering

As samples are passed back to the audio codec/interface for output, they are done so using a sample buffer the same size as the input buffer.

##### 2.1.1.6 Output filtering

Output filtering, like input filtering can commonly involve a frequency filter, such as a high pass filter to remove any direct current (DC) component from the signal.

### 2.1.2 Audio Quality

Audio quality has been reported as the most important factor contributing to the audience's level of enjoyment of an NMP (Davies, 2015). The two primary factors that contribute to the quality of an uncompressed digital audio signal are sample rate and bit depth. Together they also determine the bandwidth required for the audio stream and required processing power.

For audio to be transferred over a network, the sound waves must be captured and converted to a digital signal. This conversion is processed by an audio codec, is performed by sampling the analogue signal at regular intervals and converting these voltages to digital values. The frequency in which the samples are taken is referred to as the sample rate and is measured in Hertz (samples per second). The range of digital values available for each sample, that is, how accurately each digital sample represents the analogue signal, is the bit depth.

Common audio sample rates for NMP are 44,100 Hz, 48,000 Hz and occasionally 96,000 Hz, and common bit depths are 16-bit, 24-bit and 32-bit. In order to provide an NMP experience equivalent to a regular rehearsal, a sample rate of 48kHz and bit depth of 16 is required (Carôt, Kramer, & Schuller, 2006). A sample rate of 96000 Hz results in twice as many samples, and thus twice as much digital data must be handled in the same time period in comparison to a sample rate of 48,000 Hz.

While audio codecs process these samples individually, the high frequency at which they are sampled cannot be maintained by general purpose computer architectures (Rottondi, 2016) and wide area networks (WAN) like the internet. Therefore, to reduce the processing frequency, the samples are processed in groups, called sample buffers.

### 2.1.3 Sample Buffers

Samples are transferred between hardware and software applications both locally and over network connections using sample buffers. Although handling each sample individually would results in the lowest latency, this is generally not achievable due to scheduling and interrupt overheads of modern operating systems, and the overheads and inconsistent routing within WAN architectures. When a buffer size is too small, audio dropouts can occur as the application is unable to both retrieve and send samples to and from the audio codec at the required frequency. Compromises must be made to ensure a consistent stream of audio, which is easier with larger buffers, while achieving an appropriately low level of latency, which requires smaller buffers.

For modern operating systems, it is often difficult to achieve buffer sizes lower than 128 samples without high end hardware and optimised drivers (Carôt & Werner, 2009), although buffers as low as 64 and 32 are achievable (Carôt, 2009). At the standard sample rate of 48kHz it takes 1.33 milliseconds to fill a standard buffer of 64 samples. Since this has to occur both when receiving samples from a codec/audio interface and sending samples to a codec/audio interface this equals a total of 2.66ms for input and output.  The latency for common buffer sizes and sample rates are illustrated in Table 2.1.1.

| | | Buffer Size | | | | |
|---|---|---|---|---|---|---|
| | | **128** | **64** | **32** | **16** | **8** |
| **Sample Rate** | **44100** | 2.90 | 1.45 | 0.73 | 0.36 | 0.18 |
| | **48000** | 2.67 | 1.33 | 0.67 | 0.33 | 0.17 |
| | **96000** | 1.33 | 0.67 | 0.33 | 0.17 | 0.08 |

*Table 2.1.1 Time required in milliseconds to fill a buffer at the associated Buffer Size and Sample Rates*

Since a codec must wait for the audio to be captured in order to fill the buffer, a buffer contains samples for a segment of audio of the same duration of milliseconds.  For example, in order to fill a buffer with 3 milliseconds of audio you must wait 3 milliseconds for that audio to be captured. All systems running at the same sample rate and buffer size will take a least this long to process the audio, both on input and output, and thus achieving latencies below these values are impossible when audio buffers are used.

Although audio codecs are configured to run at the predefined sample rates stated above, small imprecisions exist with these timings between devices, causes different devices to create slightly more or slightly less samples than each other due to this clock drift.  As a result, one device will both fill its input buffer and play its output buffer slightly quicker than the other.  This is a common issue and often goes unsolved in current NMP platforms (Rottondi, 2016).

### 2.1.4 Summary

A review of the literature has illustrated the core requirements for a successful NMP are high quality and low latency audio.  Investigating the core concepts and technologies involved in NMP has shown that addressing these two requirements is a constant compromise, as improving one often negatively impacts the other, resulting in a need to prioritise one of these two requirements at various stages.

Now that the core concepts and requirements of NMP are understood, this thesis investigates the current state of the art software and hardware used in existing NMP platforms.

## 2.2 NMP Platforms

Network Music Performance Platforms consist of software and hardware components.

### 2.2.1 Hardware

Based on the concepts outlined in the previous chapter, the minimum hardware requirements for an NMP audio platform are:

1. Audio Codec for audio input/output and conversion between analogue and digital.
2. Ethernet chip for transmitting/receiving digital audio.
3. Hardware to run an NMP software application, managing system configuration and providing connectivity between the audio codec and ethernet connection for buffering and transferring audio samples, typically requiring a CPU, RAM and persistent storage.

Personal computers running Windows, MacOS and Linux generally fit these requirements and are the most common platform for NMP. External audio interfaces or internal sound cards can be added to a PC in order to provide low-latency, high quality audio that is not typically provided by on-board audio codecs, particularly on Windows based systems that require the audio interface to support the ASIO API in order to provide sufficiently low latencies and buffer sizes.

Single board computers such as the Raspberry Pi are a suitable lower-cost alternative that also meet these requirements.  Though the onboard codec is generally not sufficient, USB audio interfaces and audio expansion boards that attach to the exposed header sockets such as those by HiFiBerry, can be added which provide ultra-low-latency audio quality up to 192kHz sample rates.  The BeagleBone Black is also capable of meeting these requirements however the popular low latency audio cape known as Bela provides a sample rate of 44.1kHz which is just shy of 48kHz requirement for NMP.

Some hardware devices have also been developed specifically for the purposes of NMP. The Elk LIVE Bridge is a portable that runs Elk OS, a specially developed open source operating system designed for low latency audio and compatible with the Raspberry Pi.

### 2.2.2 Software

NMP software is becoming increasingly prevalent ranging from free command-line executables to cloud-based monthly subscription services offered through web interfaces.  A review of 18 of the current state-of-the-art NMP software frameworks (Rottondi, 2016) found that the different technologies varied very little in their implementations (Carôt, Hoene, Busse, & Kuhr, 2020). To provide a better understanding of a software NMP architecture, two of these software applications, JackTrip and LOLA, were investigated.  JackTrip was chosen as it is free and open source and available as a comparable low-cost NMP platform known as JackTrip Bridge devices. LOLA was chosen as it is used in over 140 academic institutions and is indicative of a professional high-end low-latency audio and video NMP solution.

#### *2.2.2.1 JackTrip*

JackTrip is an open-source audio streaming software and transmission protocol for low latency streaming developed by the Center for Computer Research in Music and Acoustics (CCRMA) in 2007 (Lopez-Lezcano & Wilkerson, 2007).  Originally developed for Linux, it is now available on Windows, Linux, MacOS and FreeBSD.  JackTrip is also sold as two hardware NMP platforms known as the JackTrip Analog Bridge and JackTrip Digital Bridge. These devices are Raspberry Pies that come with JackTrip pre-installed. The Digital Bridge requires a USB microphone or audio interface whereas the Analog Bridge includes a HiFiBerry DAC+ ADC Pro expansion board, providing CD quality stereo analogue input and output (JackTrip Labs, 2022).

The JackTrip protocol uses UDP which is a best effort protocol that prioritises low bandwidth and latency but provides no mechanism to determine whether a packet was successfully delivered.  To overcome this limitation and handle the possibility of packets arriving in a different order to which they are sent, JackTrip provides a configurable latency buffer and adds its own 128bit (16 byte) header to the data portion of the UDP packet that includes the timestamp and sequence number of the packet (Chafe & Caceres, 2009).  This header structure is shown in Table 2.2.1.

| Bits | Name |
|---|---|
| 64 | Timestamp |
| 16 | Sequence Number |
| 16 | Buffer Size |
| 8 | Sampling Rate |
| 8 | Bit Resolution |
| 8 | Number of Input Channels |
| 8 | Number of Output Channels |

*Table 2.2.1 JackTrip UDP Header Data Structure and Sizes (Caceres, 2022)*

Since this header is sent with every packet it provides additional overhead which negatively impacts transmission times and required bandwidth, both of which increase as the selected buffer size decreases. Only the timestamp and sequence number differ between packets and for this reason, many competing protocols have opted for alternate methods of transmitting the other configuration/control data (Rottondi, 2016).

A review of four different low latency streaming platforms, JackTrip was shown to have the lowest latency but also the highest number of audio glitches (Bouillot & Cooperstock, 2009). In their development of a competitive audio streaming software for NMP, Tsioutas, Xylomenos, & Doumanis (2019) evaluated their new platform against JackTrip, highlighting its acceptance and ongoing use as state of the art NMP software.

#### 2.2.2.2 LOLA

LOLA (low latency audio visual streaming system) is a proprietary software platform capable of streaming uncompressed audio and video with OOSE latencies as low as 5ms (Davies, 2015). Developed by Conservatorio di Musica "Giuseppe Tartini" in Italy, the software is proprietary but free to use for academic, educational and non-profit users and shareware in all other cases. It is one of the more common platforms within educational institutions with over 160 installations.

In order to guarantee optimal performance of the LOLA software, a purpose built Windows PC is necessary with hardware requirements specified by the LOLA developers. The developers of LOLA also mention their fundamental requirement for the system to be portable (Drioli & Buso, 2013) however as explained however a full setup is more akin to a small broadcast studio (Conservatorio di musica G. Tartini, 2022). External latency is minimised by limiting the use of LOLA systems to highly provisioned WAN networks typically only available at educational and research facilities.

### 2.2.3 Cost

The costs involved in establishing a NMP can vary considerably, especially when low latency video is required. For example, although the creators of LOLA stated a fundamental requirement to be a low-cost portable platform (Drioli & Buso, 2013), one full LOLA setup including the computer, audio interface, dedicated graphics card, monitors, low latency cameras, capture cards, studio monitor speakers, microphones, projectors, etc. costs over

12,000 euro (Conservatorio di musica G. Tartini, 2022). Since one complete LOLA setup is required per location this results in costs of over 24,000 euro for a full NMP setup. This cost, along with their requirement for a highly provisioned internet backbone, means this type of NMP setup is only available to well-resourced universities (Smith, Moir, Ferguson, & Davies, 2020). This has raised criticism that organising an NMP can be more expensive than the costs involved in having the musicians meet for a regular non-distributed performance (Davies, 2015). The LOLA software, however, is free/shareware.

On the other hand, if a musician already owns the necessary hardware such as a computer with ethernet port and an appropriate audio interface, an NMP can be established with no additional cost by utilising free software like JackTrip.

Laptops and desktops can cost from hundreds to thousands of dollars. MacBooks and the majority of Windows laptops don't contain ethernet ports, so a USB3 or USB-C ethernet adapter is required with prices ranging from $12-$40. Windows on-board soundcards are generally unable to meet the latency and quality requirements for NMP and MacBooks are limited to a single combined headphone/microphone port, so an external audio interface is generally required. The cheapest suitable audio interface is potentially the USB Behringer U-CONTROL UCA222 for around $25. Depending on the performance of the laptop, the user may still be limited to sample buffers of 128, 256 or 512.

Single board computers like the Raspberry Pi 3 can be used, costing $35 dollars, plus memory card, enclosure and power supply and require a certain level of technical knowledge to configure. This would ideally be accompanied with a HiFiBerry DAC+ ADC costing ~$50 for the lowest latency (Jack, Moro, & McPherson, 2016), but the previously mentioned $25 USB Behringer U-CONTROL UCA222 could also be used. The official JackTrip Raspberry Pi 4 products, the Digital Bridge and Analog Bridge, are available for $150 and $250 respectively.

The standalone platform Elk LIVE retails for $380 and a yearly subscription of $179 is required to play together with this device.

### 2.3 Summary

This literature review has presented the key components required for the implementation of a successful NMP platform with regards to low latency and audio quality. It has also illustrated the current cost for a standalone NMP platform ranges from $150 to $380+ dollars.

The resulting chapters of this thesis look at developing a prototype low-cost low-latency audio streaming platform which is subsequently tested against the requirements of NMP platforms outlined in this review.

# 3 Platform Selection and Development

This chapter details the developments of a low-cost low-latency audio streaming platform (LASP) prototype for NMP. Chapter 3.1 uses the findings from the literature review to outline the software and hardware requirements and benchmarks the device should achieve in order to be considered competitive with the current state of the art NMP platforms. In chapter 3.2 the hardware and software components selected for the development of this platform are described and chapter 3.3 describes the further development that was required to bring these hardware and software components together into a cohesive NMP platform.

## 3.1 Requirements

The platform must be able to support 48,000 Hz/16 bit audio as this was determined to be the minimum requirement for a realistic rehearsal experience. 96,000 Hz would make the system exceed current high-end NMP platforms such as LOLA.

Buffer sizes of 64 and 128 will make the device compatible with and comparable to most current NMP platforms. Supporting a buffer size of 32 will be comparable to a high end NMP platform such as LOLA. Buffer sizes of 8 and 16 would exceed current high-end NMP platforms that are limited by their underlying operating systems.

In order to be integrated with existing NMP platforms and provide the lowest latency, UDP should be used for transferring packets between.

The following processes should only introduce slight delay (100 microseconds - 1.0 millisecond) or significant delay (2-4 milliseconds).

- Slight Delay (100 microseconds – 1 millisecond)
    - Input filtering
    - Encoding
    - Sending
    - Receiving
    - Decoding
    - Output filtering
- Significant Delay (2-4 milliseconds)
    - Input blocking & driver buffering
    - Transmission
        - Routing
        - Jitter buffering
    - Output blocking & driver buffering

All other processes should cause either insignificant delay (< 100 microseconds) or virtually no delay.

The platform must also be low-cost. The current low cost alternatives are the JackTrip Analog Bridge at $250 and the Digital Bridge at $150. Modifying a laptop to be NMP compatible with a USB Behringer U-CONTROL UCA222 and a USB3 Ethernet port would cost around $40. Based on these values, the required cost of this device is to be less than the Digital Bridge at $150 and as close to the $40 necessary to convert a laptop into an NMP as possible.

## 3.2 Selection

To reach the ambitious cost target for the NMP platform, the hardware resources must be utilised efficiently. The presence of an operating system in NMP platforms necessities more CPU power, RAM and storage, resulting in more expensive hardware. The operating system is also what limits most NMP platforms from achieving their lowest buffer sizes and highest sample rates. For these reasons, the decision was made to develop a microcontroller-based

NMP as they do not require an operating system and the hardware can be more efficiently utilised.

Microcontrollers have historically required significant effort for designing, assembling and coding however with the advent of the Arduino development boards and associated Arduino IDE, many microcontroller development boards and free code libraries exist making microcontroller prototypes relatively quick to develop.  An Arduino compatible development board was therefore required that met the requirements of an NMP hardware platform.

The Teensy 4.1 with associated audio shield and ethernet port was ultimately decided on for many factors including its price/performance ratio, proven track record in the digital music instrument space and possibilities for future development which are discussed in the following chapters.

## 3.3 Development

### 3.3.1 Teensy

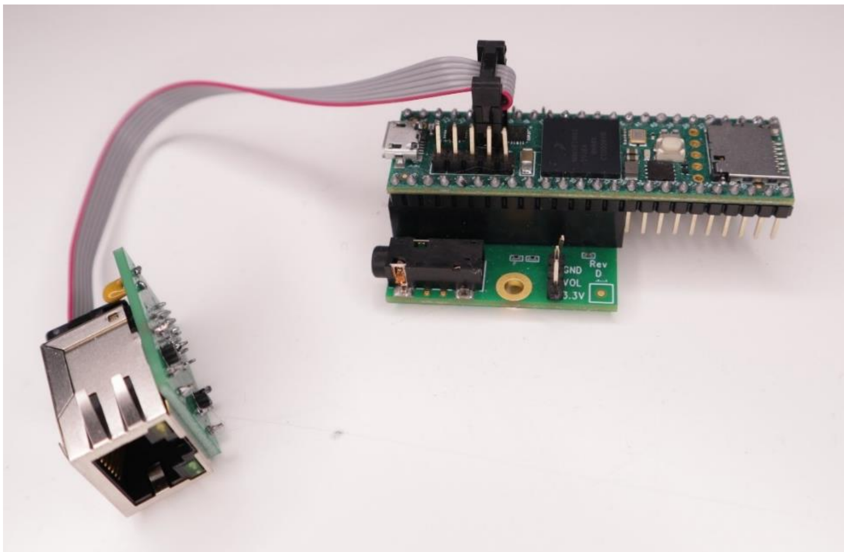


*Figure 3.3.1 Implementation of a Low-cost Audio Streaming Platform - A Teensy 4.1 with Audio Shield and Ethernet Kit*

Teensy is a range of Arduino compatible development boards which are popular due to their low cost, compact form factor and high performance in comparison to official Arduino board range.  With the high speed 32 bit ARM Cortex-M7 processor running at 600MHz and the ethernet and audio shields shown in Figure 3.3.1 detailed in the following chapters, the Teensy 4.1 is theoretically a very capable NMP streaming device. In addition, the Teensy 4.1 has been proven to be able to successfully address the issue of clock drift in the development of the UNISON (Werner & Kraneis, 2021). The total hardware cost for the Teensy 4.1 NMP prototype was $44.50, which is significantly lower than the JackTrip Digital Bridge at $150, and close enough to the target price of $40 to consider this goal achieved.  The total cost of the prototype consists of $26.85 for the Teensy 4.1 microcontroller board, $13.75 for the audio shield and $3.90 for the ethernet kit.

The Teensy has been used in low latency devices such as the open source TYMPAN hearing aid device (Tympan, 2022). Through the micro USB port on the Teensy it can be configured to act as an external audio interface when connected to a PC, supporting a transfer rate of 480Mb/s.  In this way, rather than using the analogue line in and line out ports, it can transfer digital audio directly to and from the PC while transmitting/receiving audio data directly via its ethernet port. The Teensy Audio GUI, described in 3.3.3, makes it possible to use this USB audio interface functionality alongside the regular line input/output with full control over gain, patching and mixing of each device's audio channels.

The Teensy 4.1 has an onboard ethernet chip supporting 100Mb/s and is accessed by connecting an external ethernet port to the ethernet headers.

### 3.3.2 Teensy Audio Shield

The Teensy Audio Shield has an NXP SGTL5000 audio codec which is capable of sample rates from 8kHz to 96kHz (NXP Semiconductors, 2022).

Users testing the SGTL5000 found that the line input causes an inverting of the signal's phase due to an inverting preamp which may cause unexpected behaviour in very specific use cases but can be negated by using a mixer with a -1 value in the Teensy Audio GUI or by inverting the signal values in code.

### 3.3.3 Teensy Audio GUI

The Teensy Audio GUI, shown in Figure 3.3.2, is a browser based application that provides the ability to configure and visualise the internal audio routing used by the Teensy's application code.

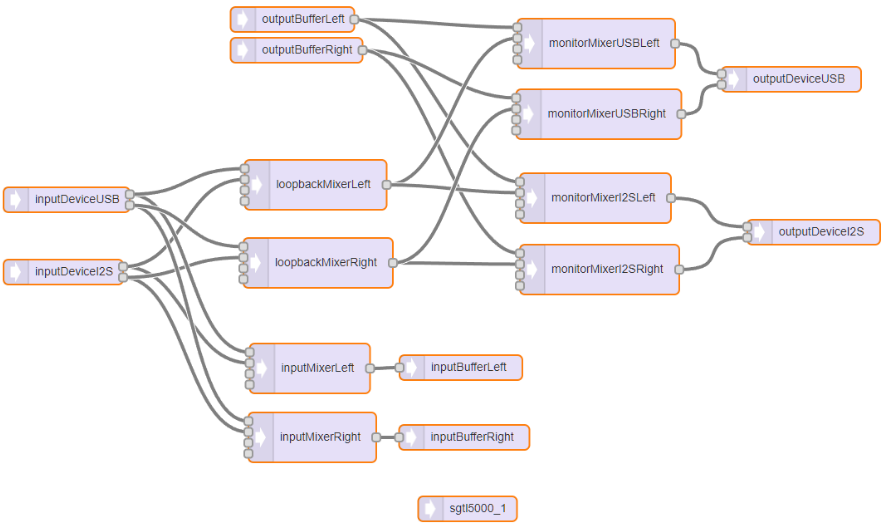



*Figure 3.3.2 Teensy Audio GUI*

Images from this tool taken during the development of Teensy application code will be displayed in the methods of the following chapter to illustrate the internal routing used for each test.

### 3.3.4 Application Code

All the code developed for these tests is available on GitHub (Murray, 2022). The ethernet library that is included with the Teeny Arduino IDE installer was found to be very inefficient at utilising the full bandwidth available, so the QNEthernet library by (Silverman, 2022) was used for tests involving ethernet connectivity. Similarly, a more efficient library was needed when reading analogue pin values on the Teensy measurement device, so the ADC library (Villanueva, 2022) was used.

# 4 Platform Evaluation

## 4.1 Overview

There were a total of eleven tests performed to evaluate the performance of the low-cost audio streaming platform (LASP). These tests were designed to determine the bandwidth, audio quality and latency of the LASP. The tests conclude with a WAN My Mouth to My Ear Delay test and an integration test using the state of the art NMP software, JackTrip. As the methodology used for each test varies, each test will include its own methodology, results and discussion sections in order provide information specific to each test, its intended outcomes and how it impacted or was influenced by the results from previous and subsequent tests. In general, each individual test configuration was automatically tested 100 times with a 1 second interval, and the average was presented as the results in this thesis.  The latency targets for the processes are presented in Table 4.1.1 for reference.

| Signal Path Stage | Latency Target |
|---|---|
| Input Filtering | 100 microseconds – 1 millisecond |
| Sending | 100 microseconds – 1 millisecond |
| Receiving | 100 microseconds – 1 millisecond |
| Output Filtering | 100 microseconds – 1 millisecond |
| Input blocking & driver Buffering | 2-4 milliseconds |
| Transmission (Total) | 2-4 milliseconds |
| - Routing | 2-4 milliseconds |
| - Jitter Buffering | 2-4 milliseconds |
| Output Blocking & Driver Buffering | 2-4 milliseconds |

*Table 4.1.1 Signal Path Stage Latency Targets*

## 4.2 i2s Passthrough

### 4.2.1 Overview

The purpose of this test was to determine the latency of the codec itself and presents the minimal latency achievable for the device when inputting and outputting audio. This configuration produces results for the following signal path stages; input filtering, output filtering, input blocking and output blocking. In this configuration there is no application buffer so the Teensy was unable to access the audio data; it was passed directly from the i2s input of the LASP to the i2s output.

### 4.2.2 Methodology

A Teensy with an audio shield was used as the measurement device. The role of the measurement device was to send a signal to the LASP and measure the time it took for the LASP to return the signal, which was the latency result.

The signal created by the Measurement Device was done by changing the voltage on pin 33 from LOW (0 volts) to HIGH (3.3 volts), a process that took less than 30 nanoseconds and thus did not impact the resulting latency measurements which are in the order of micro or milliseconds.  The measurement device was configured to run at 96kHz with a buffer of 4 samples, resulting in 41.67 microseconds per buffer. A buffer of 4 samples was used as it was the minimum achievable without making modifications to the audio library code of the Teensy.

### 4.2.3 Test Procedure

The measurement device would start a timer prior to immediately producing a signal on digital pin 33 which was connected to the LINE IN of the LASP. The measurement device would then continually check the samples obtained on its own LINE IN for the presence of this signal being returned by the LASP. A temporary record of the latency would be stored as soon as an audio buffer was registered as being available to the Measurement Device, and if the signal was present in any of the 4 samples within that buffer, that stored value would be reported as the latency of the test.

The physical setup for this test configuration is illustrated in Figure 4.2.1.

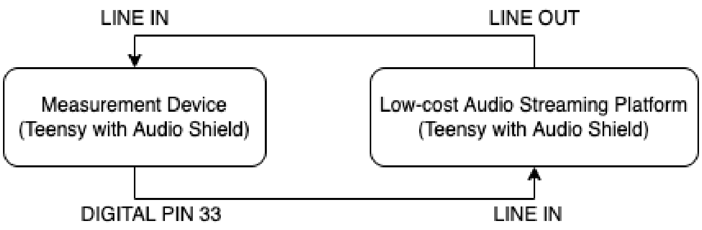



*Figure 4.2.1 Physical connectivity diagram of the Measurement Device and the Low-cost Audio Streaming Platform*

As the measurement device itself relies on an analogue to digital converter, audio codec and i2s transmission, the latency of the measurement device had to be determined with a loopback test. The configuration of this test can be seen in Figure 4.2.2.

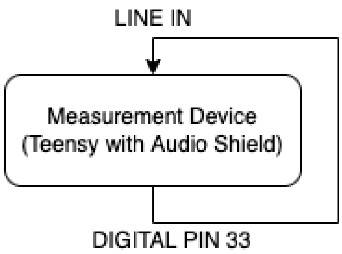



*Figure 4.2.2 Physical Connection Diagram of the Measurement Device in Loopback Configuration*

The latency for setting pin 33 to HIGH and receiving the signal on LINE IN was 162 microseconds, or 0.16 milliseconds and this latency was removed from the results of the tests that used this configuration.

The Teensy Audio GUI was used for routing the audio data within the LASP.


*Figure 4.2.3 Teensy Audio GUI Representation of Passing Through Audio Data from/to the Audio Shield*

Figure 4.2.3 shows that the LASP is receiving audio data from the *inputDeviceI2S*, which in this case is the Audio Shield and contains the audio data captured from its LINE IN. The left channel of this device is shown to be routed directly to the left channel of the *outputDeviceI2S*, which in this case is the same Audio Shield and this audio data is sent to the LINE OUT of the Audio Shield.

### 4.2.4 Results

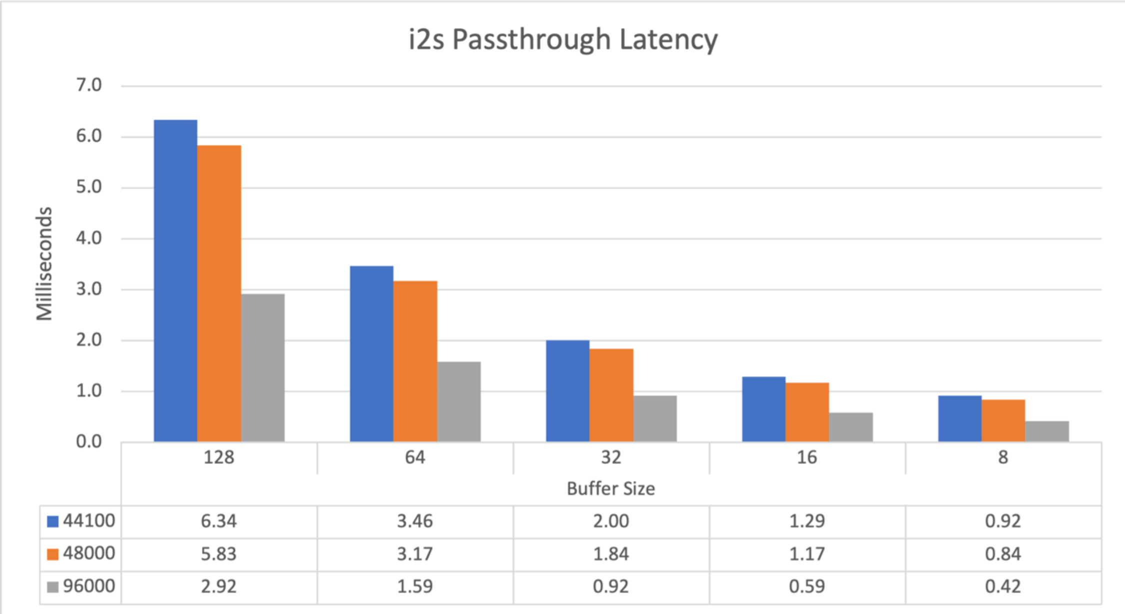



*Figure 4.2.4 Total RTT (milliseconds) when Passing i2s Input Directly to i2s Output*

### 4.2.5 Discussion

Figure 4.2.4 above shows the results of the i2s passthrough latency test and shows the total time between creating the signal on the input of the device and receiving the signal on the output of the device. These results illustrate the LASP's capability to not only meet the 32 sample buffer sizes used by high-end NMP platforms like LOLA but exceed them with buffer sizes of 16 and 8. Similarly, the LASP is able to operate at the legacy 44.1kHz, the required 48kHz, and 96kHz sample rates exceeding the high-end platform LOLA.

| | | Buffer Size | | | | |
|---|---|---|---|---|---|---|
| | | **128** | **64** | **32** | **16** | **8** |
| **Sample Rate** | **44100** | 0.53 | 0.56 | 0.55 | 0.57 | 0.56 |
| | **48000** | 0.50 | 0.50 | 0.50 | 0.50 | 0.50 |
| | **96000** | 0.25 | 0.25 | 0.25 | 0.25 | 0.25 |

*Table 4.2.1 LASP i2s Overhead Time in Milliseconds. The time required to fill the audio buffer twice (input and output) is removed from the Total i2s RTT in the Figure 4.2.4.*

Table 4.2.1 above shows the additional time the LASP introduces in the passthrough of the audio by removing the default durations for input and output blocking delays – equivalent to two times the buffer audio length shown in Table 2.1.1 – from the results in Figure 4.2.4. As illustrated, the i2s latency is only influenced by the sample rate and remains very consistent between buffer size. These latency values which include any input and output filtering, as well as possible additional delays from input and output blocking processes are well within the target ranges.

In a real-world scenario this i2s passthrough configuration could be used for a direct feedback loop enabling the local musician to monitor and hear his own audio with minimal latency.

## 4.3 Buffer Passthrough

### 4.3.1 Overview

The buffer passthrough test was designed to determine the latency associated with introducing an application audio buffer. This buffer makes the audio data available to the application code running on the LASP.

### 4.3.2 Methodology

The methodology for this test was based on the methodology for the i2s passthrough, described in 4.2.2, with the following modifications outline below.


*Figure 4.3.1 Teensy Audio GUI Representation of the Audio Data Routing for the Buffer Passthrough Test*

As shown in Figure 4.3.1 the audio data coming from the audio shield via *inputDeviceI2S* (Audio Shield LINE IN) is now stored in an input buffer which makes the data available to the LASP application code. The code on the LASP then copies the buffer samples directly from the *inputBuffer* to the *outputBuffer*, and the *outputBuffer* is then transmitted to the *outputDeviceI2S* (Audio Shield LINE OUT).

### 4.3.3 Results

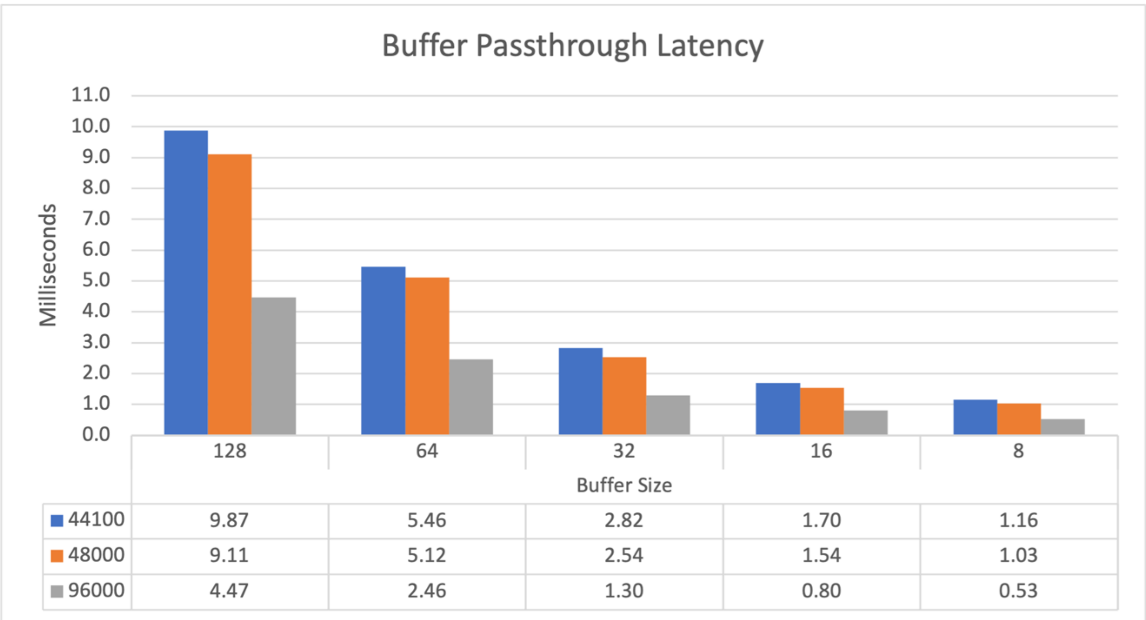



*Figure 4.3.2 Total RTT (milliseconds) when Copying Audio Data Between Input and Output Application Buffers*

### 4.3.4 Discussion

Figure 4.3.2 shows some cause from concern, especially at higher buffer sizes, where the latency values have increased by over 3 milliseconds compared to the i2s passthrough latency results. To determine the exact increase in latency caused by the application buffer, the default durations for input and output blocking delays – equivalent to two times the buffer audio length shown in Table 2.1.1 – are again removed from the results shown in Figure 4.3.2. The latencies with these blocking delays removed are shown in Table 4.3.1.

| | | Buffer Size | | | | |
|---|---|---|---|---|---|---|
| | | **128** | **64** | **32** | **16** | **8** |
| **Sample Rate** | **44100** | 4.07 | 2.55 | 1.37 | 0.98 | 0.80 |
| | **48000** | 3.77 | 2.45 | 1.21 | 0.87 | 0.70 |
| | **96000** | 1.80 | 1.13 | 0.63 | 0.47 | 0.37 |

*Table 4.3.1 Device Overhead Time in Milliseconds. The time required to fill the audio buffer twice (input and output) is removed from the Total RTT shown in Figure 4.3.2.*

Unlike the i2s passthrough overhead latency, the buffer passthrough overhead latency is influenced by the buffer size. Table 4.3.2 removes the overhead that was determined in the i2s overhead test as shown in Table 4.2.1 in order to provide a clearer representation of the additional time associated with the use of application buffers in contrast to direct i2s passthrough.

| | | Buffer Size | | | | |
|---|---|---|---|---|---|---|
| | | **128** | **64** | **32** | **16** | **8** |
| **Sample Rate** | **44100** | 3.54 | 2.00 | 0.82 | 0.41 | 0.24 |
| | **48000** | 3.27 | 1.95 | 0.70 | 0.37 | 0.19 |
| | **96000** | 1.55 | 0.88 | 0.38 | 0.21 | 0.11 |

*Table 4.3.2 Latency in milliseconds introduced when using Application Buffers*

As can be seen by comparing Table 4.3.2 to the default buffer delays in Table 2.1.1, the introduction of an application buffer has resulted in additional delay equal to the duration of approximately one audio buffer, which is to be expected.

## 4.4 Ethernet Latency – One Way

### 4.4.1 Overview

This test was to determine the latency associated with transmitting audio buffers between two LASP devices over the devices' ethernet ports. The measurement device transmitted data of the appropriate buffer sizes via UDP to the LASP, and the LASP would signal to the measurement device that it had received the packet. This test determines the latency of the sending and receiving stages, with target latencies of between 100 microseconds and 1 millisecond.

### 4.4.2 Methodology

For this test, two identical Teensys were used with ethernet ports attached. For the purposes of this test, the device responsible for sending the data and measuring the latency is referred to as the measurement device, and the device responsible for receiving the data and signalling to the measurement device that the data had been received is referred to as the LASP.

The ethernet ports of the two devices were connected directly via a CAT6 patch cable. In order for the LASP to signal to the Measurement Device that the data had been received, pin 33 of the LASP was connected to pin 26 of the Measurement Device, and the LASP would set this pin to HIGH when its application code registered that the transmitted data was successfully received. This configuration of this test setup is shown in Figure 4.4.1.

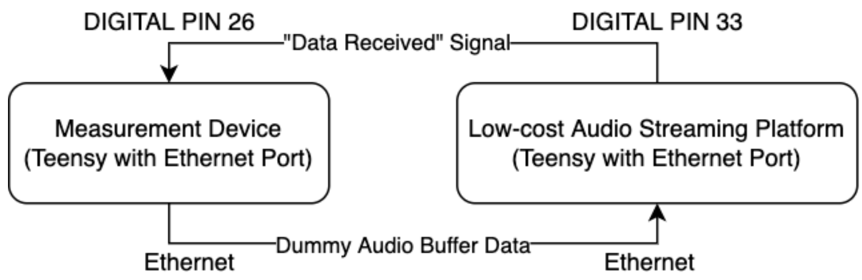



*Figure 4.4.1 Data Flow and Physical Connections between the Measurement Device and the LASP for the One-Way Ethernet Latency Test*

### 4.4.3 Test Procedure

The measurement device started a timer before transmitting a UDP packet of the appropriate buffer size to the LASP. The measurement device then waited for pin 26 to be set to HIGH by the LASP, signalling the UDP packet had been received. This test was automatically repeated 100 times at 1 second intervals for each buffer size and the average of those values was determined as the resulting latency for this thesis.

### 4.4.4 Results

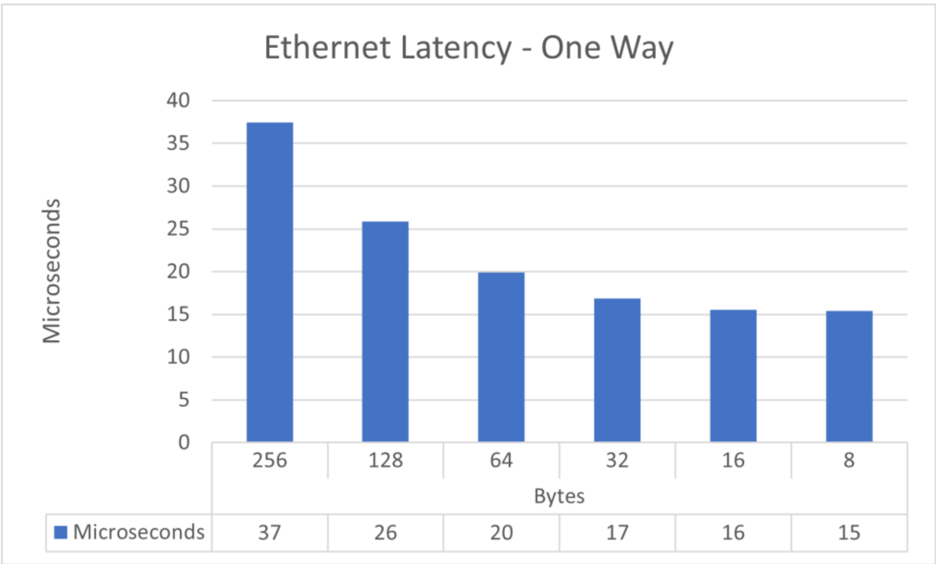



*Figure 4.4.2 Latency results in microseconds of the Ethernet Latency - One Way Test*

#### *4.4.4.1 Discussion*

It is important to note that unlike the previous graphs, the results shown in Figure 4.4.2 are in microseconds, representing significantly lower latencies than the previous tests. The greatest improvement in latency is between 256 and 128 buffer sizes, representing an 11 microsecond reduction, and from there the reduction is halved as the buffer size is halved, resulting in only 1 microsecond improvements between 32, 16 and 8 byte buffers. This is likely a result of the additional frame headers for the UDP packet and lower OSI layers that are of a consistent size regardless of the data being sent and latencies in the hardware/software for ethernet transmission itself.

These results are significant because the processes of "sending" and "receiving" in Table 4.1.1 are categorised as slight delays, equating to delay values between approximately 100 microseconds and 1 millisecond. However, these results are all well below 100 microseconds, which instead meets the requirements for classifying as insignificant delay.

## 4.5 Ethernet Latency – Loopback

### 4.5.1 Overview

This test was designed to determine the latency involved in ethernet transfer and application buffers. While chapter 4.4 Ethernet Latency – One Way only measured transmission time, this loopback required that the received data be read into a buffer within the application before being transmitted back to the measurement device via UDP. In this way that latency is not just for transmission of the data but also includes processing the audio buffer so that it is accessible in code.

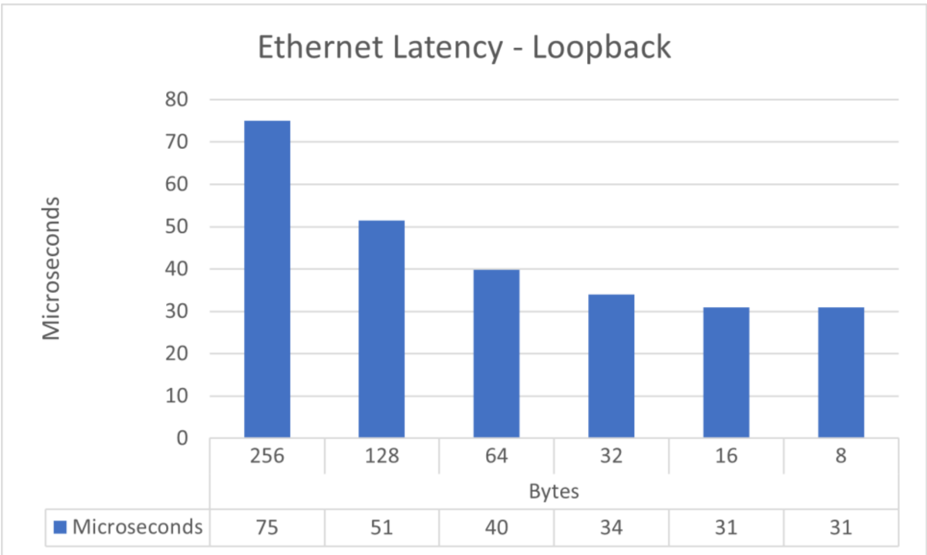
### 4.5.2 Methodology

The methodology for this test was the same as the previous test in chapter 4.4.2 with the following changes. When the data was received by the LASP it was copied into an application buffer and retransmitted back to the measurement device. When the measurement device registered this data as available it stopped the latency timer. This configuration is shown in Figure 4.5.1.


*Figure 4.5.1 Data Flow and Physical Connections between the Measurement Device and the LASP for the Two-Way Ethernet Latency Test*

The first packet transmitted by the measurement device after rebooting was never received by the LASP so the code for this test was modified to send a single packet after boot that was intended to fail. The dummy audio buffer data now also included the test number (0-99) in the first byte of the transmitted data so the measurement device could verify that the packet being received was the same packet that was sent.

### 4.5.3 Results




| | 256 | 128 | 64 | 32 | 16 | 8 |
|---|---|---|---|---|---|---|
| Microseconds | 75 | 51 | 40 | 34 | 31 | 31 |

*Figure 4.5.2 Latency results in microseconds of the Ethernet Latency - Loopback Test*

### 4.5.4 Discussion

Like the previous table, these results in Figure 4.5.2 are also in microseconds. The loopback latency (RTT) for the ethernet packets is double that of the one way latency (within 1 microsecond).

These results are significant as even after loopback, which involves two sending processes and two receiving processes, the delay incurred is below 100 microseconds, even for the largest buffer size of 256 samples. According to Figure 2.1.3 this qualifies the delay of these processes to be considered insignificant for the LASP and marks a possible improvement over existing NMP technologies.

## 4.6 UDP Bandwidth – One Way

### 4.6.1 Overview

The purpose of this test was to determine the bandwidth of the LCAPs ethernet connection.

### 4.6.2 Methodology

Since UDP does not guarantee delivery, the bandwidth had to be determined on the receiving end of the transmission.  The physical setup for this test was the same as the methodology in the previous chapter 4.5.2.

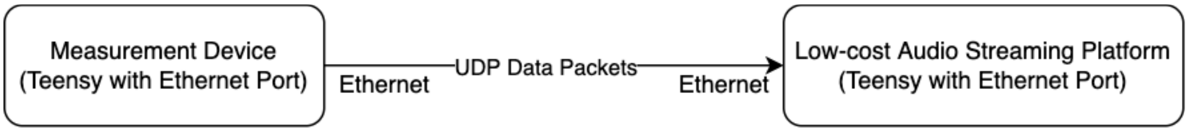



*Figure 4.6.1 Data Flow and Physical Connections between the Measurement Device and the LASP for the UDP Bandwidth – One Way Test*

### 4.6.3 Test Procedure

The Measurement Device continually sent UDP data packets to the LASP.  This dataflow can be seen in Figure 4.6.1.  The LASP would keep track of how many bytes had been received and reported this total value every second via serial connection before resetting the total to zero, resulting in a bits per second value.  This test was done for 10 seconds and the average of these bits per second values are presented in the following results.

### 4.6.4 Results

| Device | Receive Mbps |
|---|---|
| LCAP | 95.80 |

*Table 4.6.1 Results of the UDP Bandwidth – One Way Test*

### 4.6.5 Discussion

The result in Table 4.6.1 shows the LASP efficiently utilises the bandwidth of its 100Mbps ethernet chip.  A 48kHz/16bit audio stream has a bitrate of 0.768Mbps which is 0.8 percent of the device's available bandwidth. Put another way, the devices bandwidth is sufficient for sending or receiving over 100 audio streams at 48kHz/16 bit.

## 4.7 UDP Bandwidth – Two Way

### 4.7.1 Overview

This purpose of this test was to determine the bandwidth capabilities of the LASP when sending and receiving data simultaneously.

### 4.7.2 Methodology

The methodology is the same as that outlined in the previous chapter 4.6.2 except both the Measurement Device and LASP are continually sending UDP packets whilst receiving UDP packets from the other device. This configuration and dataflow can be seen in Figure 4.7.1.

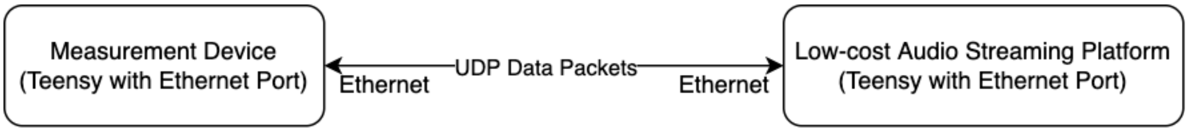



*Figure 4.7.1 Data Flow and Physical Connections between the Measurement Device and the LASP for the UDP Bandwidth – Two Way Test*

### 4.7.3 Results

| Device | Receive Mbps |
|---|---|
| LCAP 1 | 94.56 |
| LACP 2 | 94.45 |

*Table 4.7.1 Maximum Bandwidths Supported while Simultaneously Sending and Receiving UDP Data*

### 4.7.4 Discussion

The results in Table 4.7.1 show that there's very little negative impact on the devices network bandwidth when both sending and receiving in comparison to only sending or receiving. With a 1.4% decrease in total bandwidth, it is still sufficient bandwidth equivalent to sending and receiving over 100 audio streams at 48kHz/16bit.

## 4.8 My Mouth to My Ear Latency – One Channel

### 4.8.1 Overview

The purpose of this test was to determine the latency of the device in an NMP setup. The results of this test are compared against the NMP required Ensemble Performance Threshold (EPT) of 25ms. The further the latency results are under this threshold, the more time is available for routing and jitter buffering on WANs, resulting in higher quality NMP over larger distances. A LASP captures audio data and sends it to a second LASP which performs an audio loopback to itself before sending the audio data back to the first LASP. All the Signal Path Stages in Table 4.1.1 are present in this test with the exception of routing and jitter buffering.

### 4.8.2 Methodology

In this setup, two LASPs were used, referred to as the local LASP and remote LASP. A third Teensy 4.1 was introduced as the measurement device. The physical connections and data flow can be seen in Figure 4.8.1.

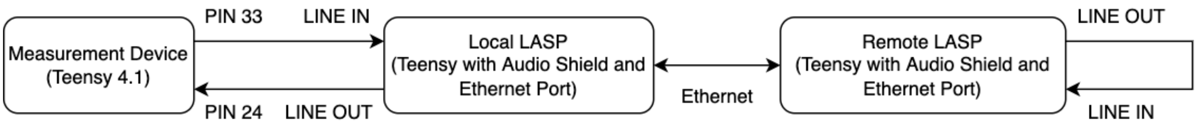



*Figure 4.8.1 Data Flow and Physical Connections between the Measurement Device and two identical LASP devices for "My Mouth to My Ear" Latency Test*

The two LASPs run identical code. Audio captured on their LINE IN was sent via their ethernet interface, and data received on their ethernet interface was transferred to their LINE

OUT. The devices were programmed to continuously capture and send audio data, and receive and output audio data.

It was discovered that the pins on the Teensy 4.1 Measurement Device could handle the negative voltages output by the LINE OUT of the LASP audio shield, which would be clipped to LOW (0 volts). This allowed much more accurate measurements as the audio shield in previous tests required a buffer of 4 samples and had a latency of 162 microseconds, whereas an analogue read of a Teensy pin took only 1 microsecond.

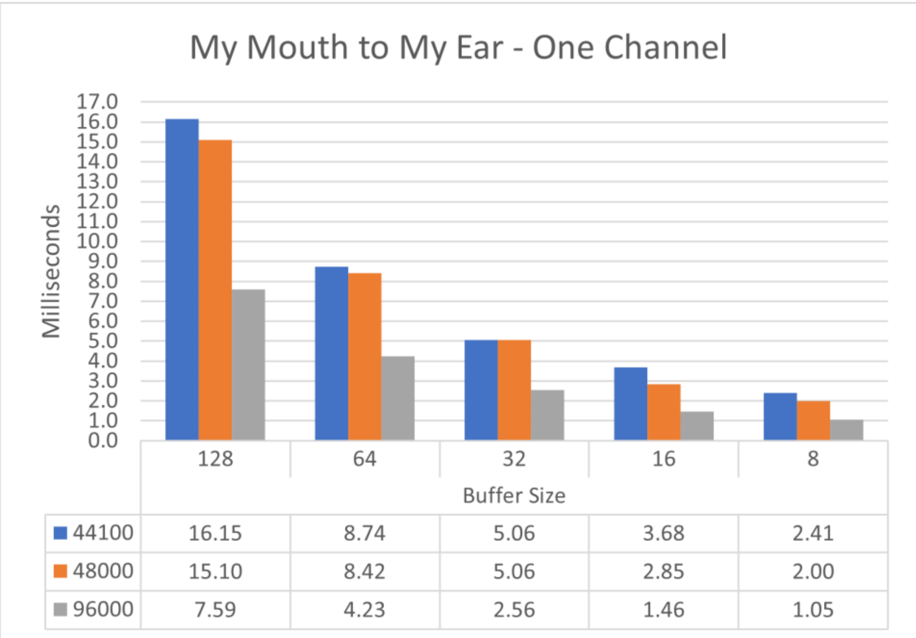
#### *4.8.2.1 Test Procedure*

The Measurement Device would start a timer and produce a signal by changing output pin 33 from LOW (0 volts) to HIGH (3.3 volts). It would then make continuous analogue readings on input pin 24 until the signal was detected, stopping the timer and reporting the latency via USB Serial Connection.

The output pin of the measurement device is connected to the LINE IN of the local LASP. The local LASP captures audio on its LINE IN and sends the audio buffer to the remote LASP via its ethernet interface. The remote LASP receives this audio data on its ethernet interface and routes the audio data to its LINE OUT which is looped directly back to the remote device's LINE IN. The remote device then captures its LINE IN audio data and sends it back to the local LASP via its ethernet interface. Finally, the local LASP receives the audio data on its ethernet interface and routes the audio data to its LINE OUT, which is connected to the measurement device's input pin.

Each test configuration was automatically repeated 100 times and the average of these latencies was reported in the results below.

### 4.8.3 Results




| | 128 | 64 | 32 | 16 | 8 |
|---|---|---|---|---|---|
| 44100 | 16.15 | 8.74 | 5.06 | 3.68 | 2.41 |
| 48000 | 15.10 | 8.42 | 5.06 | 2.85 | 2.00 |
| 96000 | 7.59 | 4.23 | 2.56 | 1.46 | 1.05 |

*Figure 4.8.2 "My Mouth to My Ear – One Channel" Latency Measurement Results*

### 4.8.4 Discussion

The results shown in Figure 4.8.2 are the total latency values for MM2ME. Although in most cases they already fall well below the EPT, these values are return trip times, so they need to be halved before comparing to the EPT of 25ms. These halved values, equivalent to the Over-all One-way Source-to-Ear (OOSE) are shown in Table 4.8.1.

| | | **Buffer Size** | | | | |
|---|---|---|---|---|---|---|
| | | **128** | **64** | **32** | **16** | **8** |
| **Sample Rate** | **44100** | 8.07 | 4.37 | 2.53 | 1.84 | 1.21 |
| | **48000** | 7.55 | 4.21 | 2.53 | 1.42 | 1.00 |
| | **96000** | 3.79 | 2.12 | 1.28 | 0.73 | 0.53 |

*Table 4.8.1 Over-all One-way Source-to-Ear Delay between Two LASP Platforms*

The results in Table 4.8.1 show excellent performance from the LASP, with OOSE latencies as low as half a millisecond at 96kHz with an 8 sample buffer size. At the NMP minimum sample rate of 48kHz and the LOLA minimum buffer size of 32 samples, the OOSE latency was 2.53 milliseconds, resulting in more than 22 milliseconds for routing and jitter buffering.

In order to determine the delay introduced by the LASP devices excluding the standard buffer blocking time, Table 4.8.2 was created which removed latencies equivalent to one input and output buffer from the results in Table 4.8.1.

| | | **Buffer Size** | | | | |
|---|---|---|---|---|---|---|
| | | **128** | **64** | **32** | **16** | **8** |
| **Sample Rate** | **44100** | 2.27 | 1.47 | 1.08 | 1.12 | 0.84 |
| | **48000** | 2.22 | 1.54 | 1.20 | 0.76 | 0.66 |
| | **96000** | 1.13 | 0.78 | 0.61 | 0.40 | 0.36 |

*Table 4.8.2 Additional delay introduced by each LASP in the My Mouth to My Ear – One Channel Latency Test*

At the NMP required sample rate of 48kHz and above, the LASP introduces latencies of 2.22 milliseconds or less.

## 4.9 My Mouth to My Ear Latency - Two Channel

### 4.9.1 Overview

This test was used to determine the overall latency introduced by adding a second channel to an NMP. The test is identical to the previous single channel test except two audio channels are now used. Similarly, the results of this test can be compared against the EPS for NMPs that require two channel audio.

### 4.9.2 Methodology

When transmitting over the ethernet connection, the first half of the UDP packet contained the samples from the left channel buffer and the second half of the UDP packet contained the samples from the right channel buffer. This is in contrast to the i2s protocol which alternates between the left and right channel for each sequential sample, but is the same format used by JackTrip.

### 4.9.3 Results

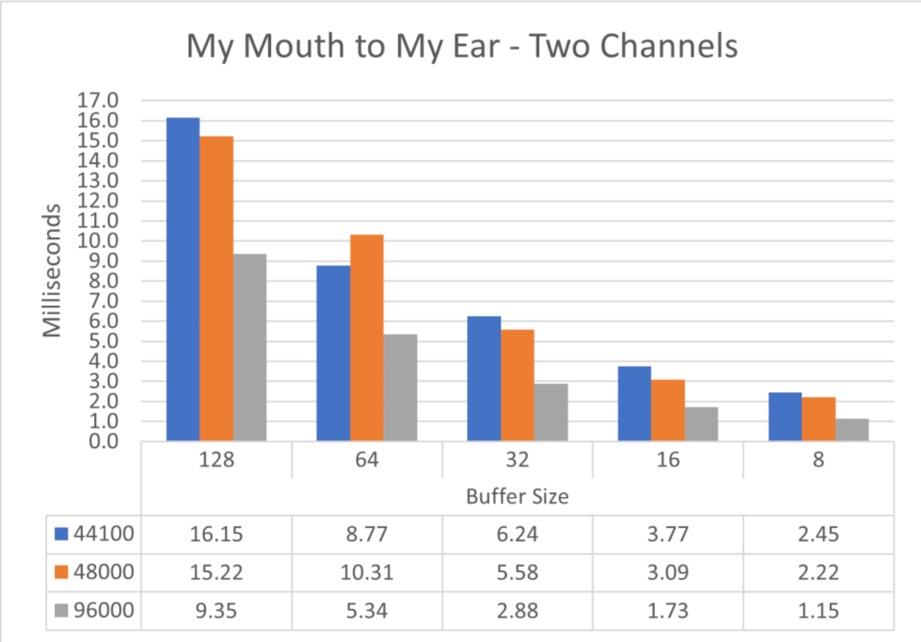



*Figure 4.9.1 My Mouth to My Ear – 2 Channel Latency results for the LASP*

### 4.9.4 Discussion

The 48khz/64 sample buffer result shows higher latency than the 41khz/64 sample buffer result, and this is not in line with what would be expected from previous results. It is most likely that during this test, LASP was correctly programmed to run at 64 buffer size while the other had an error during the update and was still running at 128 buffer size. If this assumption is correct, it would be expected that an accurate result would have returned a latency of around 8.5ms, based on the other results from this test.

To determine the latency introduced by including a second audio channel, the resulting latencies from the single channel test were remove from the dual channel test in Table 4.9.1.

| | | Buffer Size | | | | |
|---|---|---|---|---|---|---|
| | | 128 | 64 | 32 | 16 | 8 |
| Sample Rate | 44100 | 0.00 | 0.03 | 1.18 | 0.08 | 0.04 |
| | 48000 | 0.11 | 1.89 | 0.52 | 0.24 | 0.22 |
| | 96000 | 1.76 | 1.11 | 0.33 | 0.28 | 0.10 |

*Table 4.9.1 Additional Latency Introduced by Including a Second Audio Channel*

Generally, the additional latency introduced by the second audio channel is well below a millisecond, however the highest sample rate along with the largest buffer sizes were most affected and it is unclear what signal path stage(s) caused this.

## 4.10 My Mouth to My Ear - WAN Latency

### 4.10.1 Overview

The purpose of this test was to take the LASP “out of the lab” and test it in a real-world networked configuration. The previous two tests had the LASPs connected directly to each other via ethernet patch cable which was great for removing external factors that could impact latency but is not indicative of a real NMP, which typically takes place over wide area networks.

### 4.10.2 Methodology

This test used identical code to the previous two tests, except the LASPs were configured to use DHCP instead of static IP addresses and the devices were configured to send their audio data to a server in Oslo that forwarded the UDP packets back to the other LASP.

The server in Oslo was programmed to keep track of all client devices it had received UDP data from. Whenever it received subsequent UDP data packets it would forward them to all devices it had previously received UDP data from, with the exception of the device where the packet originated. This hardware setup and data flow is shown in Figure 4.10.1.

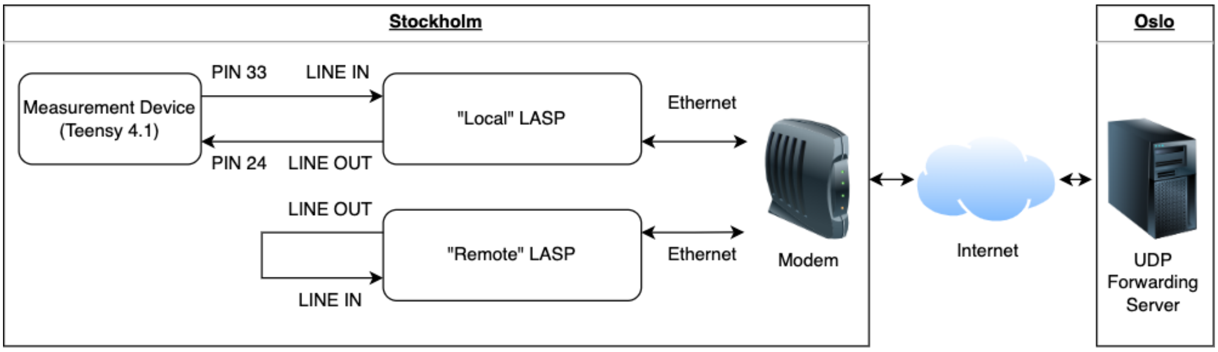



*Figure 4.10.1 Data Flow and Physical Connections between the Measurement Device and two identical LASP devices for the My Mouth to My Ear - WAN Latency test*

#### 4.10.2.1.1 Test Procedure

The signal path is:

1. Measurement Device Pin 33
2. Local LASP Audio Input
3. Local LASP Ethernet Transmit (Stockholm)
4. UDP Forwarding Server Receive/Transmit (Oslo)
5. Remote LASP Ethernet Receive (Stockholm)
6. Remote LASP Audio Output
7. Remote LASP Audio Input (Loopback)
8. Remote LASP Ethernet Transmit (Stockholm)
9. UDP Forwarding Server Receive/Transmit (Oslo)
10. Local LASP Ethernet Receive (Stockholm)
11. Local LASP Audio Output
12. Measurement Device Pin 24

### 4.10.3 Results

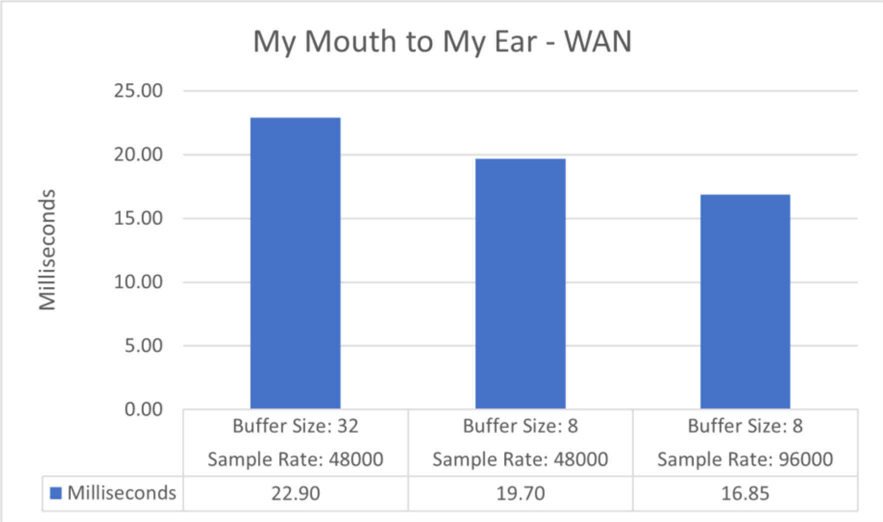



*Figure 4.10.2 "My Mouth to My Ear" Latency Measurement Results of the LASP over WAN between Stockholm and Oslo*

### 4.10.4 Discussion

The results shown in Figure 4.10.2 are MM2ME so they need to be halved before comparing to the EPT of 25ms. These values are shown in Table 4.10.1 below, and represent latency values less than half the EPT.

| | | **Buffer Size** | |
|---|---|---|---|
| | | **32** | **8** |
| **Sample Rate** | **48000** | 11.45 | 9.85 |
| | **96000** | | 8.43 |

*Table 4.10.1 Over-all One-Way Source-to-ear Delay for the LASP over a distance of ~830km*

Oslo to Stockholm is approximately 415km in a straight line. The signal is transmitted back and forth between the two devices in Stockholm via a server in Oslo, resulting in a total of *two* round trips from Stockholm to Oslo due to the loopback required for My Mouth to My Ear, for a total of 1,660km. Based on straight line distance, this would be comparable to a NMP with a separation of 830km which is approximately the distance between Stockholm and Berlin. A ping test was performed from the network in Stockholm to the UDP Forwarding Server in Oslo which showed a round trip time of 8ms. Since the signal completes this entire round-trip twice, that results in a total of 16ms of network transmission time on the WAN.

These latency results represent a best case scenario of a real world network music performance setup. Unless connected via a high performance WAN such as those provided at universities, it is unlikely that an NMP could achieve this level of latency due to network jitter, and additional buffers would be required to minimise this unreliability, resulting in higher latency. However, the results were considerably below the EPT of 25ms so larger sample buffers and/or the inclusion of sufficient jitter buffers would likely not be an issue.

## 4.11 Audio Library CPU Usage

### 4.11.1 Overview

The purpose of this test was to determine the CPU usage of the Audio Library. This could be used to determine whether the device it at risk of becoming overloaded or whether the cost could be further reduced by utilising a less powerful processor.

### 4.11.2 Methodology

The Audio Library Code for the Teensy Audio Shield has a function to determine the percentage of CPU time associated with performing audio library functions. The setup is the same as that of except no signal or measurement device is used.

#### *4.11.2.1 Test Procedure*

The LASP continually routed data from the LINE IN to the ethernet interface, and audio data received on the ethernet interface was routed to the LINE OUT. Every second, the maximum audio CPU usage reading was made which is sent to the USB Serial Port, and the maximum audio CPU usage value was reset. This was performed automatically 100 times and the average was displayed in the results below.

### 4.11.3 Results

| Sample Rate | Sample Size | Audio CPU Percentage |
|---|---|---|
| 48000 | 32 | 0.14 |

*Table 4.11.1 Audio Library CPU Usage for the LASP*

### 4.11.4 Discussion

The results in Table 4.11.1 show that the audio library uses less that 1 percent of the LASP's CPU time and is not at risk of overloading. The CPU usage is significantly low enough that a less powerful processor would be sufficient for the device's audio functions.

## 4.12 JackTrip Integration

### 4.12.1 Overview

The purpose of this test was to evaluate the LASP's ability to function as a NMP device by implementing and integrating it with an existing and proven NMP technology.

### 4.12.2 Methodology

This setup consisted of a Windows PC running JackTrip, a LASP and the UDP Forwarding Server. The PC and LASP were located in Stockholm while the UDP Forwarding Server was in Oslo.

The Windows PC used an external MOTU M4 soundcard running ASIO drivers at 48kHz and a buffer size of 64 samples. The soundcard was connected to a pair of studio monitor speakers. Voicemeeter Potato software was used to simultaneously route the PC's audio to the MOTU M4 and the JackTrip software running on the PC.

The LASP was connected to amplified speakers using the LINE OUT pins. The LASP code was updated to handle the additional JackTrip header in the UDP packets.

Although the computer and LASP were communicating via the UDP forwarder in Oslo, they still connected using JackTrip's Peer to Peer mode because the server in Oslo simply forwarded the packets between the two devices.

The physical connections and dataflow diagram can be seen in Figure 4.12.1.

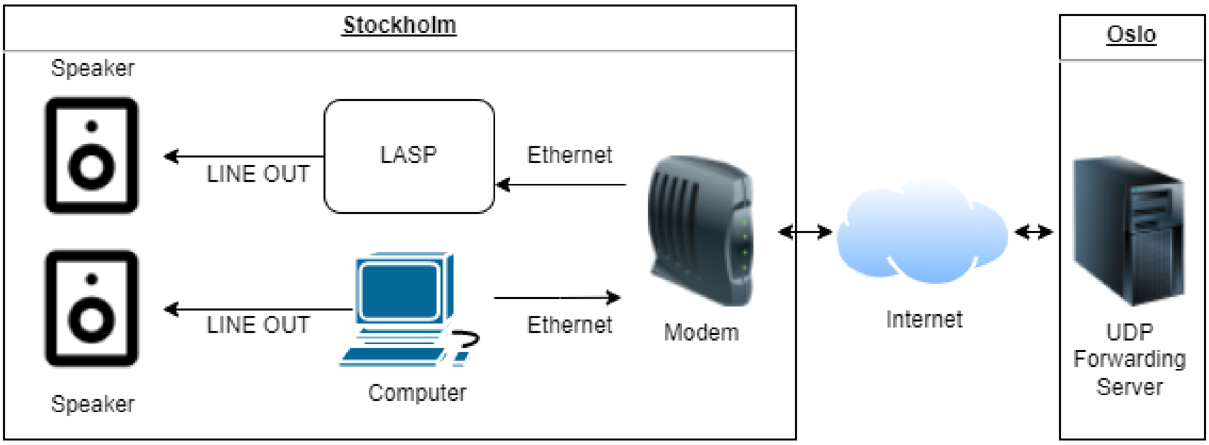



*Figure 4.12.1 Physical Connections and Dataflow Diagram for the JackTrip LASP Test*

Although Figure 4.12.1 only shows the dataflow of the audio stream emanating from the computer, there was a second audio stream containing the LINE IN audio from the LASP that travelled to the Computer via the UDP Forwarding Server, as there would be in a regular NMP setup. However, this stream contained no audio during this test.

##### 4.12.2.1.1 Test Procedure

The Windows PC in Stockholm played an audio file that could be heard locally through a speaker connected to the PCs external soundcard.  The PC also routed the audio stream to the JackTrip software which sent the stream to the UDP Forwarding Server in Oslo. The UDP Forwarding Server forwarded the audio data to the LASP in Stockholm where it was played through the external amplified speaker connected to the LASP's LINE OUT.

### 4.12.3 Results

When the volume of the PC speakers and the LASP speakers where balanced it was not possible to tell that there were two separate audio streams playing however some audio glitches were noticed.

### 4.12.4 Discussion

The LASP functioned successfully alongside a proven state-of-the-art NMP technology. Assuming the latency was consistent with previous tests of around 11ms, larger sample buffers or jitter buffers could have been used to minimise the audio glitches that were experienced.

## 4.13 Summary

The LASP was thoroughly evaluated and found to both meet the ultra-low-latency requirements and exceed the high-quality audio requirements of current state of the art NMP platforms. Additionally, the prototype cost $44.50 which firmly qualifies it as a low-cost NMP platform.

# 5 Recommendations

## 5.1 Limitations

Relying on a proven platform in the Arduino ecosystem saved significant amounts of development time.  However, this resulted in compilation and performance errors that prohibited the LASP from functioning at buffers below than 8 samples when the codec is in fact capable of lower buffer sizes.  Using existing libraries also made it difficult to pinpoint the exact cause of particular latencies however this compromise had to be made in the interests of developing a prototype in a reasonable timeframe.

Though many platforms made claims about their low latencies, none provided actual test results and test frameworks that could be replicated and compared to.  The very nature of NMP implies testing over a WAN and indeed the majority of tests involved an arbitrary WAN or simulated WAN conditions with a focus on the Ensemble Performance Threshold rather than the impact that individual components of the platform have on the final latencies. For this reason, a reproduceable environment such as a JackTrip Analog Bridge could have been purchased and benchmarked against for a direct comparison rather than the target latencies, bitrates and sample rates found from various papers and installation manuals.

## 5.2 Future Work

Now that the core benchmarks and cost requirements have been achieved, the next step should be to test the performance of the LASP in a real Network Music Performance.  Future work for a low-cost low-latency audio streaming platform for NMP could then go in several directions – further reducing cost, improving/increasing functionality and/or developing a more robust prototype.

As mentioned previously, the Teensy is capable of functioning as a USB audio interface. If the LASP was capable of functioning both as a standalone NMP platform and a low-latency USB audio interface for laptop based NMPs it would become an even more attractive as a product.

# 6 Conclusion

This thesis investigated the current state of NMP platforms and found a gap in the market for a low-cost ultra-low-latency audio streaming alternative.  A prototype device was developed which upon evaluation was shown to meet the low-latency requirements and exceed the high quality audio standards set by the current state-of-the-art NWP platforms. Additionally, it was able to provide this performance for a third of the cost of the competing low-cost platform. This thesis has shown that microcontrollers should be considered for the development of audio based NMP platforms as they are not only capable of providing higher quality audio at lower latencies than general purpose computers and some single board computers but are also significantly cheaper.

# 7 References


Bosi, M., Servetti, A., Chafe, C., & Rottondi, C. (2021). Experiencing Remote Classical Music Performance Over Long Distance: A JackTrip Concert Between Two Continents During the Pandemic,. *Journal of the Audio Engineering Society*, 934-945.

Bouillot, N. (2007). nJam User Experiments Enabling Remote Musical Interaction from Milliseconds to Seconds. *Proceedings of the International Conference on New Interfaces for Musical Expression*, (pp. 142-147). New York City.

Bouillot, N., & Cooperstock, J. R. (2009). Challenges and Performance of High-Fidelity Audio Streaming for Interactive Performances. *NIME*, (pp. 135-140).

Caceres, J.-P. (2022, Jan 7). *PacketHeader.h*. Retrieved from GitHub: https://github.com/jacktrip/jacktrip/blob/main/src/PacketHeader.h#L56

Carôt, A. (2009, May). *Musical telepresence – A comprehensive analysis towards new cognitive and technical approaches.* Retrieved from Soundjack Publications: https://www.soundjack.eu/publications/

Carôt, A., & Werner, C. (2009). Fundamentals and Principles of Musical Telepresence. *Journal of Science and Technology of the Arts*, *1*, 26-37.

Carôt, A., Hoene, C., Busse, H., & Kuhr, C. (2020). Results of the Fast-Music Project—Five Contributions to the Domain of Distributed Music. *IEEE Access, 8*, 47925-47951. doi:10.1109/ACCESS.2020.2979362

Carôt, A., Kramer, U., & Schuller, G. (2006, May). Network Music Performance (NMP) in Narrow Band Networks.

Chafe, C., & Caceres, J.-P. (2009). JackTrip: Under the Hood of an Engine for Network Audio. *Proceedings of the International Computer Music Conference.* Montreal: Center for Computer Research in Music and Acoustics (CCRMA), Stanford University.

Conservatorio di musica G. Tartini. (2022). *LoLa Suggested Hardware.* Retrieved from LoLa Low Latency AV Streaming System: https://lola.conts.it/downloads/LOLA_HARDWARE_LIST.pdf

Cook, M. (2015). *Telematic Music - history and development of the medium and current technologies related to performance.* Bowling Green State University.

Davies, G. (2015). The effectiveness of LOLA (LOw LAtency) audiovisual streaming technology for distributed music practice. *M. Research diss., Edinburgh Napier University*.

Drioli, C., & Buso, N. (2013, January). Networked Performances and Natural Interaction via LOLA: Low Latency High Quality A/V Streaming System. *Lecture Notes in Computer Science, 7990*, 240-250. doi:10.1007/978-3-642-40050-6_21

Jack, R. H., Moro, G., & McPherson, A. P. (2016). Action-Sound Latency: Are Our Tools Fast Enough? *NIME.* Brisbane.

JackTrip Labs, I. (2022). *HiFiBerry DAC+ ADC Pro*. Retrieved from JackTrip Labs Store: https://store.jacktriplabs.com/product/hifiberry-dac-adc-pro/

Lopez-Lezcano, F., & Wilkerson, C. (2007). *CCRMA Studio Report.* Stanford University.

Murray, L. (2022). *Thesis Code*. Retrieved from GitHub: https://github.com/leighmurray/thesis-code

NXP Semiconductors. (2022, January). Low Power Stereo Codec with Headphone Amp. *SGTL5000 Data Sheet: Technical Data*.

Rofe, M., & Reuben, F. (2017, September). Telematic performance and the challenge of latency. *Journal of Music, Technology and Education, 10*, 167-183. doi:10.1386/jmte.10.2-3.167_1

Rottondi, C. a. (2016). An Overview on Networked Music Performance Technologies. *IEEE Access*, 8823 - 8843.

Silverman, S. (2022). *QNEthernet*. Retrieved from GitHub: https://github.com/ssilverman/QNEthernet

Smith, G., Moir, Z., Ferguson, P., & Davies, G. (2020). Low-latency networked music collaborations: does "good enough" do enough good? *Journal of Network Music and Arts, 2*. Retrieved from https://open.bu.edu/handle/2144/42378

Tsioutas, K., & Xylomenos, G. (2021). On the Impact of Audio Characteristics to the Quality of Musicians’ Experience in Network Music Performance. *Journal of the Audio Engineering Society*, 914–923.

Tsioutas, K., Xylomenos, G., & Doumanis, I. (2019). Aretousa: A competitive audio streaming software for Network Music Performance. *146th AES Convention.* Ireland.

Tympan. (2022). *Open Hearing*. Retrieved from TYMPAN: https://shop.tympan.org/

Villanueva, P. (2022). *ADC*. Retrieved from GitHub: https://github.com/pedvide/ADC

Werner, C., & Kraneis, R. (2021). UNISON: A Novel System for Ultra-Low Latency Audio Streaming Over the Internet. *2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC)* (pp. 1–4). IEEE. doi:10.1109/CCNC49032.2021.9369466