Aus dem Institut für Telematik
der Universität zu Lübeck

Direktor:
Prof. Dr. rer. nat. Stefan Fischer

Musical Telepresence - A Comprehensive Analysis Towards New Cognitive and Technical Approaches

Inauguraldissertation
zur
Erlangung der Doktorwürde
der Universität zu Lübeck
– Aus der Technisch-Naturwissenschaftlichen Fakultät –

Vorgelegt von
Herrn Dipl.-Ing. Alexander Carôt
aus Lübeck

Lübeck, im Mai 2009

Erster Berichterstatter: Prof. Dr.-Ing. Christian Werner
Zweiter Berichterstatter: Prof. Dr. Chris Chafe

Tag der mündlichen Prüfung: 20.7.2009

Zum Druck genehmigt.

Lübeck, den 20.7.2009

gez. Prof. Dr. Jürgen Prestin
– Dekan der Technisch-Naturwissenschaftlichen Fakultät –

Preface

In my opinion working in an interdisciplinary domain always implies certain aspects of being "caught in the middle": The more one resides between two disciplines and the more one tries to take each of them into account, the higher the risk of becoming a generalist without specific expert knowledge. This special expertise, however, is required as a basis for successful research and development, which is why interdisciplinary work is often not taken seriously in the scientific community. As a consequence, one has to make a strong effort to become an expert in either discipline in order to fully understand their interrelationships. Finally, the awareness of such interrelationships allows interdisciplinary researchers to approach each discipline from a different perspective as the combination of both fields possibly leads to totally different insights and requirements.

As a computer scientist and a musician I have been exploring the intersection between the domains of IT and music for the last 15 years. This dissertation represents a personal milestone, which integrates my artistic experience in music with the scientific rigour and methodology from the disciplines of computer science and sound engineering. I am proud of having "married" both fields within this book. Nevertheless, although I see a tremendous potential and challenge in the interdisciplinary domain, I had to realize that our current economic and scientific structures barely support interdisciplinary approaches and often do not consider them useful. In my opinion this tendency has been reinforced by the crash of the new economy and the breakdown of many new media education concepts. Moreover, it appears to me that an unconventional point of view can often result in misunderstandings and in turn, leads to rejection by mainstream science. In this context it takes a huge effort to defend one's conclusions in order to be taken seriously. In fact, this insight led to my experiencing moments of frustration, which ultimately served as the motivation to establish my own approach towards computer science, music and their combination. Furthermore, while looking back at the past years of research and development, I come to the conclusion that it is of major importance for interdisciplinary researchers to find the right people to work with in order to obtain the support and appreciation that they deserve.

I have been in touch with numerous people who showed an interest in my work. Their respective fields of expertise ranged from music, musicology and audio engineering to computer science, telecommunications, electrical engineering and digital signal

processing. In the following, I would like to thank those who had a major positive impact on my path of becoming a doctor of engineering. First of all – in terms of supervision – I would like to thank Christian Werner for being an outstanding guide, advisor and editor. His help in successfully completing my Ph.D. thesis was crucial. Also I would like to thank my second advisor Chris Chafe for constantly being there as a mentor and protector of the network music idea. Regarding the administrative support I would like to thank Andreas Schrader for initiating the doctoral process, Stefan Fischer for putting me in touch with the right people and Alfred Mertins for chairing my doctoral defense. Furthermore, I want to appreciate the patience and mental support of my wife, Katja, my parents Sigrid and Germund Raywood and my parents-in-law Irmhilde and Karl Englert. Their wisdom and comments helped me to overcome hard times and to focus on my initial motivation and goal. In the context of inspirational support, new visions and their implementation, I would especially like to mention Alain Renaud, Alvaro Barbosa, Gerald Schuller and Jeremy Cooperstock. Finally, I want to thank my colleagues of the University of Lübeck, Germany for their valuable feedback, James Darcy for proofreading and editing my dissertation and anyone else I have been working with during the past years.

Abstract

Due to the conventional "best-effort" manner without any guarantee of packet delivery, the Internet was not originally developed for the purpose of sending real time traffic as it is the case with audio data. Packet loss and delay variations typically lead to dropouts in the received audio stream which result in signal errors, and correspondingly, disturbing clicks and noise cracks. Recent Voice-over-IP (VoIP) and video conferencing services overcome these problems by applying large audio frames, large network buffers, and the principle of packet retransmissions. Consequently this results in additional latencies of several hundred milliseconds. Such delays do not represent a problematic figure in the context of voice communication.

As part of the evolving globalization process and its distributed work processes telecommunication on the Internet has become a widely accepted and commonly used service. Based on these facts it is the author's interest to figure in how far distributed communication on the Internet can be applied in terms of artistic music performances: Such a scenario exhibits signal delay boundaries below 30 ms as though musicians were to perform in the same room. In this context the author aims to determine the primary problematic aspects related to delay and quality in order to achieve adequate realistic performance conditions.

Hence, as a fist step, the author will examine the precise cognitive restrictions for a delayed musical interplay and determine the relating technical requirements: Based on these cognitive restrictions, each technical aspect of latency will be examined in terms of its inherent signal delivery speed and the corresponding latency. Secondly, a new technical approach will be presented, which sends UDP packets as quick as possible across a network. It uses a minimal audio frame size and small network buffers and avoids packet retransmissions in order to eliminate any source of additional delay. Audio dropouts due to jitter and packet loss are consciously taken into account up to an individually adjustable level. In that context the author will perform latency and audio dropout measurements in order to show that current broadband and A-DSL networks are able to process low delay audio streams with the desired stability and quality.

Furthermore, the author's new approach of low delay audio processing will be used as a basis for new technical solutions. Each of them aims to reduce latency and achieve adequate live music conditions.

With this work, the author will show that – with respect to the investigated technical and cognitive results – distributed music on the Internet is possible and must be

considered as a serious domain of future applications and research. By consciously taking all relevant – also interdisciplinary – aspects of delayed musical interaction into account, the author builds a solid scientific basis for future investigations in distributed music-related research.

Kurzfassung

Das Internet ist ursprünglich nicht für den Versand von Echtzeitdaten ausgelegt. Die asynchrone Betriebsweise und das "Best-effort"-Prinzip führen zu Paketverlusten und Verzögerungsvariationen. Für den Versand von Audiodaten hat dies zur Folge, dass ein über das Internet empfangener Audiodatenstrom Dropouts bzw. Signalfehler aufweist. Internettelefonie- und Videokonferenzdienste kompensieren diese Probleme durch den Einsatz großer Audioframes, großer Netzwerkpuffer und das erneute Übertragen verlorener Pakete. Diese Verfahren implizieren zusätzliche Latenzen im Bereich mehrerer hundert Millisekunden, welche allerdings im Rahmen der konventionellen Sprachkommunikation unproblematisch sind.

Im Zuge der Globalisierung ist die Internet-basierte Telefonie zu einem gängigen Kommunikationsverfahren geworden, welches hilft, rämliche Distanzen zum einen im Sinne verteilter Arbeitsprozesse und zum anderen hinsichtlich sozialer Netzwerke überwinden zu können. Das Hauptinteresse des Autors besteht darin, den Bereich der Echtzeitkommunikation im Dienste künstlerischer Musikperformances zu erforschen. Geht man von realistischen Spielbedingungen aus, so ergeben sich für diese Anwendung allerdings wesentlich zeitkritischere Anforderungen, die Latenzen unterhalb von 30 ms bedingen. Aus diesem Grunde besteht das Ziel dieser Arbeit darin, die genauen Schwellwerte hinsichtlich Latenz und Qualität zu erforschen, um im darauf folgenden Schritt ein System zu entwickeln, welches entsprechend realistische Spielbedingungen für professionelle Musiker schaffen kann.

Grundlegend werden im ersten Schritt die genauen kognitiven Anforderungen und Einschränkungen untersucht, die im Zusammenhang einer verzögerten musikalischen Interaktion von Bedeutung sind. Darauf aufbauend widmet sich der Autor den technisch relevanten Aspekten und erforscht die entsprechenden Komponenten hinsichtlich ihrer inhärenten Verzögerungen. Aufgrund der Erkenntnis, dass trotz einer komplexen Signalübertragungskette adequate Latenzen erreicht werden können, wird zweitens ein neuer technischer Ansatz vorgestellt, der UDP-Pakete auf schnellmögliche Weise über das Internet verschickt. Es werden kleine Audioframes und kleine Netzwerkpuffer verwendet und ferner auf das erneute Übertragen verlorener Pakete verzichtet. In diesem Zusammenhang werden Audiodropouts als Folge von Netzwerkjitter und Paketverluste bis zu einer individuellen Akzeptanzgrenze bewusst toleriert.

Auf Basis dieser Technologie führt der Autor Latenz- und Audiodropoutmessungen durch, die zeigen, dass die derzeitigen Breitband- und A-DSL-Netzwerke in der Lage sind, Audiostreams mit der gewünschten Stabilität und Qualität zu transportieren. Nach der erfolgreichen Evaluierung diese Prinzips stellt der Autor schließlich neuartige

technische Lösungen vor, die eine zusätzliche Latenzverringerung und eine weitere Verbesserung der musikalischen Interaktionsbedingungen zur Folge haben.

Gemäß den technischen und kognitiven Ergebnissen dieser Arbeit ist ein verteiltes Musizieren über das Internet möglich, und der Autor kommt zu dem Schluss, dass das vorgestellte Prinzip zukünftig als ernsthafte Telekommunikationsanwendung betrachtet werden muss. Durch die Einbeziehung jedweder relevanter und auch interdisziplinärer Aspekte schafft der Autor eine solide wissenschaftliche Grundlage für zukünftige Untersuchungen in diesem Bereich.

Contents

List of Figures

Chapter 1

Introduction

A musical instrument generates sound. Regardless of whether an amateur simply hits the key of a piano or a virtuoso plays a Mozart piece on a violin, each instrument starts vibrating in its individual manner depending on the player's activity. Sound waves leave the instrument's body and travel through the air to reach our human ears. Via a complex mechanism, these sound waves are received and converted into electrical impulses, which our brain interprets as sound [103]. This principle of music signal creation and reception in the foundation of any musical performance or interplay which our ears have experienced whether in the past, present, and indeed the future. Before the 20th century a musical performances were nothing more or less than musicians playing their acoustic instrument in front of an audience. This is true for solo, small chamber music ensembles, or even huge orchestras. The musical experience was always immediate and direct at the specific moment. Ideas of possibly recording, amplifying or modifying an instrument's sound were not existing, however, the invention of the dynamic microphone in 1876 [3] slowly initiated a new technical and later musical era. In the following paragraphs the author will review and categorize the most important milestones in electronic music processing.

1.1 History of analogue music electronics

Microphones are able to reproduce a precise electronic image of the sound wave's air pressure in order to process it electronically. The opposite principle of a microphone is the loudspeaker, which converts an electronic signal to a sound signal which our ears can listen to. A microphone connected to a loudspeaker via a allows us to talk into the microphone and listen to it where ever the loudspeaker is located. In a bidirectional manner this allows two way communication, which is nowadays known as the telephone principle. First generation loudspeakers did not generate high output gains so that a close distance to the ear was required and this quickly lead to the invention of headphones. Later in 1906 the next evolutionary step was taken by Robert von Lieben, who invented the tube amplifier [118]. Tube amplifiers were able to increase an electronic signal's amplitude by an adjustable value in order to trigger

larger loudspeakers to be used as sound sources in a room instead of simple headphone usage.

Whenever a signal does exist in the electronic domain it can be treated electronically and whenever any electronic treatment is processed, we are talking of "signal processing". In audio engineering the main goal of early signal processing was the improvement of better amplifiers utilized to increase the volume of a given signal in a natural and effective way in order to address larger audiences. Julius Edgar Lilienfeld invented the transistor in 1926, which finally lead to a miniaturization and optimization of the tube amplifier. In the meantime a special kind of microphone – the pickup – had been developed intensively. In comparison to the membrane microphone, which reacts to air pressure changes caused by any sound signal, it explicitly transforms the vibration of a metallic string into an electronic signal to become amplified by an amplifier [63]. In that context the electric guitar was born and very soon the first rock 'n roll bands with amplified gear were born. In contradiction to the typically sensitive but dynamic character of classical acoustic music, electric bands were often driven by less dynamic but high gained musical sequences. The escalation of this was the fuzz tone which consciously distorts the output of an electric guitar in order to produce an aggressive and energetic sound. Originally a similar effect was produced by guitarists overdriving their amplifiers with too high input gains, which generated the distortion inside the amplifier. Additional so-called effect devices such as chorus, flanger or reverb, followed quickly as the result of further analogue signal processing and artistic experiments. Even today those devices are put straight into the signal processing chain between the output of the electronic device and the amplifier and are typically used as foot pedals. One well known example is the Wah-Pedal in figure 1.1 invented by the Thomas Organ Company in the 1960s and intensively used by Jimmy Hendrix and many other famous players.

Figure 1.1: Original Wah-Pedal of 1967

Starting in the early 1960s, electronic music became increasingly popular, filling large concert halls with thousands of people, and hence the need for powerful multichannel amplification was created. In response systems with large mixing consoles became

the major technical solution. These were able to process numerous channels of audio, amplify them and address to a desired number of speakers. Such systems are generally called “PA” as an abbreviation for the term “public address” [63]. Later combinations of mixing boards and amplifiers were called power mixers [79]. Within the area of audio engineering the signal processing domain for live performances became a highly specialized discipline. Figure 1.2 shows an important milestone and probably one of the best examples for the ongoing optimization process of audio processing gear: The U87 microphone by Neumann Microphones Berlin has set a standard for high quality condenser microphones since 1960, and due to their unique sound, particularly for the recording of vocals, it can be considered the prime choice for sound engineers.

Figure 1.2: Neumann U87 mircrophone

In the performing arts, signal processing mainly aims at the instant sound generation and modification, and is often described with the word “Live“ in sound engineering language. In order understand the complex world of audio engineering, the author divides it into three application categories. Signal processing or “Live“ describes the first category, while signal reproduction and signal transmission represent the second and the third category. In order to avoid confusion, it should be noted that each category applies the same principles of signal processing, in order to retrieve a signal in the electronic domain. In the following paragraph the author will give attention to signal recording/reproduction and signal transmission.

“Signal reproduction” as the second discipline in applied sound engineering focuses on the recording and playback of a performed music piece. Since an input source is supposed to be conserved and played back later, a recording medium is placed in the middle of the signal chain, on which the acoustic information is stored [63]. The first medium for that purpose was the record with the corresponding record player, of which first working prototype was released in 1877 by Thomas Alva Edison and this became the major music playback medium until the 1990s [68]. Alongside the record player, the magnetic tape machine achieved a breakthrough in home entertainment,

allowing the home user to record data to individual tape cassettes. Later tape machines became the major technology in recording studios, where each instrument of a musical piece was recorded on a separate track [63]. Early machines were able to record up to 4 tracks while high end devices in the mid 1980s allowed more than 32 tracks a one tape. One of the most successful tape machine vendors was the company Tascam. Figure 1.3 shows a 4 track device from that time. After the recording it was the sound engineer's task to mix these tracks with an audio mixing console, add effects such as reverb or equalization and finally record the mix to the final 2 track stereo mix [63]. Besides "Live", the the area of audio recording represents the second major discipline in audio engineering. For the music industry the domain of recording has significant importance since sound studios can be considered as the base of the music production and distribution process.

Figure 1.3: Tascam four track analogue tape machine

The third major field in audio engineering is the transmission of audio signals, where – in contradiction to the process of audio conservation or recording – a generated electric audio signal is not directly attached to an amplification chain or a recording device. Instead it is used as an input signal for an electronic sender. This sender transmits the signal in the form of electromagnetic waves via the air to a receiver. This so-called radio transmission could cover distances of hundreds of kilometers and was first patented in 1892 by Nikola Tesla [26]. In 1897 the first radio stations were established in order to provide information and entertainment programs for a mass audience. Through radio transmission, music became available for any person at any time, and in the comfort of their own home. In fact music transmission has mainly been used as a unidirectional entertainment service, while bidirectional links were mostly used for telephone voice conversations only. Since telecommunication evolved to become a substantial part of people's every day lives, so-called public switched telephone networks (PSTN) became extremely reliable and qualitatively good, so

that audio engineers started using it for the transmission of bidirectional audio in the mid 1990s. In case the need for special music parts for a music piece was immediate, music tracks could now be exchanged through a high quality telephone connection between two sound studio venues.

Any of the mentioned disciplines in audio engineering has been based on analogue signal processing, where the electronic signal is a time continuous representation of the original sound wave. Though analogue electronics has changed our world tremendously and still plays an important role in our lives, across the years it has slowly been replaced by digital electronics, which process, store and transmit a signal not as is but rather by its time discrete numerical representation of zeros and ones [37]. Further relevant aspects related to analogue and digital signal processing will be discussed intensively in sections 4.1 and 4.2. In fact digital technology has advantages and disadvantages in comparison to analogue technology. Particularly with the rapid evolution of personal computers and the possibilities of miniaturization and lossless copy processes have allowed digital technology to take over is almost any field. The aforementioned record example has been replaced by the digital compact disc (CD) which in turn has recently been replaced by small MP3 [66] devices with capacities of thousands of songs.

1.2 History of IT music performances

When the first PCs became available to home customers in 1981 nobody could have imagined what impact these machine could have in general and especially on the world of music. The most famous representative of early 8 Bit home computers was the Commodore C64, which already provided first simple sound generators for hobby musicians. In 1985 the next generation of 16 Bit machines with regard to music was made of the Commodore Amiga or the Atari ST. They supported much higher processing power, already provided four track synthesized sound and especially the Atari ST intensively used the MIDI format (musical instruments digital interface) [29] by default. MIDI was invented in 1981 by Dave Smith for the Audio Engineering Society (AES) and quickly developed to a standard interface in sound triggering and controlling. Soon any professional music device such as keyboards, synthesizers or sequencers was equipped accordingly in order to be accessible for other MIDI devices. At that state of the late 1980s or the early 1990s the home computer was the perfect control device for triggering MIDI based music setups but, despite the technical possibilities, PCs lacked appropriate software to fulfill the requirements of music creation. Finally the program “Cubase” by Steinberg Software [12] took the leading role for that particular scenario as one of the first and famous music sequencing softwares with which whole scores could be written on and played back by any attached MIDI playback device. Vice versa, the MIDI notes of a played keyboard could be recorded and modified accordingly. Altogether the combination of the Atari ST with its inbuilt MIDI interface – as illustrated in figure 1.4 – and the Cubase

software became the standard for computer driven MIDI sequencing and controling in the professional and semiprofessional domain.

Figure 1.4: Atari ST computer with inbuilt MIDI interface

Over the years also the audio capabilities of the IBM personal computer (PC) evolved to an unforeseen high standard. The PC changed its profile from an electronic typewriter to a multimedia processing platform, now able to capture, store and playback real audio streams of high CD quality with modern sound cards, rather than just triggering external devices via MIDI. In parallel the Apple Macintosh followed the same approach of creating a versatile, flexible and userfriendly computer for anyone. At the end of the 1990s computers had reached such high speeds and storage sizes that the technology of harddisk recording slowly took over and replaced analogue tape devices and their digital successors, such as the Alesis ADAT [25] or the Tascam DA 88 [81]. With the introduction of professional sound cards, the PC theoretically became the "all in one" box for music producers which could now use their machine to compose, record, edit and master. In practice sound engineers still worked with a combination of the PC and highly specialized studio gear but the more the functionality became available via software, the less external devices were used.

Hand in hand with this ongoing process, aspects of networking and computer based communication started to draw more and more attention to PC users. In 1969 the Arpanet (Advanced Research Projects Agency Network) [147] came to life, which was basically intended for military use for the US ministry of defense and connected a few research institutes and universities in its first days. The Arpanet is the ancestor of our Internet, which equally was first used as a text based academic platform for the exchange of scientific data. Following the same shift from text based to multimedia based approaches of personal computers, the Internet tried to fulfill the new quality requirements in terms of much larger data capacities, bandwidth or speed. E.g. the official release of the MP3 file format in 1995 by the Fraunhofer Institute of Erlangen dramatically reduced the file size of audio tracks, which made them interchangeable by sending them across the Internet, so that the total delivery times could be reduced from hours to minutes. Starting with modem speeds of 28 kbps in the mid 1990s, it took less than ten years for home users to access high quality movies and run decent video conferences with DSL lines up to several Mbps. With special respect to the latter, the actual process in 2009 shows a consequent move to explicit Internet usage and hence, an even more consequent step towards digital technology. These

new possibilities of fast digital signal processing, data file exchange and high quality communication also influenced the music community. In fact, the existing structures of music creation and production were solid and well established and hence the music industry had no special interest to actually change them. However, some musicians and composers, especially in the area of new music, have been intrigued by using these new technological infrastructures experimentally and in particular the possibility of remote musical collaborations. Even before the development of the Internet, these ideas and visions became present. Their most famous representatives will be described in the following paragraph.

John Cage's "Imaginary Landscape No. 4 for twelve radios" [123] is considered to be one of the earliest networked music performance experiments. The piece used radio transistors as a musical instrument. Although the levels of interactivity were limited to the dialing of radio stations, gain and tone color, the desire to investigate the possibilities of cross influence in networked instruments is evident in the piece. Later on, the first group to experiment network practice with computers was the "The League of Automatic Music Composers" in the late seventies century. The group, originally composed of Jim Horton, Tim Perkis, and John Bischoff, started using networked computers to exchange messaging data between each other with the goal of influencing their playing. The group, later renamed as "The Hub" [47], started experimenting remote collaborations between the west and the east coasts of the US. Due to the limited bandwidth available at the time, the group exchanged musical control messages and not the audio signals themselves. In August 1983, Jean Piche, Osamu Shoji and Martin Wesley-Smith performed the piece "Night Satellite" [101], with the aim to create a multisite audio performance between Vancouver, Tokyo and Sydney through satellite links. In comparison to latencies of maximal 25 ms in natural music scenarios, delays of about 300 ms were achieved. Therefore, they were perceived as too high for a natural musical interaction. As a direct result musicians slowed down in terms of timing, which finally forced them to compensate the delay experimentally. A similar experiment was performed by Seiji Ozawa at the opening ceremony of the 1998 Winter Olympics [106]. The subject of this performance was to conduct Beethoven's "Ode to Joy" with choirs on five different continents. A huge arrangement of 200 singers each in Sydney, New York, Beijing, Berlin, False Bay and 2,000 singers at Olympic Stadium plus a conductor, 8 soloists, and orchestra in Nagano was established. Since the main technical goal of this performance was a signal synchronization at the Nagano stadium, a time lag adjustor was used for each remote location in order to eliminate the corresponding satellite delays.

Even though there were several initiatives to exchange audio data from one site to another in realtime, it was mainly the development of high speed over-provisioned Internet backbones such as Internet2 [7] in the US and GEANT [5] in Europe that showed the first potential of real time high quality interactive audio links: In October 1994 Eve M. Schooler et al. held an Internet-based music performance at the 2nd ACM Conference on Multimedia Computing in San Francisco, USA. The group cre-

ated and synchronized three realtime streams of music from different Internet hosts with delays in the order of 200 ms. Though the audio quality reached an acceptable level, it proved difficult for the players to perform [131]. Later, in February 1996, Paul Hoffert demonstrated the "Cyber Soiree" which was another Internet-based four-way jazz performance between cities in Canada and the US. Here the performers suffered from more than 500 ms delay, however, after extensive practice slightly better results could be achieved. In the same year a different kind of remote music experiment by Dimitri Konstantas et al. was accomplished. A conductor was located in St. Augustin, Germany and a chamber music ensemble was located in Geneva, Switzerland. Due to the comparably small local distance, one-way delays of 80 ms could be achieved and after some trial and error, it became possible to make the conductor lead the ensemble into a playable situation [93]. From the year 2000 on, several realtime audio streaming acts were performed but perhaps the most important and consequent initiative to take advantage of the implementation of Internet2 was the "SoundWire" project, led by Chris Chafe at Stanford's CCRMA [11]. The project started in the year 2000 and developed a command line based Linux [136] application for high quality uncompressed bi-directional audio streaming. Apart from that and in terms of video streaming the McGill Ultra Video Conferencing research group [13] introduced low latency uncompressed video with first successful performances in 2001.

In any case it was now possible to transmit sound with at least an acceptable or even excellent quality, however, due to the inherent delays, in none of the tests musicians could perform as if in the same room unless free experimental music was played. This might be the reason that after 2003 the interest in network music performances decreased significantly. Recently, since 2007, a revival of interest for the subject has emerged not only on the technological but also on the cultural side. Researchers are seeking to understand the cultural implications of providing such facilities to musicians and producers as well as seeking ways to increase the level of interactions between musicians collaborating over a network connection [47].

1.3 Motivation

As previously described, modern technology has changed the music world in numerous ways. Effect devices or software synthesizers help to create a modern and interesting sound, complex harddisk recording allows to run a whole CD production in private home studios on one single PC, and apart from that the PC can even be used as a telephone substitute. In this context it is further possible to provide web radio streams with one's own music. There is no doubt of the positive aspect in using current technology but from a more critical point of view there are a significant drawbacks relation to live music. Before high quality recording became available to home consumers, musicians relied on each other and had to find potential band colleagues to actually perform their music. Since the days of sequencers, samplers and recording systems, today this statement seems no longer valid. On one hand musicians

are now able to produce their own music by themselves in their own environment but on the other hand it isolates them in the same way, which could in turn lead to social and artistic separation.

Also it can be observed that the main focus of music applications is put on recording and musicians are mainly motivated to build their own home studios based on a particular recording software, whereas aspects of live signal processing and transmission play a secondary role. In fact, so-called VST plugins [153] provide real time sound effects but they are basically intended to work on the recording score rather than for live music play. Audio streaming and distribution is rarely supported and requires additional soft- and hardware. On that point, it becomes obvious that the three areas of sound engineering are still very separate, though all of them can be processed with a standard PC and an Internet connection. Despite this fact there are many examples in the music industry where technology has had little impact, e.g. the classical music genre. Due to a variety of factors, it appears that musicians haven't advanced in line with the available technology, rather they still play and rehearse as they did a hundred years ago.

Given the numerous disadvantages, drawbacks, or lack of current technology in music, it is obvious that there is potential for improvement, which the author tries to approach in this work. In that context the highest challenge and next evolutionary step would be the instant musical interaction between the connected musicians in realtime. At the end, we could imagine a Europe-wide band that could play together, not depending on the distance, local scenes, mentality, nationality or language. Theoretically this and could bring together different cultures and styles in order to create new and interesting music. It could also be a platform, in which musicians can get in touch with each other and establish new contacts, similar to a jam session in real life where people meet and share contacts in order to found new bands. However, this principle of musical interaction would not be restricted to music genres since it provides the same scenario as if in the same room, not distinguishing between pop-, rock-, jazz- or avantgarde musicians, so that even classical musicians could be motivated to use such technology.

In fact, playing alone, connected via a low delay connection, can be compared to a phone call rather than of a real meeting in person and thus lacks certain aspects of feeling and perception, however, it should be pointed out that this principle aims not at the replacement of the conventional scenario, rather it approaches a prevention to the home musician's isolation and is intended to make life easier for professional musicians. For example, traveling musicians could still participate and rehearse in their home projects despite not physically being in the rehearsing space which consequently makes them much more flexible and available for a wider range of music projects.

Another motivation, and fascination, lies in the fusion of technical and musical aspects and the strong interdisciplinary character, which demands both widespread general

knowledge in specific fields and highly detailed and specialized skill base. Without knowledge of the music world the technical requirements cannot be determined, and without knowledge of the technical world it is not possible to assess and possibly fulfill those requirements and actually build a working system. Finally, a system, which really complies with the conditions and restrictions of a conventional music scenario, would be an important step towards the development of the Web 2.0 [24], which aims at the evolvement of socio-cultural and communication aspects, and hence this technology could have a significant impact in the area of new web based principles.

1.4 Goal and structure of this thesis

Numerous network music performances regardless of the underlying network technology have been performed already, and despite some promising results, a precise and detailed evaluation of the actual technical and musical implications barely happened. The total delay is often confused with the pure network delay and the implication of the music styles is often underestimated. Any network music related project still remains in an experimental state and altogether there is no clear and precise elaboration which covers all of the relevant musical, technical or interdisciplinary aspects of remote musical interaction. In combination with the high complexity in current audio and networking technology, such an elaboration is mandatory.

The general goal of this work is a requirement determination and the actual achievement of a system, that allows Internet based realistic network music performances. Hence, the main focus lies on existing current technical principles and restrictions with respect to the transmission of low delayed audio data. In that context a parameter approximation and optimization of how a minimum of latency can be reached for any involved component will be proposed.

The remainder of this thesis is structured in such a way, that in chapter 2 the basic principles of human audio perception and their meaning in terms of a musical interaction are outlined. Based on these fundamentals, in chapter 3 the author presents his results of numerous cognitive experiments with professional musicians in order to answer significant questions regarding delay and musical interaction. Chapter 4 gives an overview of the technical basics and concepts, which have relevance in general and in particular on the Internet. This implies aspects of digital signal processing and network engineering likewise. With respect to the so far covered cognitive and technical facts, the author introduces a taxonomy of various musical interaction approaches in chapter 5. Furthermore, the related work is embedded into this chapter by a classification according to the proposed categories.

Although the transmission of realtime data on the Internet is a common practice in Internet telephony, it does not fulfill the requirements for realistic music interaction and hence chapter 6 firstly introduces an own optimized low delay audio streaming concept. According to this concept the author describes the actual implementation

and evaluates it in realistic test scenarios with a number of european endpoint connections. The results of chapter 6 represent the basis for the new sound- and network engineering approaches, described in chapter 7. Finally, chapter 8 summarizes and discusses the results of this elaboration in order to outline the research and development potential for the future.

Chapter 2

Fundamentals of audio perception

Any ability such as dancing, writing, cooking or painting requires a natural talent, training, and addresses at least one of our five senses as a basic precondition. Painters and phytographers need a good eye, cooks have to smell and taste and medical doctors or technicians have to feel various temperatures. In music it is the ear which acts as the main sense of information input and without the ability of hearing, musicians could neither listen to their own music nor respond to others. Hence, as a preliminary and fundamental step towards a distributed music system, this chapter investigates the main basic principles of human hearing, psychoacoustic effects and music cognition, which finally leads to musical interaction. This is, in particular, of major importance towards determining the fundamental requirements of a usable system.

2.1 Basics of sound signals

The ear is the human being's interface for the exchange of audible information and in order to explain its principles and functionality it is important to clarify, what precisely audible information is. Audible information, whether spoken word, the buzz of a machine, or a piano chord, is sound and sound always originates from the motion or vibration of an object [113]. This motion is impressed upon the surrounding medium as a pattern of changes in pressure. In our daily lives this typically happens via the air and what actually happens is that the atmospheric particles are squeezed closer together in a process called condensation. Afterwards they are pulled farther apart in a process named rarefaction [113]. Though the sound wave moves outward from the vibrating body, the involved molecules do not advance with the wave. They vibrate about an average resting place instead and the vibrations occur along an axis that is aligned with the direction in which the sound is propagating. This wave form is known as a longitudinal wave. The further a sound wave moves, the weaker it becomes, plus its actual characteristic might be modified due to reflections and overlaps of other sound waves. Hence, the sound image reaching the ear differs somewhat from that initially generated.

One of the simplest and most basic types of sound is the sine wave, also known as a sinusoid. Figure 2.1 shows its waveform, which describes the pressure variation

plotted against time. In this context it is important to mention that due to the dependency of time, this graph uses the so-called time domain presentation [151]. How this pressure variation would look if the air was visible to our eyes, is shown in figure 2.2. This sine wave is simple both from the physical and mathematical point of view. Sinusoids have a very clean or "pure" sound, similar to a tuning fork. Thus they are also called simple tones or pure tones. A sinusoid can be fully described by the frequency, the amplitude and the phase. The graph also shows the term period, which describes the amount of time between two successive maxima (or minima). This is the time for one complete cycle. Hence, the frequency is the reciprocal of the period and states the amount of cycles per second. It is specified in Hertz, where

$$1[\text{hertz (Hz)}] = 1\text{cycle/s} \tag{2.1}$$

Sound frequency and the perceived pitch are typically in direct relation – the higher the frequency of the sinusoid, the higher the pitch of the sound to our ears. The lowest audible frequency human being's ears can detect is determined at 20 Hz while the highest audible frequency varies with the age [103]. At a very young age children are able to hear up to 20 kHz but depending on how much loudness the ear undergoes over the years, this upper limit can decrease down to 15 kHz for persons in their mid 20s. Beyond 60 years this upper threshold typically resides below 10 kHz. Assuming a constant frequency, the perceived loudness is directly linked to the amplitude, which describes the amount of pressure variation about the mean. Our ears perceive a larger amplitude as a louder sound as opposed to a sound with lower amplitude. The ear's audibility threshold resides at a sound pressure of 20 μPa (micropascals) for a 2 kHz sound [77]. This is the reference value for the sound pressure level (SPL) measured in the logarithmic unit decibels (dB). Normal conversations reach an SPL of about 40 to 60 dB and the threshold of pain resides at 130 dB, which is louder than strong thunder. In terms of sound localization or the comparison of two sounds the phase has to be taken into consideration. It is described as the portion of the cycle through which the wave has advanced in relation to some fixed point in time and is specified in degrees. Relevant details regarding frequency, amplitude and phase will be covered in context with our auditory system later in this chapter.

Though the previous explanations already give an idea of how sound is generated, this chapter needs to further outline theory and principles in order to cover relevant future aspects. In the following the author will introduce the term signal and the sinusoidal wave already is the first example of a such. A signal is defined as any physical quantity that varies with time, space or any other independent variable [100]. Mathematically, we describe a signal as a function of one or more independent variables. The equation's actual value depends on the actual moment of time t and further involved variables. For this sinusoid the amplitude is a function of time t and the sine's frequency f as described by the following mathematical equation:

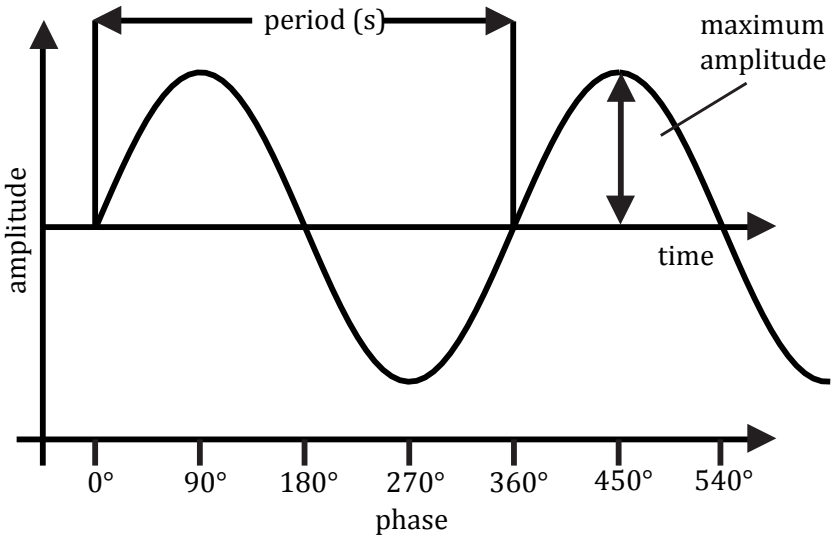


Figure 2.1: Sine wave

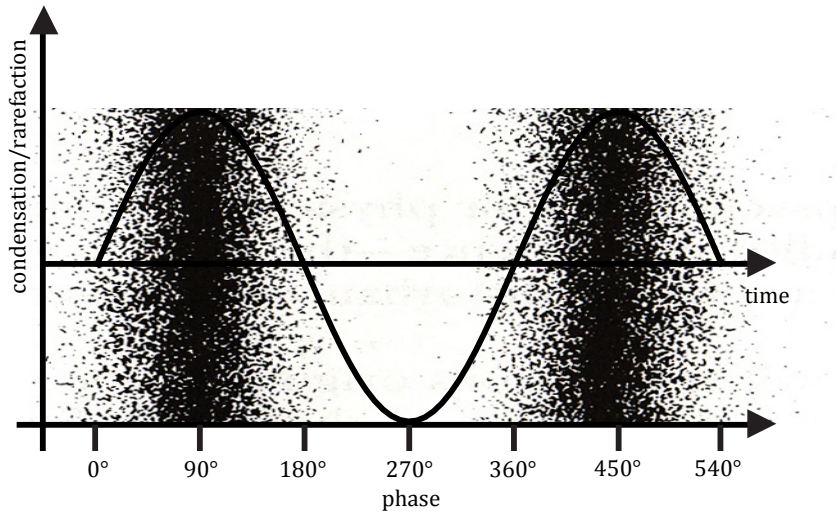


Figure 2.2: Sine wave in the medium air illustrated by air pressure variation

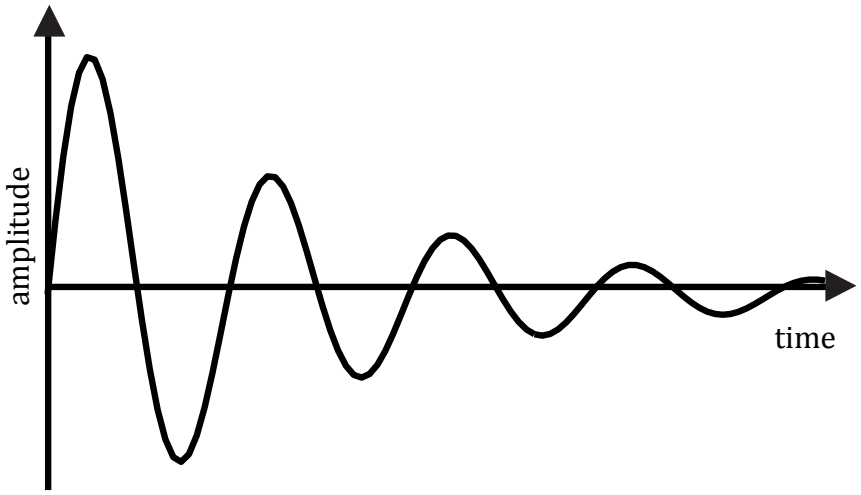


Figure 2.3: Damped sine wave

$$y(t) = sin(2\pi \cdot ft) \tag{2.2}$$

Signals can be of various kinds. Whenever a signal is of a sinusoidal kind and so can be described by its frequency, amplitude and phase, this signal can be considered as a harmonic signal [111] as is our presently considered case. In turn, signals which have a non sinusoidal character and require more complex descriptors, are called non harmonic and will be explained later. The sinusoidal also represents a periodic signal [111], whose duration continues and repeats endlessly in contradiction to an aperiodic signal, which does not repeat regularly over time. A special case of of the periodic sinusoidal vibration is the damped sine, which starts vibrating in the same way but its amplitudes lowers with time until it ends. The aforementioned tuning fork produces exactly this damped sinusoidal oscillation which is illustrated in figure 2.3 and described in equation 2.3, where d represents a decay constant indicating how fast the signal is damped to zero after its generation.

$$y(t) = e^{-dt} \cdot sin(2\pi \cdot ft) \tag{2.3}$$

The reason for the decreasing amplitude over time is contingent to the fact that the fork is stimulated with a single hit and is not constantly stimulated with energy. While the fork oscillates, it looses energy until the vibration stops. This is equal to the process of plucking a guitar string and in fact this is a more realistic example for a music signal than the pure sinusoidal, which vibrates endlessly with the same amplitude.

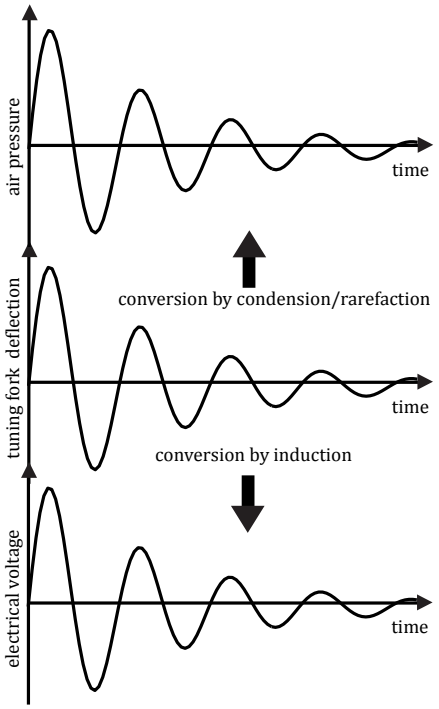


Figure 2.4: Conversion of the mechanical fork oscillation to air pressure and voltage

If the tuning fork was used in a room close to a busy street, its generated signal would be mixed with the background noise of cars and pedestrians. Since the former is the desired information and the latter the disturbing part, we also talk of the wanted signal (sound of the tuning fork) and the disturbing signal (background noise) [111].

Another differentiator is the medium, through which a signal is present. Due to a high variety of signal representations, the distinction is typically done between an electronic or a non electronic signal [63] but particularly in context with the process of hearing. The author will later discuss additional signal representations such as mechanic or hydraulic ones. The tuning fork produces a non electronic signal, since the waves are being transmitted through the air by changing its pressure over time. However, in another setup the fork vibration can also produce an electronic signal: An electronic conductor moving within a magnetic field generates an electrical voltage by the principle of induction [63]. The same effect happens by moving the magnet instead of the conductor. As a third case both – the conductor and the magnet – remain still and the magnetic field is changed by the tuning fork's oscillation. Now the generated voltage follows the fork's vibration in precisely the same way, which creates its electronic signal representation. This principle is illustrated in figure 2.4.

The following paragraph is based on explanations in [129], [151] and [63]: A generated damped oscillation can be clearly described and calculated depending on the energy,

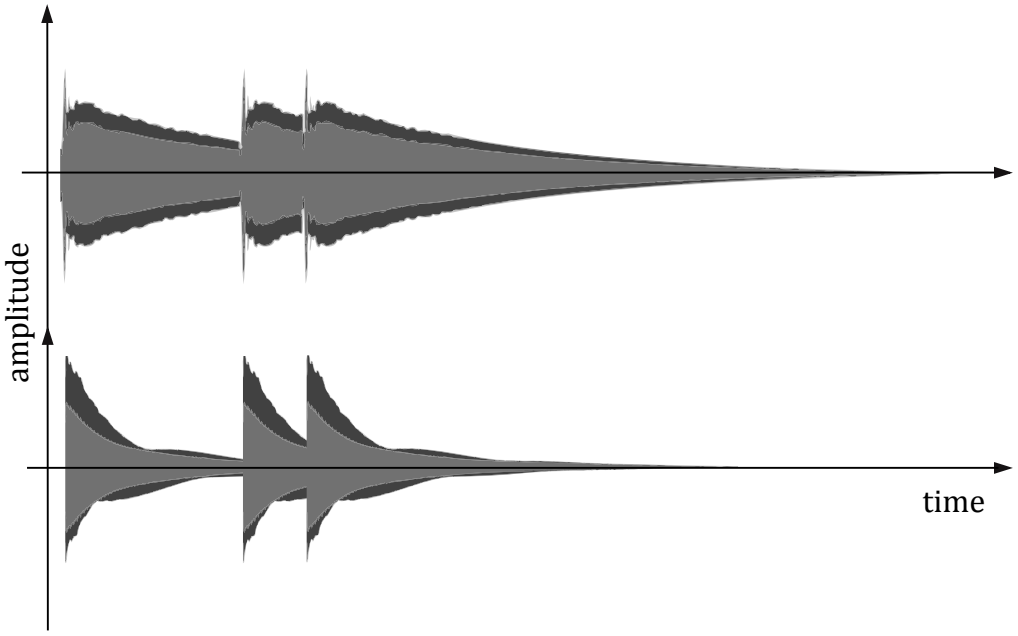


Figure 2.5: Tuning fork and bass guitar played in a simple rhythmical pattern

with which the fork was stimulated, which is why it is not of random kind. Hence, the result is a so-called non stochastic signal [151]. If the fork was triggered by a person over a random amount of time with a random number of hits of random energy, this result would be a stochastic signal. By principle music and speech are stochastic signals and with respect to our example, if this person was a musician hitting the fork in a rhythmical and musical manner, a simple percussive music piece would be the stochastic signal result. As a summary of the former explanations this stochastic signal needs to have a low disturbing signal and a strong wanted signal which is of non electronic, aperiodic and harmonic kind. The upper part of figure 2.5 shows about 10 seconds of the recorded tuning fork. Due to a low zoom factor, it not yet possible to see the precise sinusoidal vibration but each hit with the corresponding attack and decay [63] of the signal are clearly visible.

Repeating this experiment by plucking the string of a bass guitar also shows the same rhythmical pattern but the wave itself looks different as the lower part of figure 2.5 shows. By zooming into the recorded signals it is possible to observe the precise waveform – the result is illustrated in figure 2.6. The tuning fork in fact vibrates in a sinusoildal manner, however, the bass guitar signal shows a totally different wave form. Hence, there must be other additional signal parts, which make the signal look and sound different than the one of the tuning fork.

In fact the bass guitar string does not only vibrate in a simple sinusoidal manner. Instead it also vibrates in parts of $\frac{1}{2}$, $\frac{1}{3}$, $\frac{1}{4}$ and possibly further parts of the string as

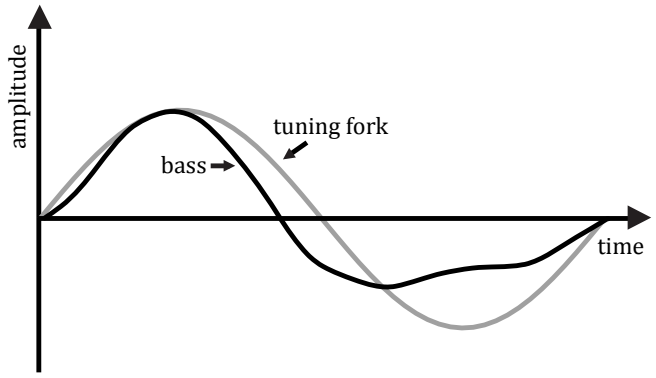


Figure 2.6: Zoomed signal of a tuning fork and the author's bass guitar

if the fret on the neck of the instrument was held at those spots and hence the signal we hear and see is the sum of them [103]. These additional tones are called harmonics [103] if their frequencies are integer multiples of the fundamental or inharmonics [61] if their frequencies are non-integer multiples of the fundamental. The term overtone [63] combines both of them. Overtones change the signal from a harmonic or simple waveform to a non harmonic or complex waveform and their weight in terms of audible presence is different for any type of instrument. The actual constellation of overtones finally creates the instrument's individual sound character and is called the spectrum [117] of the instrument. In order to visualize this spectrum in a convenient and clear way a theorem by Fourier [113] can be applied which tells that any waveform can be analyzed or broken down into a series of sinusoids with specific frequencies, amplitudes and phases. On one hand this means that any waveform can be built by the use of various sinusoids of various frequencies. On the other hand this means that any signal can be represented in a different manner, in which the present sinusoidal frequencies and their weights are shown. In contradiction to the time domain this form of presentation is called the frequency domain [151]. In order to present a signal in the frequency domain, the so-called Fourier Transformation has to be applied as described in equation 2.4. The term $f(t) \cdot e^{-j2\pi vt}$ is responsible for the transformation into the frequency domain. Due to the integral with the continuous time as the running index every signal's instant can be observed respectively. As a result F(v) holds the signal's containing frequencies in a complex representation. This, however, does not lead to a descriptive graphical presentation of the signal's frequency distribution over the total frequency range.

$$F(v) = \int_{-\infty}^{\infty} f(t) \cdot e^{-j2\pi vt} dt \tag{2.4}$$

Hence, $F(v)$ must undergo a further calculation in order to retrieve the so-called power-spectral-densitiy (PSD), which delivers the desired representation: The PSD describes how the energy of a signal is distributed with frequency. The spectral density $\Phi(v)$ of the signal is the square of the magnitude of the signal's FFT as shown in equation 2.5. The term $\frac{1}{\sqrt{2\pi}}$ represents an additional normalization factor. This elaboration continuously uses the PSD in terms of signal spectrum representation.

$$\Phi(v) = \left| \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(t) \cdot e^{-j2\pi vt}\, dt \right|^2 \tag{2.5}$$

Respectively the lower part of figure 2.7 shows the bass guitar signal in the frequency domain. The lower part of figure 2.7 shows the one of the tuning fork after applying the Fourier Transformation. If desired it is vice versa also possible to recalculate a Fourier-tranformed signal back to the time domain by applying the inverse Fourier Transformation (IFFT) equation as described in equation 2.6. Here the term $F(v) \cdot e^{j2\pi vt}$ is responsible for the signal's transformation back into the time domain, while the integral's index now represents the signal's frequencies accordingly.

$$f(t) = \int_{-\infty}^{\infty} F(v) \cdot e^{j2\pi vt} dv \tag{2.6}$$

Though every instrument has its own individual spectrum, certain genres of musical instruments have at least similar spectra, which is why a piano is typically identified as a piano and a guitar is typically identified as a guitar. Apart from the characteristics of conventional music instruments other basic wave forms do exist, which matter in terms of sound synthesis and in general signal processing likewise. Synthesizers [79] use various types of those basic wave forms in order to compose new artificial sounds with them. Figure 2.8 shows the sine wave, the rectangle and the triangle in time domain and in frequency domain respectively. Additionally the signal and its spectrum for white noise and a short sinusoidal tone are given. It becomes obvious that a simple sine wave shows just one single frequency in the frequency domain. Additionally the rectangle and the triangle include further countable overtones: Both signals consist of the fundamental sine wave and unequal harmonics (first overtone, third overtone, fifth overtone, etc.), however, their weights are different. Equation 2.7 and 2.8 show the rectangle's and triangle's mathematical description.

$$\mathrm{rect(t)} = \sqcap(\mathrm{t}) = \frac{4}{\pi} \cdot ((\frac{1}{1})^2 \cdot \sin(\omega \mathrm{t}) + (\frac{1}{3})^2 \cdot \sin(3\omega \mathrm{t}) + (\frac{1}{5})^2 \cdot \sin(5\omega \mathrm{t}) + ...) \tag{2.7}$$

$$\mathrm{tri(t)} = \wedge(\mathrm{t}) = \frac{4}{\pi} \cdot ((\frac{1}{1})^2 \cdot \sin(\omega \mathrm{t}) - (\frac{1}{3})^2 \cdot \sin(3\omega \mathrm{t}) + (\frac{1}{5})^2 \cdot \sin(5\omega \mathrm{t}) - ...) \tag{2.8}$$

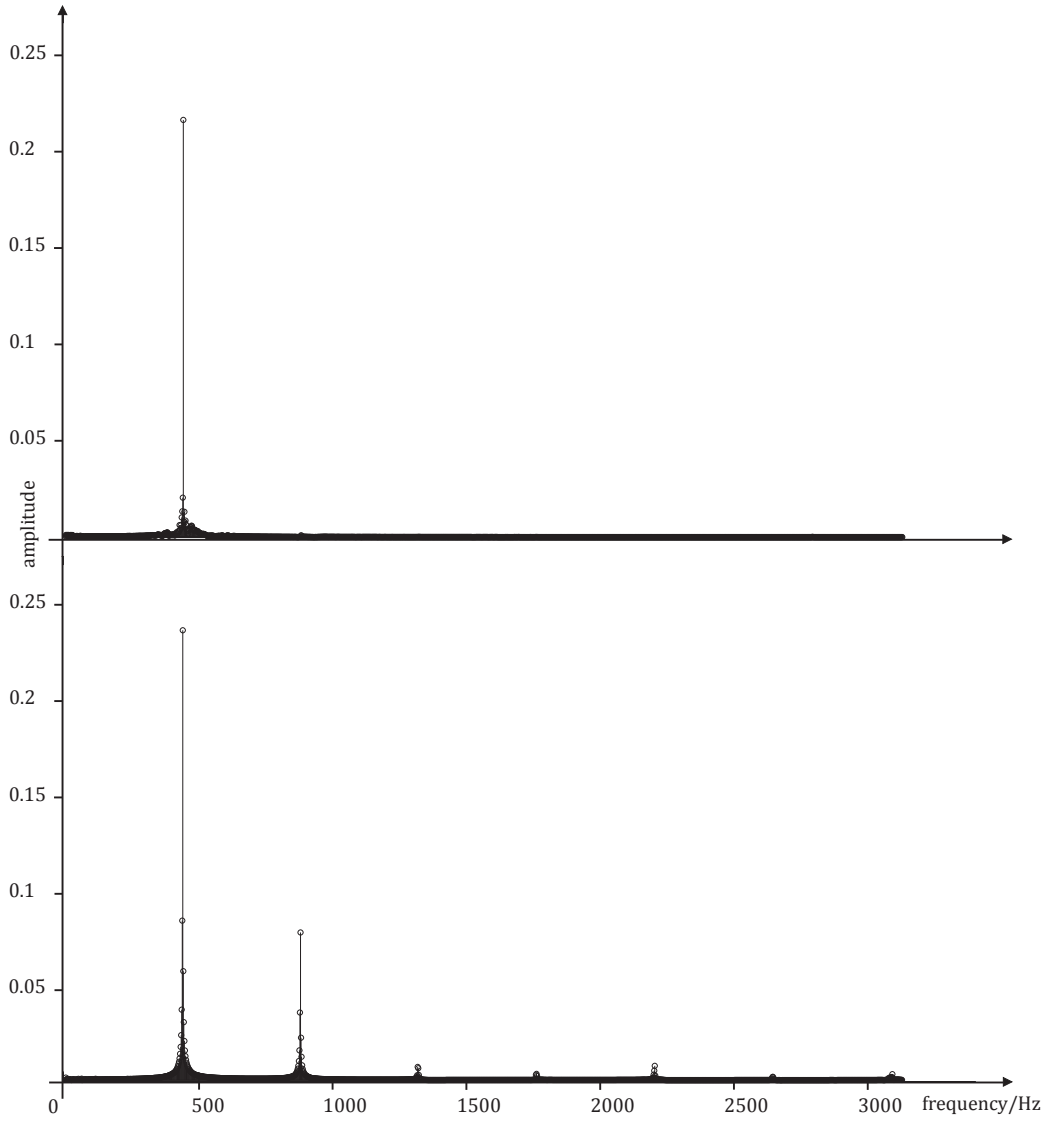


Figure 2.7: Spectrum of the tuning fork (upper part) and the bass signal (lower part)

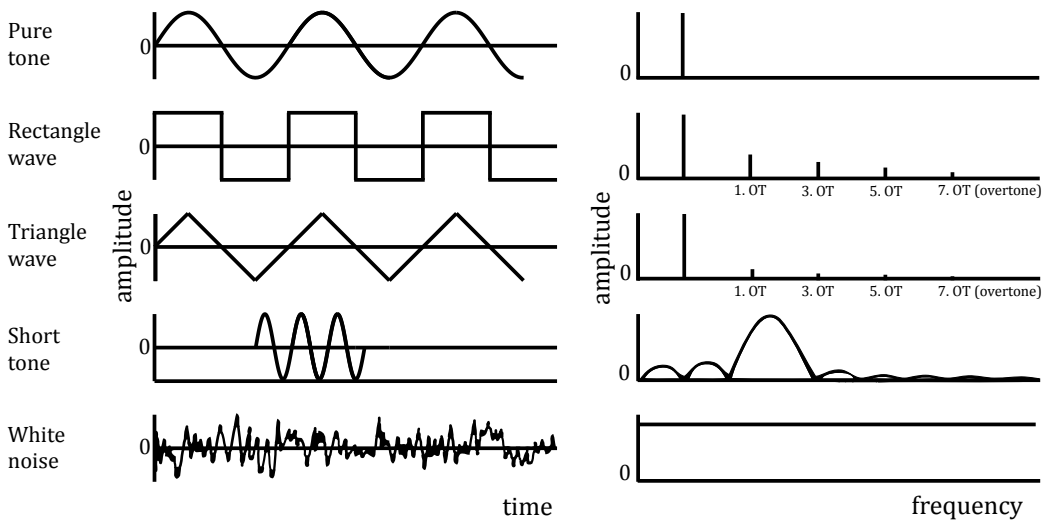


Figure 2.8: Basic sound signals [113]

Comparing the spectrum of the sine, rect and triangle with the spectrum of the white noise and the short tone, it is clear that the former consist of numerous lines as expected, while the latter has a continuous character. Hence, in the former case we speak of a line or partial and in the latter case of a continuous spectrum [151]. The continuous spectrum consists of noise components continuously distributed over the total audible range, which appear to the stochastic signal character: In case of white noise frequencies are generated randomly anyway – in case of the short tone the moments of time, at which the tone is stochastically switched on and off, are taken into consideration. In contradiction a line or partial spectrum consists only of the discrete frequencies a non-stochastic signal is supposed to have.

2.2 The process of hearing

Apart from sound signal analysis it is important to understand how our ear responds to sound and how the process our hearing actually works. In the following the author will describe the ear's anatomy, its pathways and and how sound is processed. A look at figure 2.9 illustrates the division of the hearing mechanism into three parts – the outer, middle and the inner ear [108]. The outer ear comprises protrusions at the sides of the head, a sound canal and the eardrum membrane at the end of the canal. The middle ear is more complex and consists of an air filled space with a link to the human's throat, the Eustachian tube and the ossicles. It consist of a chain of very tiny bones – the malleus, the incus and stapes. With approximately 3 mm^2 the stapes is the smallest bone in the human body [32]. Following up, encased in a bony labyrinth,

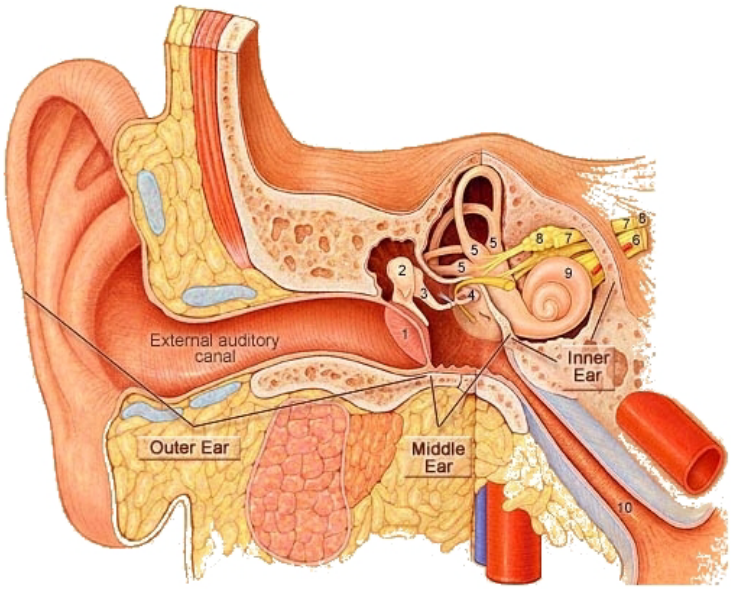


Figure 2.9: Profile of the human ear [21]

the inner ear is the most complex part of the ear. It contains the semicircular canals with the vestibular nerve, the auditory nerve and the cochlea. In contradiction to any other part of the ear the Eustachian tube in the middle ear and the semicircular canals of the inner ear have no impact on the process of hearing. The Eustachian tube's responsibility lies in the pressure equalization between the middle ear and the atmosphere. Normally this tube is closed but if the environmental conditions require a pressure equalization, it opens for a short amount of time. The semicircular canals are continuous with the cochlea and form a series of fluid filled tubes that are the primary receptor for the sense of balance which is linked by the vestibular and the facial nerve [103].

When a sound wave reaches the outer ear, it enters the auditory canal until it reaches the eardrum membrane. At that point the sound wave is still an acoustic signal transported through the air. The eardrum membrane receives the sound wave's air pressure with its actual frequency and intensity and triggers the malleus of the middle ear accordingly which in turn triggers the incus and the stapes. This way the acoustic signal is converted into a mechanical signal, whose output is provided by the stapes, vibrating in an opening in the cochlea, called the oval window [145]. The oval window is the interface to the inner ear and the cochlea with the so-called organ of Corti represents its main part. The organ of Corti is illustrated in Fig. 2.10. The cochlea is made of a spiral-shaped structure containing the basilar and the tectorial

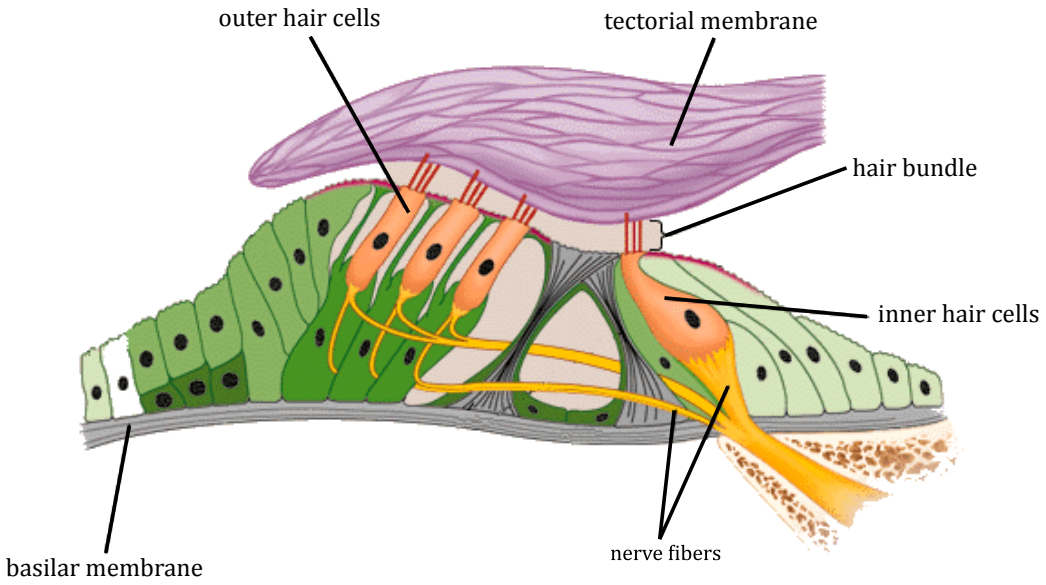


Figure 2.10: The organ of corti [16]

membrane, a fluid and fine hair cells. These cells have stereocilia or "hairs" that stick out. Their bottom is attached to the basilar membrane and the stereocilia touch the tectorial membrane. The stape's movement results in appropriate pressure changes in the cochlea's fluid which in turn causes the basilar membrane to vibrate. Here the signal has again passed a conversion from the mechanical impulses of the stapes to hydraulic energy in the cochlea's fluid. The resulting vibration creates a shearing force between the basilar membrane and the tectorial membrane. As a result the hair cell's stereocilia bends back and forth. When the stereocilia are bent, an electromotile response occurs. This way the last conversion from a hydraulic signal to an electric signal is being processed. Auditory nerve fibers rest below the hair cells and pass these electric signals on to the brain. So, the bending of the stereocilia is how hair cells sense sounds. Altogether, within the various stages in the process of hearing, the signal undergoes conversions from acoustic to mechanical to hydraulic and and finally electrical energy [108].

2.3 The ear as a system

The ear can as well be separated in a simpler way - the conductive portion, consisting of the outer and middle ear, and the sensorineural portion, consisting of the inner ear and the auditory nerve. Other than the previously described process of hearing via air conduction, it is also possible to bypass the conductive part by vibrating the skull mechanically and stimulating the inner ear's cochlea directly. In this way the sound is

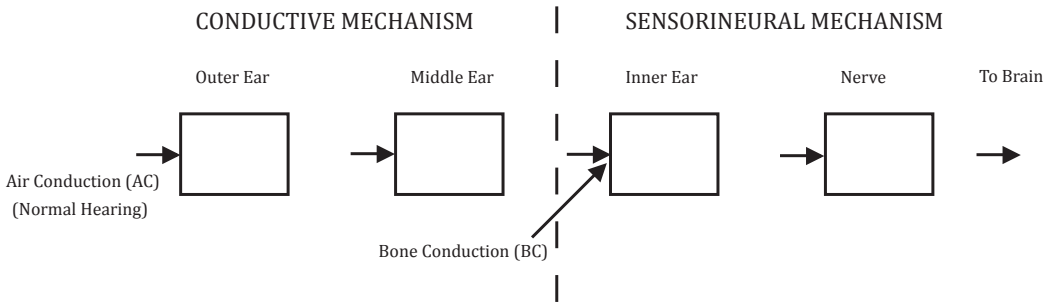


Figure 2.11: Block diagram of the ear

heard by bone conduction. This principle and breakdown to the mentioned sections is illustrated in a block diagram of figure 2.11. For people with hearing loss, medical doctors typically place a tuning fork on top of a patient's head in order to test the inner ear's efficiency. If the patient hears the sound, the sensorineural part works properly [108]. However, in terms of sound perception and analysis, the relevant signals are typically transmitted via the air and hence received by air conduction, so that bone conduction plays no significant role in this context.

As the block diagram shows, the auditory system includes a series of stages. The output of a given stage forms the input to the next stage. In a theoretical abstraction each of the stages can be considered as a device or system with a signal input and a signal output. A system processes an incoming signal in a specific manner and finally produces a signal output depending on the system's characteristics. Whenever we talk of signals and systems we deal with the area of signal processing and in that context a certain kind of system is of fundamental importance – the LTI system (linear time invariant system) [138], which will be described as follows.

For LTI systems the conditions of linearity and time invariance have to be fulfilled. In terms of linearity, two significant relationship conditions between the input and output must hold true. Firstly, if the input's amplitude to the system is changed by a factor k, then the output should also change in amplitude by the factor k. This condition is called homogenity. For example, if the input is doubled, then the output is also doubled, but without changing the output signal's characteristics. Secondly, the output of the system in response to a number of simultaneous independent inputs has to be equal to the sum of the outputs that would have been obtained if each input were presented separately. If the response to input A is X, and the response to B is Y, then the response to A and B together is X+Y. This condition is called superposition. As mentioned earlier, LTI systems also assume time invariance. This means that the input/output function does not change over time and remains constant. If the input is I and the output O, the relationship between the input and the output would be

$$O = c \cdot I \tag{2.9}$$

where c is a constant that does not vary with time. When a sinusoid is used as an input to a linear system, the output is a sinusoid of the same frequency, however, the amplitude and phase may be different from those of the input. Assuming, that the input is a single sinusoid, whose wave form is a function of time $I(t)$, it can be described by

$$I(t) = A \cdot sin(2\pi ft) \tag{2.10}$$

where A is the peak amplitude of the input and f is the frequency in Hz. The output as a function of time $O(t)$ can then be represented by

$$O(t) = G \cdot A \cdot sin(2\pi ft + \theta) \tag{2.11}$$

where G is a constant representing the amplitude ratio between the input and the output. It is also known as the gain factor of a system. θ is a constant representing the phase shift between the input and the output signal of the system. If the input is sinusoidal but the output is of non sinusoidal character, the system has somehow added additional frequencies, which are called distortions. A system that distorts a sinusoidal input to a non sinusoidal output is a non-linear system [113]. In that context it is important to mention, that with respect to complex wave forms an LTI system can still generate signal modifications: In LTI systems the gain factor and phase shift constants can vary with the input's frequency. Hence, an LTI system does not assume the same gain factor and phase shifts for signals all over the whole frequency range. Whenever a sine signal is used as the input to an LTI system, the output will always be a sine signal but its amplitude and phase will depend on its actual frequency. For example, if an LTI system is fed with an input sine wave of 440 Hz, it amplifies it with a gain factor of 2, while a sine wave of 880 Hz is amplified only by a factor of 1.5 but both signals keep their sinusoidal character. Hence, for complex waveforms, such as a rectangle or a triangle, LTI systems do not necessarily provide the same output waveform. As mentioned previously, complex waveforms can be considered as the sum of various sinusoidal waves of various frequencies and if some of these frequencies might undergo different amplifications or phase shifts, the final output signal contains a slightly different composition of sinusoidal waveforms.

The "system ear" is supposed to receive and interpret environmental sounds as naturally and unmodified as possible on one hand but should remove or suppress useless information on the other hand. As a result the ear is an LTI system with a varying gain factor over the total auditory spectrum.

2.4 Psycho acoustic effects

The previous sections explained sound and the hearing process with its involved components as a physical phenomenon, which finally stimulates the auditory nerve and transmits electrical data to the brain. Here, the information is interpreted and finally perceived as sound, however, one might question if this interpretation absolutely represents the original signal or if it underlies any modifications. Due to the complex structure of the human's brain, it is often impossible to retrieve adequate measurement data and hence the examination of perception requires a different approach. In general experiments in psycho acoustics work with numerous test persons, who individually respond to certain sounds in various test scenarios, which tells how they actually perceive the test signals. Finally, results are generated by statistical means [110]. This perceptual phenomenon of sound is subject of psychoacoustics and this section in particular. However, the area of psycho acoustics represents an extremely complex research field with a huge variety of directions and orientations. Hence – especially for this elaboration – it is required to narrow down relevant and fundamental aspects, which matter in terms of practical music creation.

2.4.1 Perception of pitch

In general pitch can be considered as the sound quality most closely the frequency of a pure tone [113]. A high frequency sound is perceived as high pitch and a low frequency sound as low pitch, however, their relationship is not a simple linear one. Pitch is a function of the fundamental frequency and in order to clarify this relation, the unit "mel" has been introduced. As a reference the pitch of a 1,000 Hz tone at a sound pressure level of 40 dB SPL (sound pressure level) has been given a fixed value of 1,000 mel, and the following phychophysical scaling technique has been used to determine the number of mels that are associated with different frequency tones: Numerous subjects were presented with this 1,000 Hz tone and the information, that the actual tone equals 1,000 mel. Each subject was then asked to manipulate the perceived pitch to multiples of the 1,000 mel reference tone – up to 2,000, 3,000 and more and down to 500, 250 and less – and finally the related frequency was measured. Figure 2.12 shows that pitch and frequency are not related in either a linear or a logarithmic fashion. Their relationship is rather complex [103].

In addition to that the sound intensity has an impact on pitch perception. Firstly, although sounds have specific pitches within the frequency range – as shown in the previous graph – it is not possible to make equal discriminations between different frequencies over the whole range. By measuring the difference threshold (limen) or smallest frequency difference for which two pure tones can be discriminated, and repeating this measurement for a large set of reference tones, it could be observed that from 0 to 3 kHz the difference threshold is approximately 1 Hz to 3 Hz [103]. The louder the stimuli, the better subjects were able to distinguish between two different

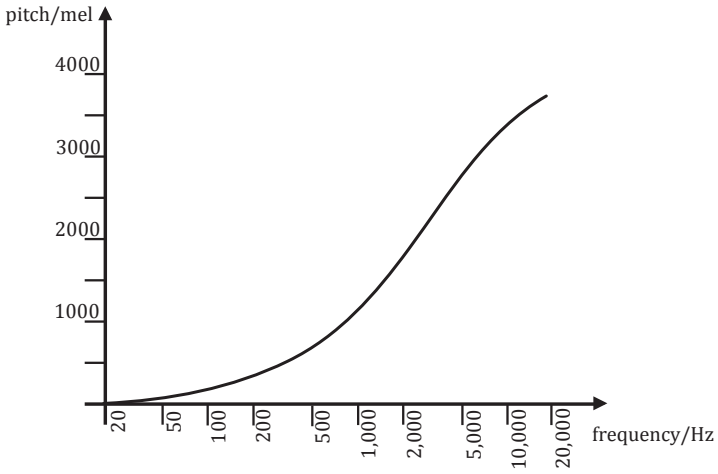


Figure 2.12: Pitch as a function of frequency [103]

tones within this 3 Hz range. This effect becomes worse for frequencies beyond 3 kHz as illustrated in figure 2.13: E.g. at a base frequency of 62.5 Hz subjects could identify comparison tones if their frequency differed by 0.5 Hz if the loudness was 25 sones and by 1 Hz if the loudness was 5 sones. At 4 kHz, however, the difference had to be at least 8 Hz if the loudness was 25 sones and 34 Hz if the loudness was 5 sones. The sones measure will be further explained in the following section 2.4.2.

Secondly another important realtion between pitch and sound intensity was observed by [77]. Again standard tones of fixed frequency and intensity were presented to subjects asked to manipulate the frequency of a fixed intensity comparison tone until it matches the standard tone. In subsequent tests the standard tone was changed from 500 Hz to 7,000 Hz. The interesting outcome is that comparison tones were perceived as either higher if the standard tone was beyond 2,500 Hz or as lower if the standard tone was below 2,500 Hz. This effect increases drastically with the intensity of the comparison tone. Figure 2.14 shows the changes in perceived pitch as a function of intensity for nine different frequencies. The abscissa is the change in frequency required for the comparison stimulus to match the pitch of the standard tone at the intensity given by the ordinate [113]. Theoretically this effect would result in serious problems for musical performances where the pitch of notes would change with intensity but fortunately this effect only appears with pure tones. It is an instrument's extensive harmonic structure, which finally prevents a note's dependency of the actual intensity [103].

2.4.2 Perception of loudness

Analogous to experiments and investigations in context with pitch perception, a number of subjects are presented a pure 1,000 Hz tone with a specific intensity and asked

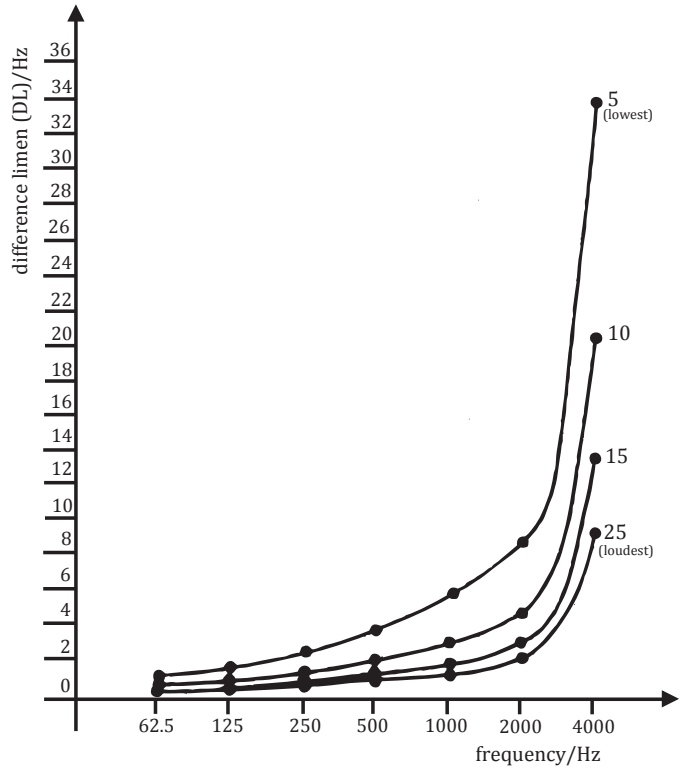


Figure 2.13: The difference limen (DL) for pitch as a function of frequency [103]

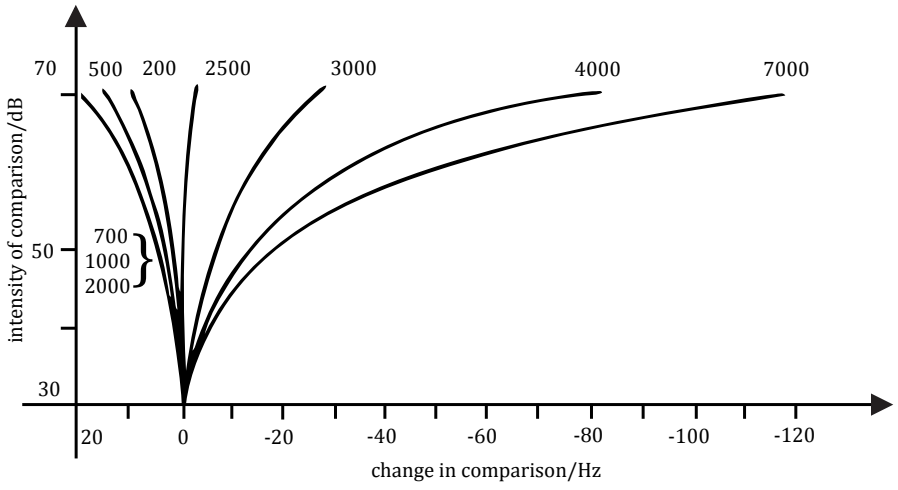


Figure 2.14: Change of pitch with intensity [77]

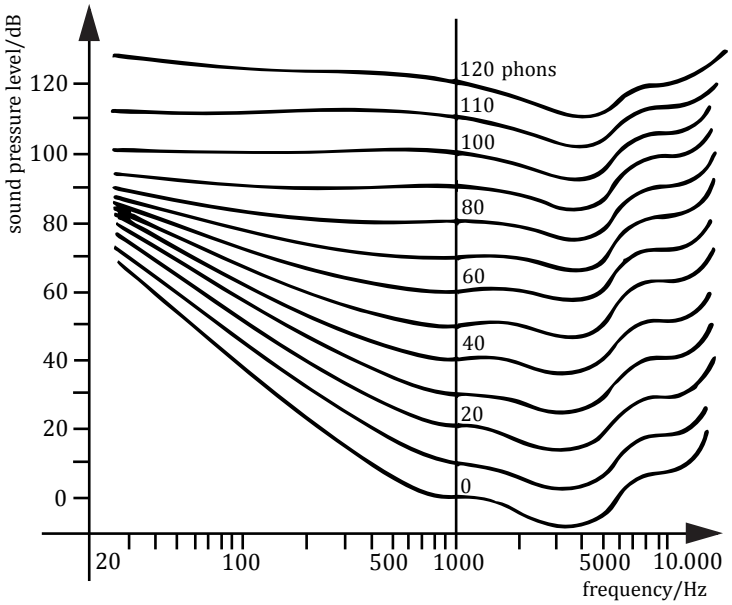


Figure 2.15: Equal loudness contours [103]

to manipulate the intensity of further higher and lower frequencies until their perceived loudness was equal. The results is a graph illustrated in figure 2.15 with the so-called "equal loudness contours" [103] which show at which frequency and intensity a pure tone is perceived with an equal loudness. This graph defines the "phon" scale as a unit of subjective loudness perception. The graph clearly shows that on one hand the human ear is most sensitive to frequencies in the mid range of 500 Hz to 7 kHz and requires significantly more intensity for frequencies below and above this range. On the other hand this discrepancy fades with the level of loudness so that for a perceived loudness of 120 phons the sound pressure level is almost equal all over the audible spectrum.

A more direct way to create a scale for subjective loudness is to use the direct magnitude production technique developed by Stanley Smith Stevens [144] defined the sone as the unit in such a loudness scale. The loudness of a 1,000 Hz tone at 40 dB intensity is arbitrarily assigned to 1 sone. To obtain a tone with the loudness of 2 sones, a subject is asked to manipulate the intensity of the stimulus so that the loudness is exactly twice that of the 1 sone sound. Similarly a 0.5 sone stimulus is one that is perceived to be exactly one-half the loudness of the original tone. Loudnesses that are not even multiples of the original 1 sone sound are obtained by bisection: A 3 sone tone is one that is perceived as midway between the loudness of a 2 sone and a 4 sone stimulus. [143] obtained loudness estimation data from a large sample of subjects and determined that loudness (in sones) was related to the sound intensity of a sound by a power law of the form:

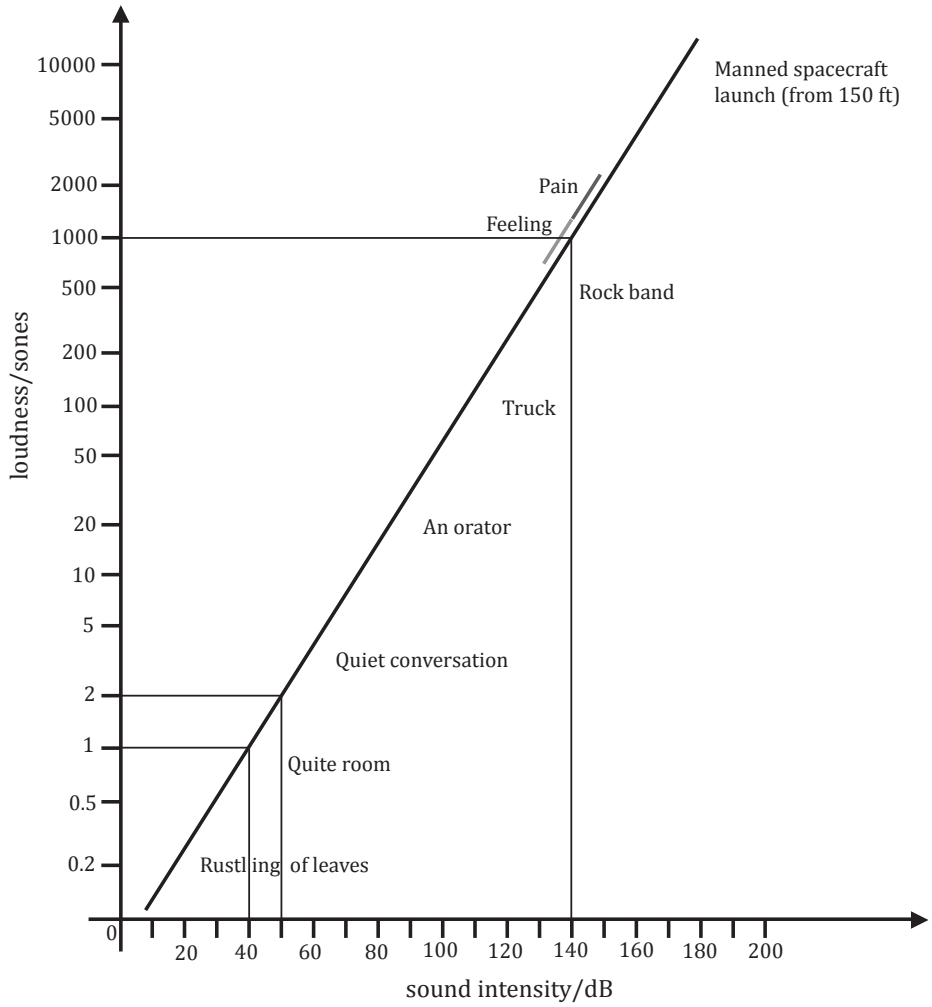


Figure 2.16: Graph relating loudness in sones to the intensity level of a stimulus [144]

$$\text{loudness} = k \cdot (\text{intensity})^{0.3} \tag{2.12}$$

The foregoing again applies exclusively to pure tones. [144] and others have proposed models that allow more complex sounds to be scaled according to their perceived loudness. These models all involve dividing the stimulus into a number of narrow frequency bands, and obtaining values for the SPLs within each band. The sone level of the total stimulus is then determined by summing the loundness of the individual bands. The complex tones whose scaled loudness are displayed in figure 2.16 were obtained in this way. According to the data presented in this figure, the loudness of a rock band is approximately three times greater than the noise of a nearby truck, which in turn is about 30 times lounder than the sound of a quiet conversation.

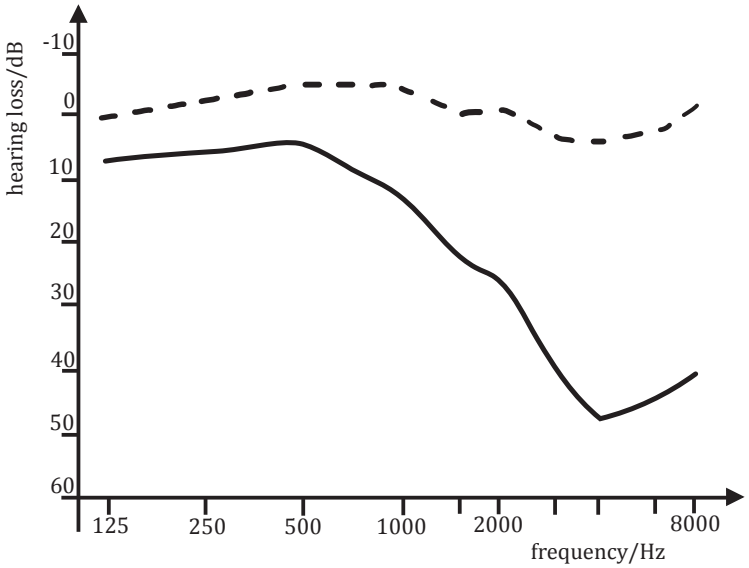


Figure 2.17: Hearing loss as a function of age [155]

It is commonly known that older people typically suffer from hearing loss. As outlined previously hearing loss can appear due to numerous reasons but in most cases a direct relation with abrasions of the middle and/or inner ear does exist [103]. The more intensity a sound has, the higher the risk for abrasions of the middle ear's malleus, incus and stapes or the inner ear's hair cells. In figure 2.17 hearing loss is plotted as a function of frequency for two groups of people. The dashed line shows the auditory capabilities of a group of young people in the age from 18 to 30 years. The solid black curve was derived from a group of people older than 65 years old. This group is at least 7 dB less sensitive than the younger group with a maximum sensitivity difference beyond 1,000 Hz. For frequencies $>$ 5,000 Hz the difference results in even up to 50 dB, which equals a factor of about 300 [155].

2.4.3 Sound localization

Besides pitch and loudness it is also important to know from which direction a sound is actually coming from. It is mainly the fact that we have two ears which equips us with the ability of sound localization. Figure 2.18 shows a human's head from the top with the corresponding azimuth of $+90°, -90°, +180°, -180°$. A sound source is placed at 35 degrees in front of the subject and the sound waves reach both ears respectively. As illustrated the waves travel on two different paths resulting in a different sound impression as if the path was equal. In that context the most significant differences occur in the timing and the amplitude between the two received signals. Besides the fact that sound waves traveling to the right ear pass a longer way than they do when traveling to the left ear, they also need to pass through parts of the head. The former

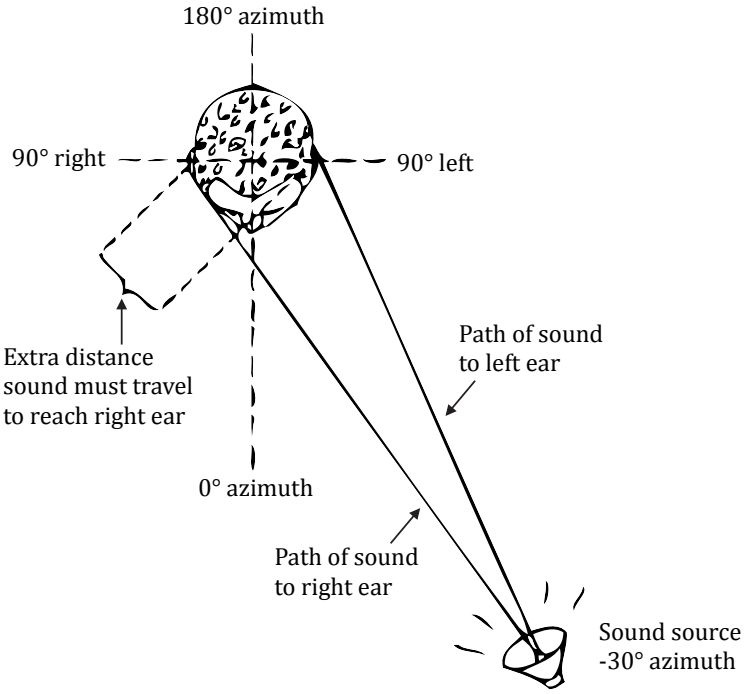


Figure 2.18: Intensity and timing differences due to longer signal paths [103]

circumstance leads to a slight time delay in comparison to the left ear, while the latter leads to a damping of the signal's amplitude – especially for higher frequencies. As the position of the sound source changes, these signal differences change accordingly, which finally gives us information about the actual location of the sound source. This effect can be reinforced by a signal's phase: Assuming a sound's wavelength is of a longer duration than the time gap between the two ears, our brain will notice the same signal on both ears, however, since the signals on both ears differs in phase, it is able to use this phase shift information in order to reach a conclusion about the sound's location [103].

Apart from the localization of external sound objects we also have an ability to localize our own position in a given space. Subconsciously every human being perceives delays as a natural part of a given environment. These delays are typically caused by the reflections of sound waves on objects close to us. If we consider ourselves clapping with our two hands in an imaginary room, each wall would reflect the sound back to our ears, however, since these delays range in relatively small dimensions, we recognize their existence not as a separate or doubled signal but as a so-called reverb instead [63]. This finally result is a signal which is mixed with the original or direct signal and the corresponding reflection. The closer the reflecting object is, the smaller the reflection time will be and this in turn gives us the ability to conclude, how large a room is. This phenomenon becomes more complex by the fact

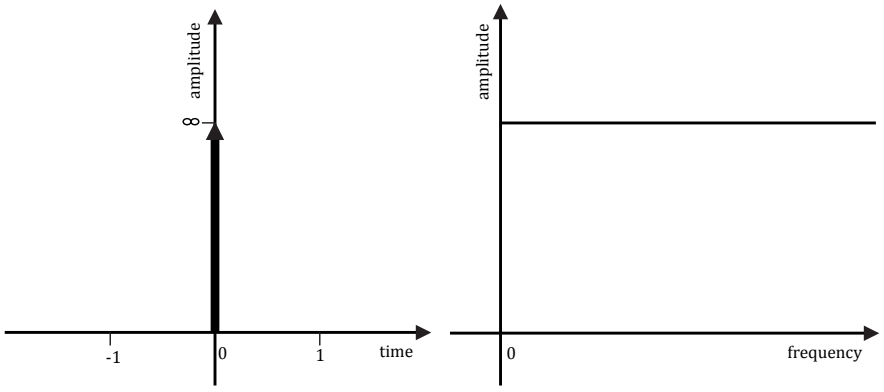


Figure 2.19: The Dirac pulse (left) and its frequency spectrum (right)

that various reflection times of various walls can be expected. Additionally the kind and intensity of a reflection depends on an object's material and typically leads to more than one single reflection. The final result is the direct signal of our clap mixed with the reflections. This varies from space to space and depends on the room's reflections, which are determined by the so-called "room impulse response". Before approaching the room impulse response mathematically, another important signal – the Dirac pulse – requires to be introduced: The Dirac pulse [151], introduced by the British theoretical physicist Paul Dirac, is a mathematical function, which represents an infinitely sharp peak: $\delta(x)$ has the value 0 everywhere except at x = 0 where its value is infinitely large resulting in a total integral of 1. Hence, its mathematical description requires the two equations 2.13 and 2.14. However, the author has to point out, that this is merely a heuristic definition as no real function can fulfill such properties. This discrepancy is further outlined in [39] and [84]. In the context of signal processing the Dirac pulse is often referred to as the "unit impulse function". The shape of the Dirac pulse is illustrated in the left part of figure 2.19 as a probability density function.

$$\delta(x) = \begin{cases} \infty & x = 0 \\ 0 & x \neq 0 \end{cases} \tag{2.13}$$

$$\delta(x) = \int_{-\infty}^{\infty} \delta(x)\,\mathrm{d}x = 1 \tag{2.14}$$

In practical terms, the Dirac pulse can be described as a very short clap with a very high amplitude and hence represents the ideal signal for the retrieval of the room impulse response [63]. This statement is enforced by observing the Dirac's spectrum in the frequency domain, where it contains any frequency, as illustrated in the right part of figure 2.19. If we consider a room as an LTI system with a Dirac pulse as the

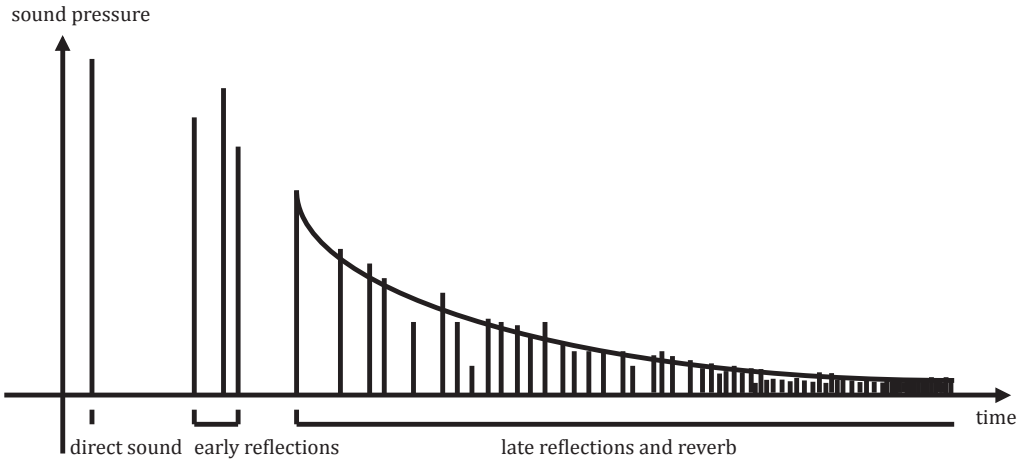


Figure 2.20: Example of a room impulse response [63]

input signal, the output signal of this system represents the room's impulse response. In other words: The LTI system "room" is completely characterized by its impulse response. Figure 2.20 shows the impulse response of an example room. As usual in audio engineering, the reflections are divided into early, late reflections and reverb [63]. In order to calculate how a given signal would sound in a given room with a certain impulse response both signal's need to undergo a so-called "convolution" [146], which flips one of the signals in time and finally multiplies their values. The following equation shows how a given input signal s(t) is convolved with an impulse response h(t) resulting in signal g(t):

$$g(t) = s(t) * h(t) = \int_{-\infty}^{\infty} s(\tau) \cdot h(t - \tau) d\tau \tag{2.15}$$

In the following the author will use an example room made of a floor space of a uniform rectangle: If the listener was standing right in front and in the middle of one of a room's walls, he would receive this wall's reflections of his clap with a minimal time delay in comparison to the maximal reflection delay of the opposite wall. Due to the centered position the lasting two left and right wall's reflection delays will be perceived as half of the maximal expected delay. The perceived reflection times change accordingly with the actual room position so that each location results in a different audible impression. In the middle of the room each wall produces an equal reflection delay. In combination with the aforementioned time or phase shifts between the two ears it is possible to draw a conclusion concerning one's actual position in the room and the current line of sight.

2.5 Music cognition

From the physical sound generation towards its psychoacoustical treatment, the next and final step is the observation, how our brain interprets the perceived sound signals as what we call music. In case the signal is produced by a musician, it is of interest, how musicians deal with psycho acoustic effects and under which conditions a musical interplay is possible. Research in this field leads us from from psychoacoustics to the area of music cognition [110]. Music cognition is an interdisciplinary approach of psychoacoustics, sensation and music theory, which focuses on the mental processes that support musical behaviors and make people understand music. Altogether this includes various aspects such as auditory scene analysis, perception of acoustic sequences or attention to auditory events but also auditory memory, recognition of sound sources and even auditory agnosia or the roots of music and language development [110]. Any of these fields has undergone intensive investigation but not all of them matter in context with this elaboration of distributed music. In the following the author will give a brief introduction into the fundamentals of music theory as a basis, in order to outline, which special areas of music cognition have significance.

In any part of the world, music has always been existing as a substantial part of human cultural life, and over thousands of years different cultures have developed different kinds of music based on their natural understanding of what music is or what it should be. Even without any knowledge of music theory these differences appear quite obviously when we listen to traditional german music in contrast to chinese and arabic music. Instead of emphasizing the differences and their reasons, the author will point out the main aspects which are equal in any music style, which can be broken down to three main parts – rhythm, harmony and melody [96].

Many aspects of our general daily life underly an individual subjective perception. Without a reference, adjectives like big, small, long or short can be considered relative since they depend on the observer's point of view. The same problem of how to apply a commonly accepted reference appears in context with these three aspects of music.

2.5.1 Rhythm

Whether a music piece is characterized as slow, medium or up tempo, or whether the duration of a single note is short, medium or long, mainly depends on the individual, unless special references or conventions are taken into account. Fortunately such conventions exist in form of "bpm" (beats per minute) for the actual speed of a song and the note length as a measure for the duration of a single note. In general a bpm below 80 describes a relatively slow beat, beyond 80 up to 130 a medium speed beat, while a tempo above a bpm of 130 can be considered as relatively fast [104]. In practice this breakdown is more fine grained. 60 beats per minute are an appropriate descriptive example, since this speed equals 1 beat per second. Back to the question,

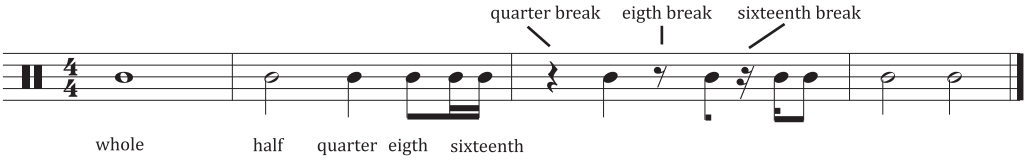


Figure 2.21: Example of a rhythmical pattern with whole to sixteenth notes

how a reference to a single note duration can be found, it is now assumed that each beat equals a quarter note so that in the current case this quarter note would have a duration of precisely 1 second. Respectively the duration of a half note equals 2 seconds and a full note takes 4 seconds. Theoretically shorter note durations down to $\frac{1}{128}$ or less exist but practically durations below $\frac{1}{32}$ are rather uncommon. However, since the note duration depends on the actual bpm, a $\frac{1}{64}$ note is more likely to appear in a piece of 60 bpm (where $\frac{1}{64}$ = 62,5 ms) rather than of 120 bpm (where $\frac{1}{64}$ = 31,25 ms).

Furthermore, an important characteristics of rhythm is an arrangement of the beats over time [139]. These arrangements lead to pulse groupings, called measures or bars for which the number of beats is specified and how the value of a written note is counted and felt as a beat. The most common measure mainly related to pop- and rock music is in $\frac{4}{4}$ so that each bar is made of 4 quarter notes but also the the $\frac{3}{4}$ measure is not uncommon and well known e.g. in the waltz. Besides this jazz music and various styles in new or avantgarde music often use rhythms of $\frac{5}{4}$, $\frac{7}{8}$ or even changing measures. Theoretically and mathematically such rhythms exists and build an interesting alternative to the common standard rhythms but an untrained ear often perceives these rhythms as complicated, strange or not understandable. In order to the point out the clear character of a current rhythm, it is mainly the first beat of a measure, which is accented through increased attack and possibly subtle variations in duration. There are conventions in most musical traditions for a regular and hierarchical accentuation of certain beats to reinforce the meter. Special forms are syncopated rhythms that accent unexpected parts of a beat or even polymeter, in which simultaneous rhythms of different meters are played at the same time [104]. Figure 2.5.1 shows a rhythmical example sequence of 4 measures.

2.5.2 Melody

Under rhythmical aspects a note is assigned a note duration of $\frac{1}{x}$ but in order to determine and identify the actual pitch of a note, the so-called tuning process has to be applied. In this process note names in the form of alphabetical letters from “A” to “G” are assigned to a certain note’s frequency [82]. It is typically the frequency of 440 Hz which is declared as the standard pitch and the note name “A”. When comparing two notes, it is the difference in frequency between the two notes, which is called an interval. The most basic interval is the octave. A note and another note

semitones	common interval name	interval width in cents
0	perfect unison	0
1	minor second	100
2	major second	200
3	minor third	300
4	major third	400
5	perfect fourth	500
6	augmented fourth	600
7	perfect fifth	700
8	minor sixth	800
9	major sixth	900
10	minor seventh	1000
11	major seventh	1100
12	perfect octave	1200

Table 2.1: Interval width in equal 12 step temperament

with twice its frequency form an octave. For example, if the pitch of 440 Hz is A, then the pitches with frequency 880 Hz, 1760 Hz as well as 220 Hz, 110 Hz, and 55 Hz are also A notes. Such notes are typically written into a score, in which a higher positions means a higher pitch, while a lower position means a lower pitch. In this regard we add a line identificator to the various A notes in order to be able to distinguish between them. The standard pitch A is assigned the one-line A, while the higher octave is assigned the two-line A and so forth. With respect to lower pitches, the 220 Hz has no additional identifyer, while the 110 Hz A is called the "great", the 55 Hz the "contra" and the 27,5 Hz the "sub-contra" A. In order to obtain further intervals, one octave is generally subdivided in a 12 step resolution, while each step represents a so-called "semitone". The respective intervals can be measured in the logarithmic unit cents. 1,200 cents are equal to one octave and an equally tempered semitone (the interval between two adjacent piano keys) is equal to 100 cents. Hence, the ratio of frequencies between two adjacent semitones is the twelfth root of two:

$$r = 100[\text{cents}] = \sqrt[12]{2} \approx 1.0594630943593 \tag{2.16}$$

Based on this 12 step resolution it is possible to retrieve the frequencies for the respective intervals as shown in table 2.1.

Using specific notes of this choice, assigning them with a temporal value and finally making this series of notes sound in succession, is the process of composing a melody. However, this process implies another important aspect of music theory – the principle of scales [42]. In contradiction to seeing an octave as the sum of 12 semitone steps, a scale uses patterns of half and whole steps tones (2 half tone steps equal one whole

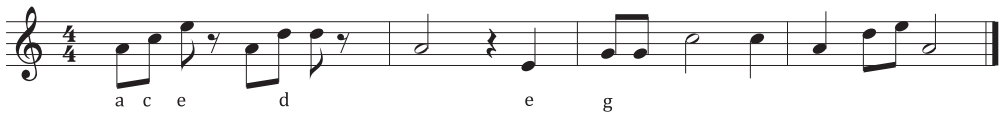


Figure 2.22: Example of a melody sequence

tone step) within that octave. Due to historical and cultural differences, cultures have established different commonly used scales, which is why e. g. chinese music sounds different than european occidental music. As an example figure 2.22 shows a melody in the A minor scale.

Furthermore, a composer can change the first note of a melody to be played as long as the subsequent half/full tone step relations of the following notes remain fulfilled. Transposing a piece from C major to D major will make all the notes sound two semitones higher. This so-called "key change" often affects the music's timbre and can also have technical implications for the performers. However, changing the key in which a piece is performed may go unrecognized by the listener, since changing the key does not change the relationship of the individual pitches to each other. Therefore, different keys are often considered equivalent and a matter of choice on the part of performers, which is especially true for non-classical music such as pop or folk.

So far we assumed an octave being divided into 12 absolutely equal steps, which refers to the equal twelve-tone temperament. Though its principle of identical half tone steps descriptively shows how notes can be arranged over frequency, it has a practical drawback. Whenever a musician wants to change to another key than the instrument was originally tuned in, the instrument will sound out of tune for specific intervals. The precise reason for that can be found in [82] and finally lead to the well tempered tuning system, which was introduced around 1700. The term "well temperament" usually means an irregular temperament in which the tempered fifths are of different sizes but no key has very impure intervals. This slight detuning of specific intervals all over the instrument's range finally leads to an acceptable tuning for any key. However, since each key has a slightly different intonation, different keys have distinct characters, which were often consciously used by classical composers. Also in that context the unit cent has become the standard method of comparing musical pitches and intervals with relative accuracy. E.g. if two frequencies a and b are known, the number of cents measuring the interval between them may be calculated by the following formula:

$$n = 1200 \log_2 \left(\frac{a}{b} \right) \tag{2.17}$$

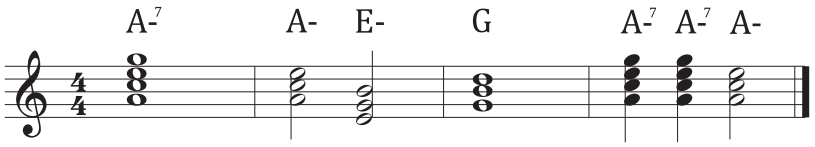


Figure 2.23: Example of a harmony sequence

2.5.3 Harmony

Harmony and melody mainly interdepend, and a melody can as well outline a harmonic structure of a music piece [96]. However, the significant difference is that harmony describes the vertical sonorities in music, rather than horizontal sonorities of a melody. Vertical sonority refers to considering the relationships between pitches that occur together – an interval sounding at the same time. A larger structure involving multiple pitches is called a chord. Following the scale theory chords of any composition can be created but in common practice and popular music, harmonies are generally tertian. Jazz music typically adds the seventh of a scale to a chord, giving it the a "jazzy" character [42]. However, the study of harmony represents a rather complex area and especially in the 20th century many alternative types of harmonic structure were explored. Further information toward harmonic structures can be found in [91]. In terms of analyzing and visualizing a song's harmonic structure, figure 2.23 illustrates the most commonly used roman numeral system and the system of chord symbols for jazz and popular music. In aural training musicians try to consciously hear certain intervals and possibly further harmonic and/or melodic structures and write them down appropriately without further information. In that context it has been observed that some people are born with an absolute pitch, which equips them with the ability to name a sound's pitch directly as opposed to the majority of people who are only able to name the interval between a note's pitch and a reference note [82].

2.5.4 Individual delay perception and reaction times

Besides perceiving the natural reverbation of a room, another subconscious aspect of delay appears which is related to the physics of sound generation and the reaction time of musical instruments. E.g. in a piano performance the time elapsed between pressing a key and the corresponding note onset is about 100 ms for quiet notes and around 30 ms for staccato, forte notes [28]. Even if the pianist expects the note onset to happen somewhere in the middle of the course of the key, it is very likely that latencies will be different for different dynamic levels. However, pianists do not seem to have problems playing with such different latencies. It is likely that as part of the learning process on the instrument a musician's brain learns to deal with different latencies and in turn musicians subconsciously take such delays into account

and adapt their playing respectively. This especially matters in pieces, which have continuous dynamic changes, in which the performer has to adjust him or herself to the corresponding changes in latency. This phenomenon appears to be more or less dependent on the actual musical instrument. The latency between the performer's action to play a note and its actual onset is present, is generally known as delayed feedback [33].

However, delayed feedback does not only occur artificially by external devices and instruments. Due to the fact that a human's motor system cannot react immediately, it implies a natural delayed feedback and in turn the brain has to issue motor commands ahead of time in order to perform in time on a specific beat. The authors of [109] even conclude that various feedbacks for our actions are used to calibrate how much ahead of time commands are issued. This relationship was further examined by [27] so-called tapping-experiments, in which a number of subjects were asked to perform synchronous tapping along with a metronome reference, using a one degree-of-freedom haptic device. The final outcome showed that subjects tapped along with a metronomic stimulus virtually always about 10-80 ms ahead of time without noticing it – the median was at 30 ms. In a musical context this effect can also be observed when comparing voices of a piano piece. The so-called "melody lead" happens, when notes of one melody voice are played typically around 30 ms before supposedly simultaneous notes of the second voice. Despite the percussive characteristic of the piano sound and corresponding relatively short attack times, these asynchronies are not consciously perceived by neither the performers nor the audience. These facts make the author believe that latencies of up to at least 30 ms do not impair music performances with conventional instruments and hence can be considered normal and acceptable. It is likely that on a subconscious level such asynchronies are used by the ear as strong cues for the identification of simultaneous tones [124]. Even if such asynchronies are not consciously perceptible, they play a musical role and can be partly under the control of the performer [98].

Apart from the phenomenon of delayed feedback in tapping experiments a number of researchers also seeked after the question, how precisely human beings perform with beats: [130] showed that we can tap a steady beat with typical variations in intertap intervals as low as 4 ms and can consciously detect timing variations of around 6 ms [71]. If such variations appear in a cyclic and slightly higher dimensions up to 10 ms, we include them spontaneously into our performance [148]. Again these adjustments happen on a subconscious level. As the author of [133] suggests that this rhythmic perception is based on the comparison between the expected and actual time for each sound attack it is likely that such tiny variations are perceived not as timing variations, but as some kind of fuzzy musical characteristic like the so-called "feel". In jazz music e.g. it is common to play "laid back" or "forward", which describes a note onset slightly behind or before the beat and generally the better a musician is trained, the better he or she can consciously play with a given beat and consciously influence the "feel" of a music piece [98].

We summarize that depending on the musical skill and the played instrument musicians subconsciously trigger notes ahead of a the current beat. This beat develops in a musician's inner feel and builds the rhythmical reference. For the practical process of music creation, the natural delayed feedback is a natural precondition for playing notes on the beat of a current song and the higher a musician's skill, the better he or she can even consciously play before or after the beat as a stylistic device. In the following subsection the author will observe, what happens, if the beat is the reference not for a single musician but for an ensemble of musicians separated by space.

2.5.5 Delay perception in musical interplay

Live music requires musicians to interact with each other, which means that both musicians have to hear each other's instruments and when following the musical rules of rhythm, harmony and melody, they can finally build up musical structures. The process of interaction happens continuously and both musicians rely on each other's playing. When talking about music and its process of creation, it is generally uncommon to mention, that a musician's sound does not immediately reach the other musician's ear. As outlined earlier, a sound wave firstly travels through the air and remains an acoustic sound wave until it reaches the ear drum membrane first and after various signal conversions finally reaches the brain as a set of electric impulses. This propagation through the air needs time and the further both musicians are separated from each other, the more time the sound propagation takes. In air the propagation speed depends on the air temperature. The colder the air, the slower the air molecules transport the pressure wave. In general the speed of sound is stated with $c = 343\ \frac{m}{s}$ as a reference value with 20 degree celsius room temperature [67]. The precise value can be calculated with equation 2.18, in which ϑ represents the temperature in °C.

$$c = (331,5 + 0,6\vartheta)\frac{[\mathrm{m}]}{[\mathrm{s}]} \tag{2.18}$$

Musician's typically remain a couple of meters apart from each other and with the speed of sound, the sound propagation takes approximately 2.92 ms for every meter the sound wave passes. This propagation time prevents the instant signal delivery and results in a signal delay between the sender and the receiver of the sound wave. In a conventional rehearsing space scenario delays of possibly more than 20 ms are not uncommon. On the other hand on large stages, it is common practice to use stage monitors in order to compensate disturbing delay effects, musicians typically suffer from [63]. This in turn means that signal delay obviously matters in terms of musical interaction. In how far this is the case and how our brain perceives and deals with signal delays, will be outlined in the following.

From a pure physical point of view the problem of delay between two musicians A and B theoretically prevents musical interaction under convenient conditions for

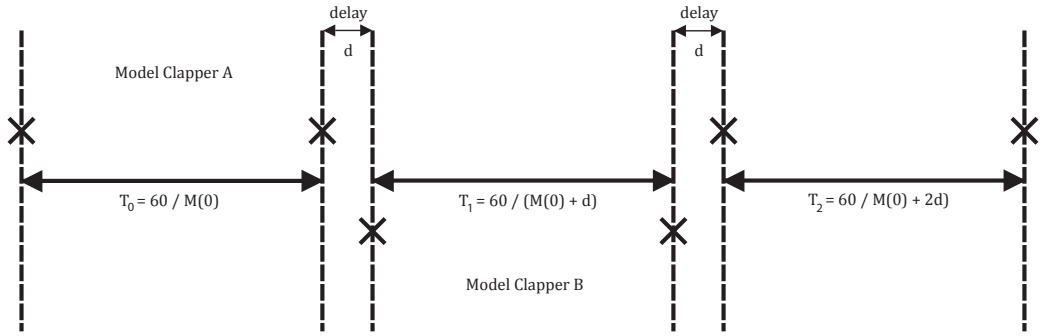


Figure 2.24: Theoretical "robot clapper model" [55]

both musicians. Due to the signal transmission time the signal per se cannot be played together and would lead either to a situation in which musician A plays to the signal of B or B plays to the signal of A but never in real unison and rather in a manner that one person leads while the other person follows. A simple model of mechanically performing robot-like musicians based on this theory is illustrated in figure 2.24 as a system that would perfectly detect any slightest tempo change, and finally incorporates any perceived delay as a decrease in tempo [55]. The tempo M would therefore decrease in bpm according to equation 2.19.

$$M(n) = 60/(T_0 + nd) \tag{2.19}$$

$T_0 = 60/M_0$ is the starting period in seconds, n is the quarter-note number and d is the delay time. Only a delay of d = 0 would produce a perfectly steady tempo, and any delay > 0 would cause deceleration [55]. The predicted deceleration times are depicted in figure 2.25, which assumes a starting tempo of 90 bpm. 90 bpm corresponds to a beat duration of $T_0 = 0.\overline{6}$ s. With 0 ms delay the top curve indicates a stable tempo of M(n) = 60 / $0.\overline{6}$ s + 0 ms) = 90 bpm all over the time. In contradiction to the lowest curve with a delay of 77 ms and after 35 beats the tempo has slowed down to M(35) = 60 / $0.\overline{6}$ s + 35 · 77 ms) = 17.85 bpm.

Though this purely physical explanation seems to be valid, there is a discrepancy with musical practice, where a musical performance is always declared as something, which really happens together in an interactive manner. This discrepancy can be explained by the fact that despite a certain amount of delay between two musicians both players perceive each other's signals as simultaneously as though there was no delay. First research in the area of delay perception was carried out by Ira J. Hirsh in 1959. His work was related to measurements of thresholds for identification of order for pairs of items (consisting of sounds such as hisses and clicks) in order to determine the requirements for temporal resolution in speech perception [110]. To illustrate these requirements, he cited the need for listeners to identify the order of "s" and "t" for the

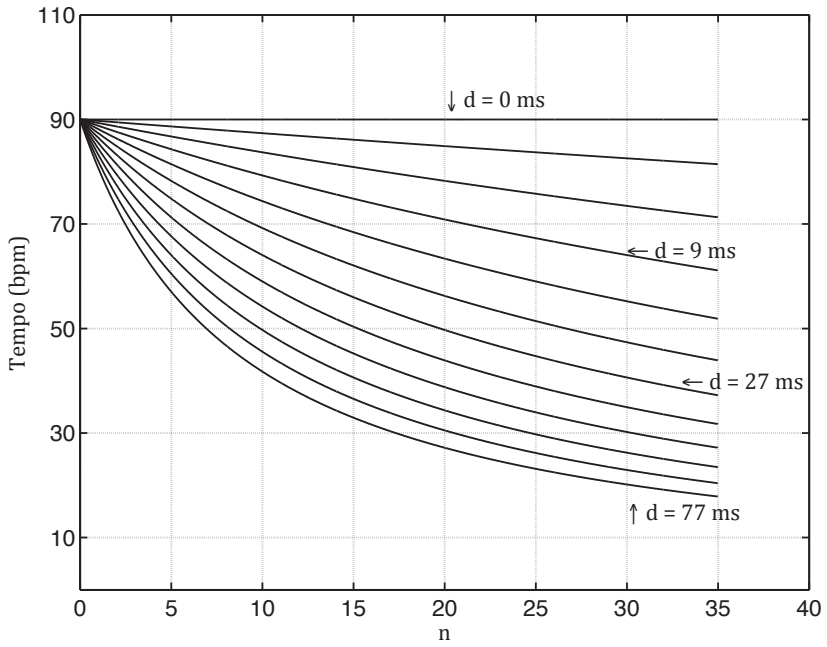


Figure 2.25: Predicted tempo deceleration for "robot clapper model" [55]

discrimination between the words "mist" and "mitts". His procedure involved a pair of qualitatively different sounds with asynchronous onsets, and the listener was required to identify which of the two possible orders was heard. The threshold was considered to be the separation required for 75 per cent correct responses. Hirsh reported that the threshold for identification of order – regardless of the items employed – was about 20 ms. Hirsh and Sherrick [110] expanded the study to pairs of sounds delivered to opposite ears, and again the threshold was found to be about 20 ms. They also used pairs of stimuli in other modalities (vision and touch), and in addition had one stimulus delivered via one modality and the other via a second. In all cases, roughly 20 ms was found to be the threshold for identification order, and it was concluded that this value represents a fundamental value for the identification of order [110]. Generally the effect of interpreting two successive events as simultaneous or not and reacting appropriately is part of the cognitive research area of "integration processes" [83].

The threshold for order identification represents an important fundamental cognitive value but it does not necessarily lead to a conclusion in terms of real musical interaction, which has to take rhythmical aspects of a real musical interplay into account. Research by Chris Chafe in 2004 [55] actually bases its findings on such aspects and can be considered as the first and most fundamental work in this area. The goal of their experiment was to identify the region of delay time that is most conducive

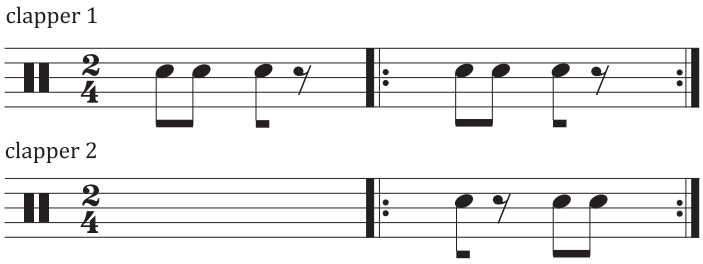


Figure 2.26: Duo clapping rhythm used in experiment

to maintaining a steady tempo. Numerous pairs of musicians were placed apart in isolated rooms and given a simple rhythm to clap together. A microphone was placed as close as possible to each performer's hands. Each monitored the other's sound via headphones, and an additional delay was introduced between the source and listener. Pairs of subjects were randomly chosen from students and staff at Stanford University regardless their musical qualification. At the beginning of the experiment, subjects were given the rhythm in music notation as illustrated in figure 2.26 and then listened to an assistant performing it. The task was to "keep the rhythm going evenly" during each trial without information about strategies or hints about how to do that. They were allowed to practice the rhythmical patterns face-to-face outside the separated setup until they felt comfortable. For each trial, one randomly chosen subject was presented with a starting tempo (a calibrated series of recorded claps) and the other heard nothing until the initiator began to clap. For each trial the delay between the two subjects randomly differed between 0 ms and 77 ms. After 36 seconds the monitoring was switched off in order to end the trial. Afterwards the recordings of the trials were analyzed with a precise event detection algorithm, which located clap onsets, from which the tempo was inferred [55].

After performing 15 session of altogether 173 trials, for each of them a graph in figure 2.27 was generated, which showed how performers clapped under different latency conditions. It shows that from 0 to 5 ms delay, the entire 95 % confidence intervals are positive, and this surprisingly indicates significant acceleration. Starting with a delay of 20 ms, the entire confidence intervals were negative, indicating the significant deceleration. Finally, the mean of all tempo slopes m_{acc} was calculated respectively, which also indicates a strongly linear relationship between m_{acc} and delay. Based on the latter, the authors concluded to a model expressed by equation 2.20, where m_{acc} represents the mean tempo deceleration. The variable x represents the actual delay and ϵ is a measure for potential timing errors. The negative slope of this linear model in this equation confirms that the mean tempo tends to decelerate with increasing delay, however, for delays < 11.5 ms the mean tempo tends to accelerate with decreasing delay. The theoretical best delay was found at 11.5 ms, where the average clapping result did show neither an acceleration nor a deceleration [55].

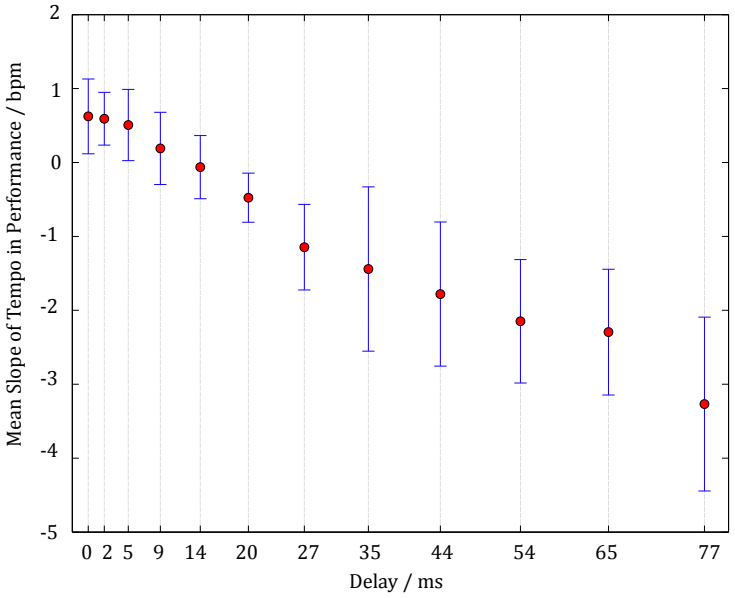


Figure 2.27: Slope of performed tempo vs. delay time [55]

$$m_{acc}(x) = 0.58 - 0.05x + \epsilon \tag{2.20}$$

The outcome of this experiment confirmed the anticipation of cognitive researchers, that in terms of rhythmical interaction our brain is able to function even when the remote player is separated by a delay. In fact the first significant deceleration appears at a delay of 20 ms, which equals the threshold for order identification but this does not lead to a musically unacceptable result. For delays up to 30 ms the degree of deceleration was within the limits of what could be considered musically acceptable. The linear relationship of m_{acc} to delay time implies that there is some specific "best" delay time, rather than a wide range of conducive conditions as expected.

2.6 Results

The human ear is a highly sensitive organ for the reception of sound signals. The actual process of hearing is of a complex kind, which includes numerous stages, through which the incoming sound waves are processed in. The total processing follows the general rules of signal processing and hence the terms signals and systems can be used respectively. Altogether the ear acts as an LTI system, which processes the environmentally audible information and transforms it into an electrical signal which in turn is fed to our brain.

How we actually perceive sound waves, underlies the area of psychoacoustics with its various phenomenons. Due to these psychoacoustic effects the total hearing process acts as an LTI system with different frequency depending gain factors and we can conclude that the perceived pitch has not a linear relation to the frequency of a sound and the perceived loudness has not a linear relation to the amplitude. Fortunately psychoacoustic effects can be considered as equal for any human being or at least most of them and hence we can assume that every human being perceives sounds equally. People consciously live with psychoacoustic effects, consider them as a normal part of their hearing ability and in turn any acoustical interaction such as speech conversation or musical interplay automatically involves them. However, there is still one major discrepancy in terms of audio perception: Audible information ranges in a frequency area of 20 Hz to a theoretical maximum of 20 kHz but he upper limit of 20 kHz is only valid for young children. Due to abraisons of the middle and the inner ear, this upper limit decreases tremendously and possibly below 10 kHz depending on the age and sound levels the ear undergoes over the years. Hence, due to the mentioned effects of hearing loss not all human being perceive these signals in the same way – some with a larger frequency spectrum and in turn in a better quality, others with a narrower frequency spectrum and in turn in lower quality.

The transition from the perception of sounds towards the perception and understanding of music finally leads us to the research area of music cognition with its interdisciplinary aspects of perception, music theory and psychology. Cognitive processes finally make us understand to distinguish between sounds and music and the more we train our cognitive skill, the better we understand musical principles. Musicians as the creators of musical content constantly need to trains their cognitive abilities in order to fine tune their hearing, musical analysis and instrumental reaction. In contradiction to psychoacoustic effects, the cognitive abilities are not equal to any human being. Although can be improved through training, there must be an underlying natural talent. Some people are born with an absolute pitch, which makes them recognize certain frequencies constructs and name them accordingly. In contradiction the majority of people hears with a relative pitch, which requires a reference tone.

Due to the variety of factors the overall hearing process involves, it is clear that how people hear and respond to music, is not equal. The perceived result is a mixture of the physical ear properties, certain psychoacoustic patterns and a significant amount of cognitive awareness in combination with training skill. Apart from that it is undeniable that the process of music creation is of a highly sensitive kind with strong personal, environmental and other individual and sometimes not explainable preferences. This includes sympathy/desympathy with others, suitable rehearsing spaces and even the look or smell of others. In addition to that of course the actual instrument skill determines musical quality and it typically takes years of practice and training for musicians to reach a certain skill.

However, despite the mentioned aspects of hearing, music cognition and creation the key problem for a distant musical interaction is the ability of keeping a common

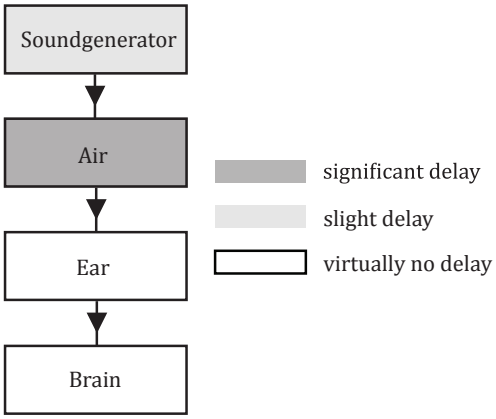


Figure 2.28: The natural signal path

rhythm. This ability suffers from latencies the medium air introduces due to its propagation speed of 343 m/s. The outcome of an experimental setup confirmed that musicians are able to cope with latencies up to approximately 30 ms, which corresponds to a maximal distance of 10 m. Beyond this value musicians speed down with the performed tempo so that a musical interaction becomes impossible. Figure 2.28 illustrates the natural signal path between two performing musicians, in which delay critical stages are marked in light or dark grey depending on their actual relevance. As some musical instruments have delayed response times already the sound generator can represent a source of delay.

Chapter 3

New cognitive findings in delayed musical performance

As described in the previous chapter, research carried out by Chafe in 2004 [55] can be considered as the first and fundamental work in the area of delay influenced musical interaction. However, the author has to remark, that in order to prove the relationship between latency and a musically playable situation, test cases would require the involvement of further musical aspects and confirmation of real musicians in real musical performances – rather than a random choice of people in an artificial clapping scenario. Two experiments is this direction were performed in 2004 by Elaine Chew et al. in [57] and the author himself in [44]. Both tried to identify a latency threshold up to which a real musical interplay is possible and in both cases two musicians performed in two places, separated by a delay processor, which was adjusted from 0 ms up to a maximal delay threshold. However, the target of both experiments was different: While [57] used a classical music piece for four handed piano, [44] used a scenario, in which an electric guitar and an electric bass player performed swing and funk music patterns. The results of both experiments can be seen as an interesting complement to the fundamental work of Chris Chafe in [55]: [57] stated, that up to 50 ms can be considered as an acceptable delay value for classical four handed piano music. With increasing delays beyond that value, players struggled to keep the time. Up to 75 ms musical interaction was still possible but difficult, while 100 ms was considered as extremely difficult and 150 ms as almost impossible to play with. In contradiction, the author's experiments in [44] showed that for rhythmical music the players slowed down at delays of 30 ms in such a way that musical interaction became impossible – similarly to the clapping tests in [55]. However, with fast and time critical music such as funk music, which requires note onsets to be precisely on the beat, this threshold required adjustment down to 25 ms.

Further studies in collaboration with Alvaro Barbosa [33] lead to the conclusion that there seems to be a relation between the actual speed (bpm) of a performed song and the perceived latency tolerance [33]. The findings were based on the performed results of three different music styles with varying speeds. The overall outcome again was a general delay acceptance threshold up to 35 ms, however it significantly decreased for speeds beyond 100 bpm. Despite the fact that [33] already takes the issue

of speed into account, it leaves open questions concerning musical and cognitive aspects of delay-influenced musical interaction, which will be emphasized in the first section of this chapter. In the second section the author will additionally investigate situations, at which the latency resides beyond the required interaction threshold in order to conclude, if modified or compromised forms of delayed musical interaction have significance to professional musicians or not.

3.1 Comprehensive analysis of delay-influenced musical interaction

With respect to a comprehensive evaluation of delayed musical performance, it is clear, that the actual test setup has to be as realistic as possible. Besides the involvement of real music at different speeds musicians should play in an acoustic environment, which suits the original sound conditions. The authors of [55], [33] and [44] use the same sound studio setup, in which individuals were connected via head- and microphones. As a result, the individuals‘ ears receive the direct sound of the remote side. Apart from the artificial delay line, this setup is also used in a conventional sound recording scenario of professional sound studios. Though this principle offers pristine sound transmission on one hand, it can lack aspects of natural sound conditions on the other hand. In comparison to the natural room reverb in rehearsing spaces, the direct and dry signal can be perceived as “cold“, “hard“ and in turn as unnatural, which is why singers in particular often ask for an artificial reverb in order to compensate this effect. The question if this perceptual phenomenon has an impact on the actual delay tolerance, requires consideration.

Another problem in context with a realistic music scenario is the choice of involved musicians and their musical skills. In general it can be said that the higher a musician’s skill, the more stable musicians can play and in turn the better a song can be performed. So far investigations in perceptual latency measurements have not taken this into account. The authors of [55] choose a number of individuals regardless of their skill and concludes the outcome of clapping experiments by statistical means. The authors of [33] performs the tests with a set of 4 talented amateur musicians, whose trials were recorded. Later the evaluation of the recordings were realized by an online survey open to the public. Either of the two evaluation schemes is absolutely valid and common practice in cognitive sciences: In 1959 the researcher Ira J. Hirsh (c.f. section 2.5.5) discovered the “threshold for the identification of order” by statistical means regardless of the musical skill [110]. This threshold describes the delay time below which individuals are not able to distinguish the order of two played sounds. However, especially in context with musical or at least rhythmical content, an individual’s music skill might have an impact on the test results: The authors of [41] repeated Hirsh’s experiments under modified conditions and found that an untrained ear performed much worse than a trained one: In contradiction to Hirsh

stating a general value of about 20 ms for the identification of order [110], they firstly distinguish between listeners, who practiced under the given conditions and became familiar with the situation and task, and those, who had to perform the test without any training. For the former group they state a value of 30 ms, while the untrained ear determined thresholds of up to even 150 ms for identification of order.

Altogether it is obvious that existing experiments towards delayed musical interaction do not exhibit a common basis or equal test conditions: Although [57] and [33] work with real musicians, other important aspects such as rhythm/solo constellation, note resolution, music style, a musician's skill and rhythmical attitude are not taken into account. Hence, the outcome of either elaboration varies significantly. In comparison the results of [33] and [55] seem to correlate since both declared a delay threshold of 35 ms as musically acceptable. However, neither clearly states on what aspects this judgement is based on and in how far professional musicians would agree with this decision. Nevertheless their choice of beat oriented patterns seems to be better than a classical piano piece: The former obviously appears to be much more time sensitive but as yet it is unclear why.

In the following, the author will – based on the huge variety of open questions – present an evaluation concept and its realization with professional musicians, the goal of which is to overcome the general confusion about valid latency thresholds in delay influenced musical interaction.

3.1.1 Concept

Unlike classical conducted music in beat oriented music we generally have to distinguish between a solo instrument and a rhythm instrument. In turn, constellations of rhythmical instruments are called "rhythm section" and constellations of solo instruments are called "solo section". As an example with drums, bass and saxophone, a simple case is present in which drums and bass form the "rhythm section" while the saxophone player represents the "solo section". Though, of course, the solo section and any musician must have a sense of rhythm, it is basically the interplay of bass and drums, which forms the essential fundamental groove of an ensemble which allows other solo instruments to play upon. In this scenario the saxophone player relies on and plays on the groove that is produced by the rhythm section [49]. In context with this section it is the rhythm section, which has significant and major importance. The most common and simplest case of a rhythm section is the previously introduced drums and bass constellation. Hence, for the upcoming trials of this experiment the author will focus on the delay influenced interplay of rhythm sections made of drums and bass. Each person plays in a separated room of a professional sound studio [18], connected via microphones or direct cable links and headphones. The audio signals have CD quality. Regarding the choice of musician constellations the author considers it as problematic, to replace both musicians in subsequent trials: Each musician's personal playing style could have an impact on the overall perfor-

mance and since it is not clear, which musician's impact is stronger, a comparison of different constellations is easier, where only one of the musicians is being replaced. Hence, the author asked 5 professional drummers to perform in the proposed trials. For each trial the author proposed a single professional bass player to be the rhythmical counterpart. This way a direct comparison of each rhythm section constellation is possible. The choice of the drummers assumed professionalism as a collective term for playing regular commercial concerts, playing tracks for professional sound studio productions, teaching students and an overall acceptance in the north German music scene. Although the author did not consider it as a mandatory requirement, each drummer held a university degree in music.

In terms of the trial procedure the author chooses a test scenario, which consists of subsequent $\frac{1}{4}$, $\frac{1}{8}$ and $\frac{1}{16}$ bass sequences followed by a free improvised bass pattern based on the actual rhythmical resolution. This basic test pattern is made of $3 \cdot 12 = 36$ bars as illustrated in figure 3.1. Furthermore, the author created a reference one bar drum pattern for each of the 36 bars shown in figure 3.2, however, drummers have the choice to play in such a way they consider as most suitable and convenient. The drum kit itself consists of bass drum, snare drum and hi hat.

Before the trials start both players practice the given sample piece under conventional conditions in the same room until they feel comfortable with it. As the trial starts the drummer first listens to a metronome for 4 bars before he plays 2 bars by himself. After these two bars the bass player starts joining in with the test pattern. The experiments are performed at speeds of 60, 100, 120 and 160 bpm and the delay between the two players is increased from 0 ms in steps of 5 ms until one of the player feels uncomfortable with the playing situation or players tend to slow down. In that context the players have to evaluate the actual delay situation as "excellent", "tolerable" and "not tolerable". In the latter case the author additionally applies an artificial reverb to the received signals and force the drummers to modify their played rhythms regarding a shuffled or straight phrasing. If this still leads to an unacceptable situation the end of the trial is reached.

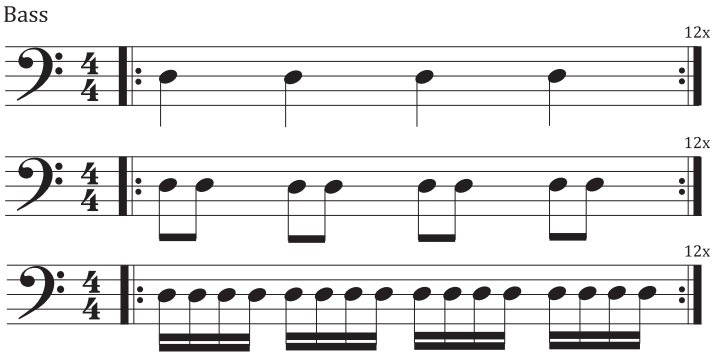


Figure 3.1: Performed evaluation bass pattern

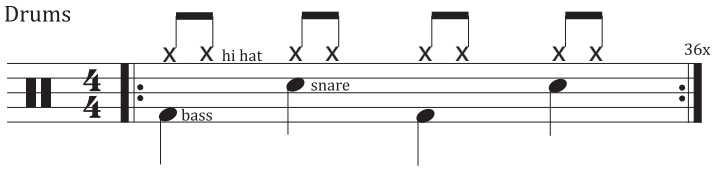


Figure 3.2: Reference one bar drum pattern

3.1.2 Evaluation

The results for each performing constellation are illustrated in figure 3.3. For each constellation it shows the delay acceptance value in ms (x-axis) against the speed and the note resolution of the performed pattern: Each constellation corresponds to a block of three columns, where the left column shows the performance in $\frac{1}{4}$ bass note resolution, the middle column in $\frac{1}{8}$, and the right column in $\frac{1}{16}$ resolution. The black color indicates a perfect playing condition, while the grey color indicates a tolerable situation. Beyond the maximal grey value the musical interaction was not possible anymore. The results show that the overall delay thresholds range between a minimal delay of 5 ms and a maximal delay of 65 ms.

The most important observation is, that the players do not exhibit a common latency acceptance value as the direct comparison for each bpm trial shows. E.g. the largest difference of 35 ms occurs between constellation 3 and 5 at a speed of 60 bpm with $\frac{1}{4}$ note resolution and between constellation 3 and 4 at a speed of 100 bpm with $\frac{1}{4}$ note resolution. The test results of the different players vary in such an extreme way that it is not possible to define a general valid delay threshold. It must instead be considered as an individual acceptance value, which depends on the player's rhythmical attitude. Apart from that outcome each player's graph shows a similar progression as the author will explain in the following paragraphs but the actual dimension differs significantly.

Furthermore, for each player it is obvious that – apart from a few exceptions – the delay acceptance threshold is directly related to the speed and the note resolution of the performed pattern. Generally a faster bpm and a higher note resolution lead to a lower acceptance threshold for each of the players, however, the transitions are more or less obvious: In some cases the threshold remains equal between subsequent trials, where either the bpm or the note resolution was increased (e.g. see constellation 4 at 100 bpm and constellation 1 at $\frac{1}{16}$ note resolution for 100 and 120 bpm. A surprising effect occurs with 4 constellations at 120 bpm: Here the $\frac{1}{8}$ note pattern could be better performed than the remaining patterns. Later the players's stated that speed of 120 bpm with a slightly laid back $\frac{1}{8}$ bass pattern represents the typical rock music style, which is why they felt more comfortable in this situation. Equally constellation 5 had problems with $\frac{1}{4}$ notes at 120 bpm: Although the bass player had no problems with delays at 5 and 10 ms , the drummer didn't like the trial's outcome due to

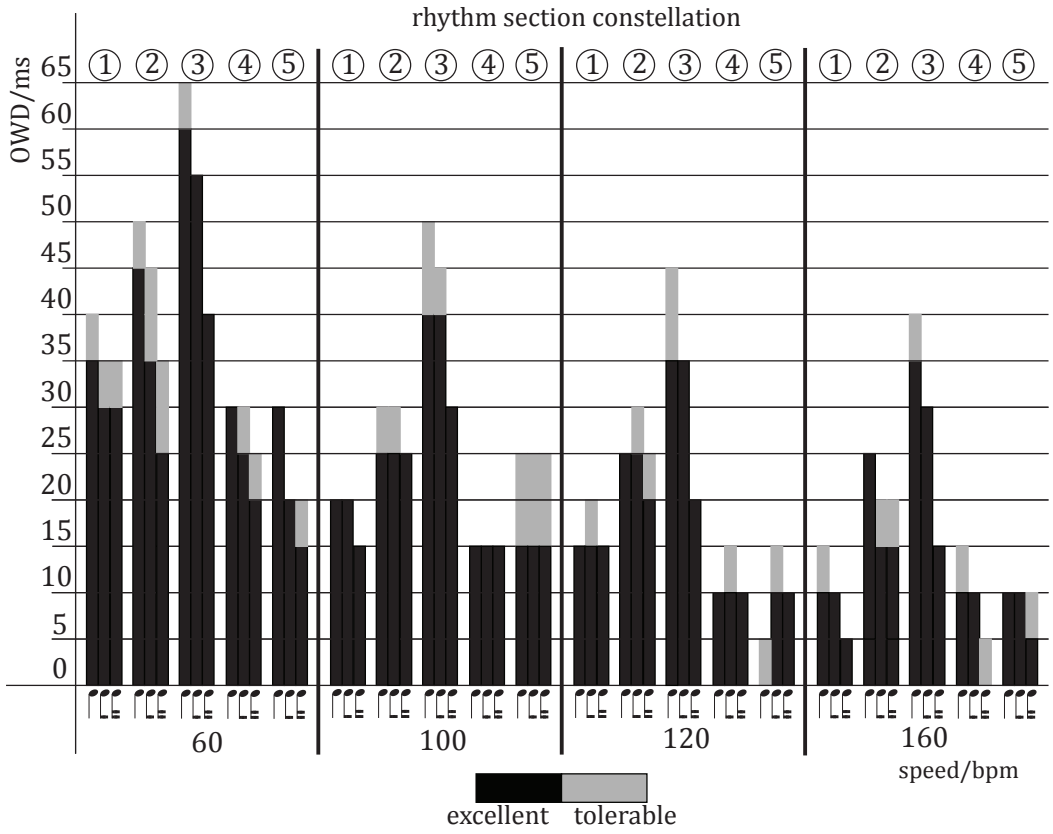


Figure 3.3: Test results of the performing drum/bass constellations

stylistic issues. Finally a repetition of the same trial with a delay of 0 ms didn't lead to any complaints.

Anyhow the graph does not illustrate aspects related to an artificial signal reverb: In none of the trials the author could observe a positive impact of this effect. In each parameter setting of an artificial reverb on either side the players rather complained about a lack of clear note onset and in turn preferred playing with the pure unmodified signal. Hence the author did not consider this effect as relevant and useful. Rhythm scheme changes from binary to ternary beats lead to an improvement of the playing conditions, therefore, the author did not consider them as useful information for the graph either.

3.1.3 Conclusion

According to section 2.5.5 the speed of sound of about 340 m/s results in signal delays depending on the physical distance between rehearsing musicians. Hence, two musicians‘ beats can never occur in precise synchrony. In this experiment the author

simulated this effect up to any desired dimension with an artificial delay line. In order to explain the outcome of this experiment in a clear and descriptive way the author introduces a model, which is illustrated in figure 3.4. It shows the time shift between a local and an external pulse. In the following this is defined as the “inter pulse delay" (IPD). According to this experiment's outcome the maximal IPD, music can be performed with, depends on the musician's personality and style. However, it decreases with an increasing rhythmical resolution and the speed of a tune: Depending on style and personal rhythmical attitude, musicians consider the time of a musical piece more or less as a fixed and precise beat reference. As a result a musician's motivation to play precisely on the beat can vary significantly and – depending on the skill – one might even consciously play in advance or behind the beat or in a changing manner. Depending on the value of this stylistic device it is possible to define a time spread around the theoretical beat reference, within a played pulse can be perceived and considered as correct. The so-called personal beat shift range (PBSR) describes a musician's temporal range of acceptance, which can be divided into the left range before the theoretical beat and the right range after the beat [54].

In case of a musical interaction it is the PBSR of a musician, which determines, in how far an external pulse is considered as correct or as out of time, however, the closer the external pulse resides at the range's edges, the more difficult and inconvenient the interaction will be. Figure 3.4 illustrates the closeness of an external pulse to a theoretical beat, which could have been generated with a precise metronome, by a continuous transition from white to black. The darker the color is, the larger the distance from the theoretical main pulse and the harder it is to interpret external pulses within this area as correct. As shorter notes result in a larger number of pulses in a fixed time frame, it is also clear that an increase of speed and a higher note resolution both correspond to a smaller PBSR. The first row of figure 3.4 shows a sequence of eighth note pulses. The incoming external pulses range twice as much in the darker area as they do in the lower quarter note sequence below. Any musician unconsciously accepts and copes with this phenomenon depending on the personal taste and favor. In order to compensate these effects of latency, musical gear such as stage or “Inear”-Monitors, rhythm devices and special band setups can be applied according to this personal rhythmical attitude and style.

Due to the described aspects numerous musicians generally do not exhibit the same PBSR, which is why it is not possible to define a common valid latency threshold, under which musical interaction is feasible or not. However, since a musical interaction involves at least two musicians, respectively two PBSR and the corresponding playing styles have to be taken into account. Hence, it is rather a collective latency-acceptance-limit, which determines the latency threshold, which defines the ideal conditions musicians can perform under. This so called “ensemble delay acceptance limit" $EDAL$ is typically not a known number and must be figured in a dedicated test setup. In the past the author assumed a common valid maximal delay threshold of 25 ms for any musician under any musical condition [44]. However, according to this

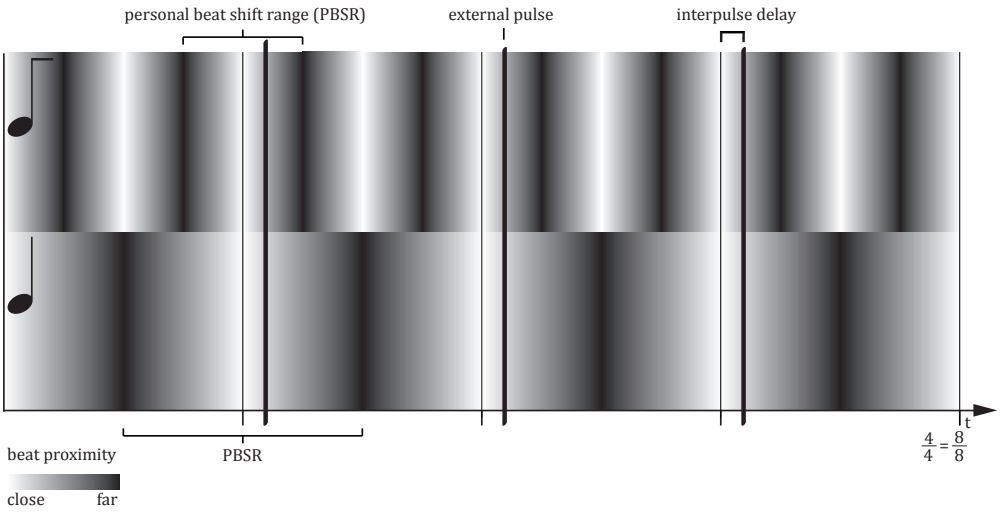


Figure 3.4: Visualization of human musical interaction

experiment and its findings this number can only be used as rough guide value since the maximal playable latency depends on a number of variable factors and therefore cannot be stated by a fixed number [54].

3.2 Comprehensive analysis of compromised musical interaction

As described in the previous section a delay beyond the $EDAL$ leads to an inconvenient or unacceptable playing situation. In this section the author investigates such high delay cases in terms of alternative music approaches and in how far they can improve the interaction. The first alternative is based on the principle of a displaced rhythm section like in the previous section, however, the high delays beyond the $EDAL$ will be compensated by applying an artificial delayed feedback as introduced in section 2.5.4. The second alternative assumes a separation of the rhythm and the solo section. In this case the author expects a higher latency acceptance of the non-delayed rhythm section in one place regarding a separated, high-delayed soloist.

3.2.1 Concept for displaced rhythm sections

Many musical instruments have a delayed onset, which is the latency gap until a musician gets audible feedback of a triggered note [98]. Furthermore, it is clear that musicians typically consider this effect as a normal part of their playing and unconsciously play notes in advance. Hence, this effect could be used in terms of

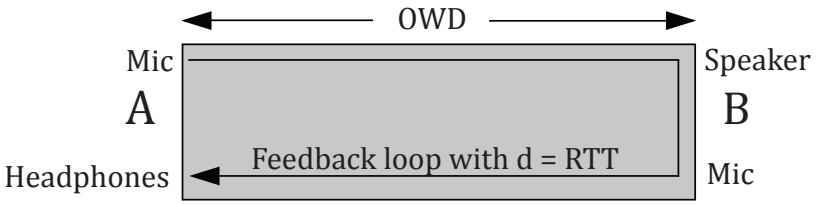


Figure 3.5: SDF (Single Delayed Feedback) with a real acoustic feedback loop made of a microphone and a speaker

latency compensation between two musicians: By muting one musician's direct signal he could listen to the remote side, which consists of the remote musician's instruments and his own signal. With respect to the latter the latency between the players appears twice – it takes the so-called "one-way delay" OWD to the remote player, before it enters the remote feedback loop in order to take again the OWD and finally return back to the sender. The overall latency refers to the term "roundtrip time" RTT. This principle is illustrated in figure 3.5 with two displaced musicians A and B. Since only player A is delayed the author calls this principle "single delayed feedback" (SDF). Since both parties listen to the same signal mix, the effective latency or "offset" between them would be offset=0 ms, which would in turn lead to a perfect synchronization between the players. Instead of applying a real feedback loop the same effect appears if an artificial self delay is used. In this case the player A could theoretically lower his own self-delay, which would in turn lead to an increase of the offset as shown in equation 3.1: If the self delay dF_{playerA} is decreased to values below the RTT the offset increases.

$$\text{offset} = RTT - dF_{\text{player}_\text{A}} \tag{3.1}$$

Instead of having one side delayed with the roundtrip delay it should also be possible to distribute half of the roundtrip delay $RTT/2$ to both players. In this case each player has to perform in the future and the time emerges half way between them. The author will call this principle of two self-delayed players "dual delayed feedback" (DDF). In such a situation the offset would depend on both player's self delay as expressed in equation 3.2 .

$$\text{offset} = RTT - (dF_{\text{playerA}} + dF_{\text{playerB}}) \tag{3.2}$$

In order to evaluate the playing conditions with the SDF and the DDF the author first chooses an OWD based on the results of the first evaluation phase, which have clearly lead to an unplayable situation beyond the $EDAL$. With each trial it is increased in steps of 15 ms. Furthermore, twice this value is added as an artificial delayed feedback to the bass player. Subsequently the bass player's self delay is decreased in steps of

5 ms in order to increase the delay offset. From this experiment the author expects clarification if how far the idea of an increased offset would still lead to a playable situation. In case of a positive evaluation the author will increase the offset up to a maximal offset acceptance limit. After the completion of this experiment the author applies the same one-way latencies but applies them also as the delayed feedback to both players. In this DDF trial the offset increase is realized by decreasing both player's delay in steps of 5 ms. Alternatively the author will distribute the total self delay asymmetrically in order to uncover any impact of this modified delay situation.

3.2.2 Concept for displaced rhythm/solo sections

In case a rhythm section plays in one location separated from a solo section, a different form of musical interaction can be approached: The drummer and the bassplayer provide the rhythm fundament of a piece and in turn their timing may not depend on the playing style of a solo saxophone player. Hence, rather than achieving an unplayable situation a different effect is likely to be expected: Although the latency resides at dimensions beyond a playable situation, bass and drum will be able to hold a solid groove, while the saxophone's solo will still sound right but "laid back". Playing "laid back" means to play slightly behind the groove, which musicians often try to achieve consciously in order to make their solo appear more interesting and free. In this case the saxophone represents the solo section, based on the rhythm section made of drums and bass. The saxophone has a perfect synchronization on its side and is transmitted back with the roundtrip time to the rythm section. This delay adds the artificial laid back style on it and hence this playing constellation is no longer to be considered problematic.

The evaluation concept of this theoretical approach is based on the same performing musicians and an additional professional saxophone player. According to the proposed principle altogether 5 rhythm section constellations play in one room with the delayed remote saxophone player in another room of the studio. As the "laid back" impression of a musical solo is the subject of this experiment the musicians play jazz standards at speeds of 60, 100, 120 and 160 bpm rather than a fixed set of rhythmical patterns. As in the previous experiments the delay is increased in steps of 5 ms until the the rhythm section complains about the saxophone player's solo. Again the players have the choice to evaluate the situation as "excellent", "tolerable" and "intolerable".

3.2.3 Evaluation

First the author started with separated rhythm sections, which perform with unplayable delays compensated with an artificial delayed feedback. As a first step the author chooses a speed of 120 bpm at $\frac{1}{8}$ note resolution and applied a one-way delay of 50 ms as due to the first experiment's outcome this parameter combination resulted in an unperformable situation for each of the constellations. Then he applied the SDF

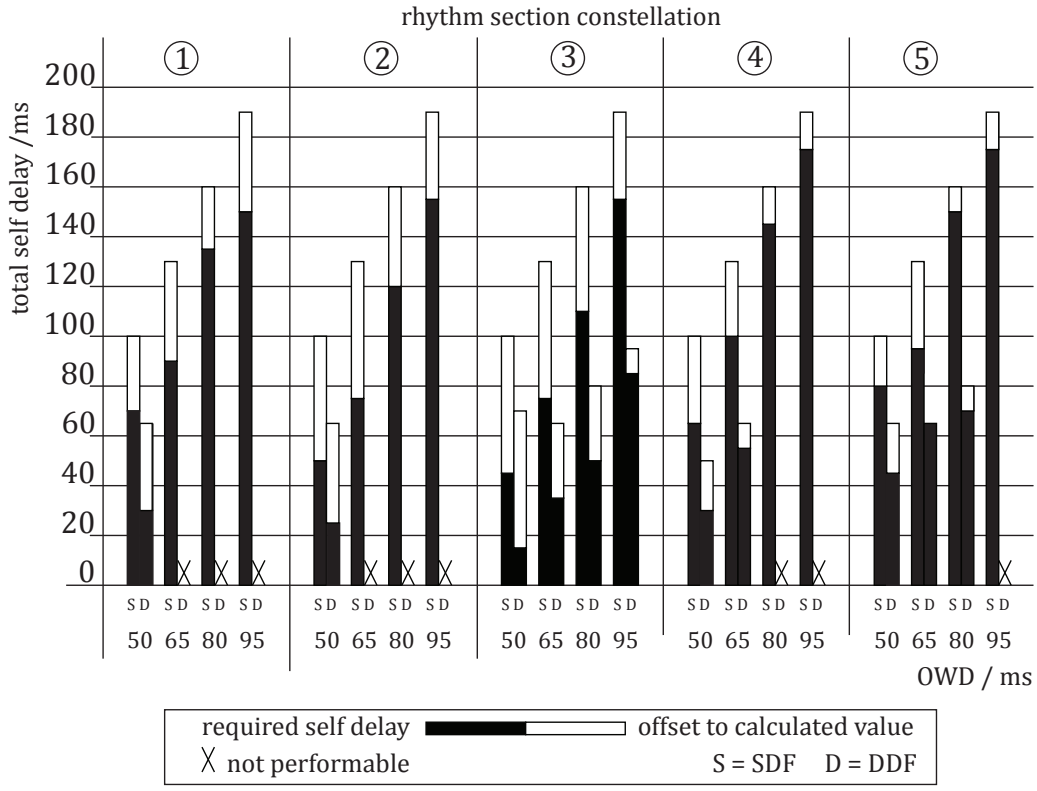


Figure 3.6: Test results of the performing drum/bass constellations with single and dual delayed feedback

and DDF as described in the concept and plotted the results in figure 3.6. According to the author's expectations the players could cope with a delayed feedback up to a certain limit. However, only constellation 3 was able to perform in each of the proposed trials but claimed that self-delays beyond 40 ms could not be considered as a natural situation anymore. Other drummers canceled the respective trials, where the delayed feedback exceeded their personal acceptance limit. Such trials are marked with an "X" and declared as not performable. Furthermore, the main outcome of this second experiment is that the delayed feedback can be decreased significantly. This statement is valid for SDF and DDF likewise. Asymmetric DDF trials are not considered since an additional practical analysis exhibited situations equal to the symmetric DDF or SDF.

Finally, in the last part of this experiment each rhythm section performed in one room with a remote saxophone player. The subject of this trial was the jazz standard "The days of wine and roses" of again 60, 100, 120 and 160 bpm, which consisted of one chorus of melody, one chorus of solo and another final melody chorus. The delay between the rhythm section and saxophone player was increased until the rhythm

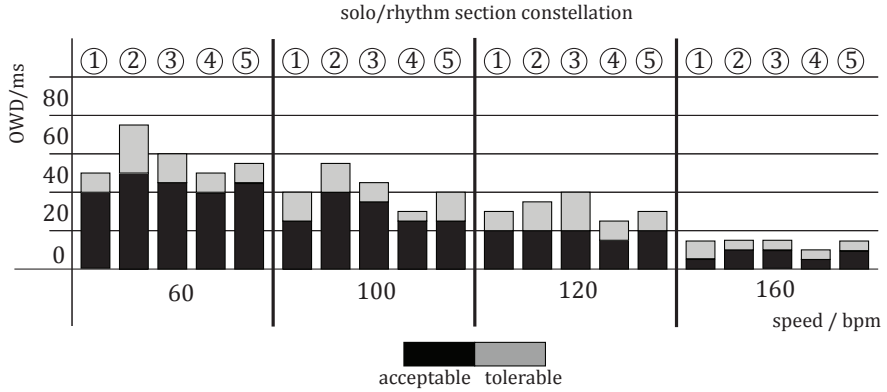


Figure 3.7: Test results of the performing drum/bass constellation with a remote saxophone solo player

section complained about a confusing melody or solo. Again the results were plotted as illustrated in figure 3.7. Again the graph indicates a perfect situation in black color and a tolerable solo in grey color. Beyond this maximal value the solo was perceived as "out of time". Again it is clear that each constellation's delay acceptance significantly decreases with the speed of the tune. Apart from the fact that the players again exhibit slight threshold deviations, the general delay acceptance resides at lower dimensions in comparison to a distributed rhythm section. However, this statement looses significance beyond a speed of 100 bpm.

3.2.4 Conclusion

In these two experiments it became clear that even beyond playable delays further compromised musical approaches can be applied: One possibility is to delay one musician's signal with the roundtrip delay. If the player is able to cope with the actual self delay the resulting delay gap is set to zero and results in perfect synchronization of the two players. In this situation the delayed player has to play ahead of the time. This principle corresponds to the term "single delayed feedback" (SDF). Furthermore, it is confirmed that the total delayed feedback can as well be symmetrically or asymmetrically distributed to both player in order to achieve the same playing condition. In this case the time of the song is performed "between" the players but can be shifted towards either direction by reducing and increasing the self delay on the respective sides. This principle corresponds to the term "dual delayed feedback" (DDF). Furthermore, it was clear that the delayed feedback can be reduced up to a certain point, which automatically leads to a delay offset between the players [54]. Figure 3.8 descriptively illustrates the three DFA modes: The block between players A and B corresponds to the OWD. If the players choose SDF as their alternative interaction approach only player A is delayed with a corresponding delay made of

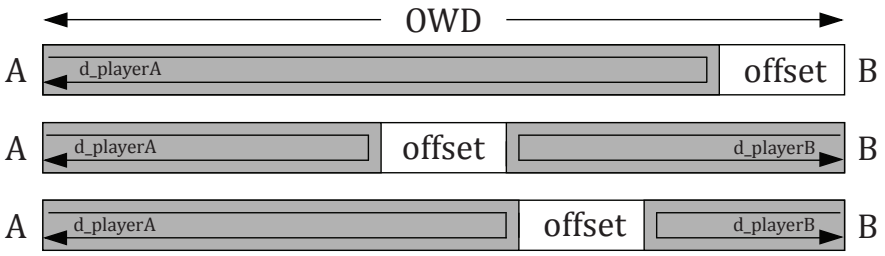


Figure 3.8: Visualization of the SDF principle (top) and the DDF principle with symmetric self delays (middle) and asymmetric self delays (bottom) including the delay offset

two times the OWD, which is equal to the RTT. However, the self delay dF can be lowered and leads to a delay offset. If the players choose DDF as their alternative interaction approach, also player B has to play with a delayed feedback. As clearly obvious this leads to a linear decrease of player A's self delay. The offset remains constant. Furthermore, the bottom drawing of 3.8 illustrates, that in DDF the required amount of self delay can also be distributed asymmetrically to the players.

In fact the involved musicians accepted delay offsets in significant dimensions. According to the measured numbers the author concluded to the following interrelationship between the OWD and the minimal self-delay dF_{min}: The minimal required total self delay dF_{min} can be calculated by equation 3.3, in which firstly the $EDAL$ is subtracted from the actual OWD and the result is multiplied with the factor 2. Secondly the equation adds a certain amount of delay due an undetermined human error ϵ, which occurs because of the inconvenience of the delayed feedback principle. In this experiment the author could observe that ϵ increases with the amount of self-delay and in some cases even leads to a not performable situation. Finally the actual offset can be calculated, by a subtraction of dF_{min} from the RTT as shown in equation 3.4 [54].

$$dF_{min} = (OWD - EDAL) \cdot 2 + \epsilon \tag{3.3}$$

$$\text{offset}_{max} = RTT - ((OWD - EDAL) \cdot 2 + \epsilon) \tag{3.4}$$

Nevertheless, the principle of delayed feedback means that the delayed players require to play an instrument with a low acoustic sound in order to prevent the reception of the undelayed direct noise. Hence, instruments such as acoustic drums, saxophone, violin or flute do not represent the recommended choice in comparison to their electronic counterparts. Though DFA improves the delay situation between two musicians, it is no doubt that a delay of one's own signal can typically be considered as

inconvenient and unnatural. The larger the delay gets and the louder the instrument's direct noise, the worse the realistic instrument feel and playing conditions.

The second alternative interaction approach lies in the separation of the rhythm and the solo section. In the experiment the same rhythm sections performed together on one side one specific jazz standard at various speeds with a remote saxophone player at delays beyond a normally playable situation. Due to this constellation the rhythm sections could keep a stable beat, while the saxophone player was received with an artificial "laid back" delay effect equal to the RTT. In how far this effect was considered as disturbing or not obviously depends on the speed of the performed tune. As expected, again the results exhibit slight deviations from player to player but as an overall conclusion it is clear, that in this distributed constellation – especially for slow pieces – most of the players accept a higher latency. Beyond 100 bpm this modified setup does not provide better performance conditions [54].

In addition to the evaluation of a realistic interaction scenario of the previous section, further alternative cognitive approaches have been introduced and evaluated. Even if the latency resides in a dimension, which prevents a realistic interplay, these approaches still provide an accurate rhythm-based interaction in order to perform conventional music such as pop, rock or jazz. In the following fourth chapter the author will outline any aspect related to latency introduced by technical equipment, network architectures and transmission media. The subsequent fifth chapter will finally combine the technical outcome of the fourth chapter with the cognitive conclusions of this chapter. In that context further alternative interaction styles will be introduced, which overcome the issue of delay by a change of the musical and rhythmical attitude.

Chapter 4

Technical fundamentals

As covered in the previous chapters, music perception including the process of music creation has undergone a comprehensive discussion. Today's various technologies in music recording, production and reproduction exist and have become both a common and an integral part of any musician's life. In contradiction, the special case of playing live music with another person abroad, is not an established or generally accepted idea [49] and hence guidelines, restrictions and experiences with this kind of scenario barely exist. Hence, in the previous chapter the author introduced the term "EDAL" (ensemble delay acceptance limit) to describe delay conditions, which match the situation with musicians in the same room. While in this conventional scenario the speed of sound and the actual distance between musicians determine a playable situation, a technology driven scenario implies further and different issues. In order to clarify, in how far recent technology supports or restricts distant live music play, this chapter introduces the physical and technical sources of delay and compares them regarding their actual latency dimensions.

4.1 Analogue electronic signal path

As we know from listening to radios or by using telephones, signals can be transmitted from a sender to a receiver. Unless special electronic devices like electronic pianos or synthesizers already generate an electronic signal, a signal conversion has to be applied. Like our eardrum membrane is stimulated by incoming air pressure waves, a microphone's membrane triggers a coil in a magnet field, which finally generates an electric output by the principle of induction. Similarly in an electric bass or guitar the vibrating string influences the electromagnetic field of the so-called sound pickup, which in turn generates the electrical representation of the vibration. Vice versa, loudspeakers can convert inaudible electronic signals back to audible sound signals. Here an electronic signal changes the electromagnetic field of a loudspeaker, which results in movements of an attached coil and speaker membrane and generates air pressure sound waves accordingly [63]. The generation of air pressure waves requires a speaker membrane to vibrate with a certain depth, which depends of the amplitude of the electronic signal. Since the output amplitude of a microphone or a pickup is

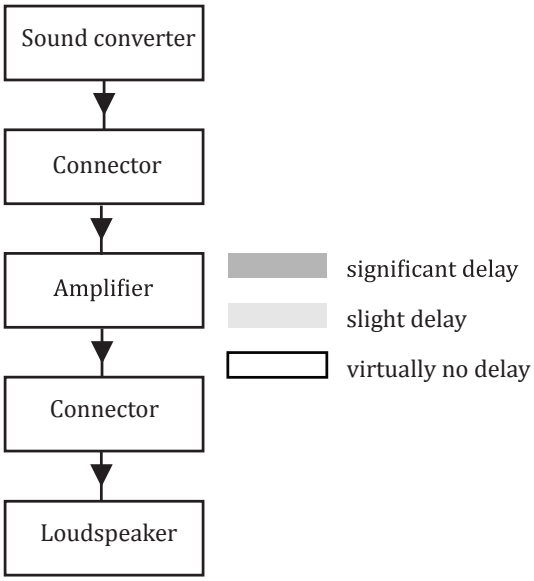


Figure 4.1: Analogue electronic signal path

too low it finally requires amplification in order to generate an audible sound at the speaker membrane. Like the transmission of sound waves though the air, this electronic signal chain represents an LTI system with three components – the microphone (or more generally the soundconverter), the amplifier and the loudspeaker. Typically each component is designed as a stand-alone device linked with wire connectors. This LTI system, as illustrated in figure 4.1, has to transmit each signal frequency equally. This is also the case in the natural transmission medium of air. The basic elements in the electronic signal path are resistors, capacitors and coils [126], and although this path implies a higher complexity compared with the air propagation, its transmission speed is significantly higher: On one hand – as the signal voltage analogously and immediately follows a respective input signal's wave form – analogue signal conversion does not introduce any latency [126], on the other hand in the electronic domain signals are transmitted with the speed of light, as stated in equation 4.1.

$$c = 300,000 \text{ km/s} \tag{4.1}$$

$$v_{\text{medium}} = c \cdot r_{\text{medium}} \tag{4.2}$$

The speed of light is almost 1,000,000 times faster than the speed of sound [40]. However, this value is only valid in a vacuum under laboratory conditions and cannot be reached with real cable connections under realistic conditions due to the so-called "reduction factor" r_{medium} [127]. The reduction factor is a conductor-dependent number,

which describes the amount of speed reduction of the specific material in comparison to the propagation in a vacuum. The actual transmission speed a medium delivers can be calculated by equation 4.2. A copper cable connection exhibits a reduction factor of 0.77 [127]. Hence, connecting a sender and a receiver in a real live environment, the transmission speed reaches 77 % of the speed of light. Assuming an EDAL of 25 ms, the theoretical cable length might hold a maximum of 5,775 km.

$$\frac{300,000\ \text{km}}{1,000\ \text{ms}} \cdot 25\ \text{ms} \cdot 0.77 = 5,775\ \text{km} \tag{4.3}$$

Of course it would not make sense to run a custom cable of hundreds of kilometers underneath the earth to a remote musician. On top of enormous work and material costs the electronic signal looses energy on the way to the destination and after several kilometers its amplitude would finally fall under usable values resulting in an unacceptable signal-to-noise-ratio (SNR) [79]. As a consequence, amplifiers could be installed in order to reincrease the amplitude and finally get rid of this effect but in a way this work has already been done by telephone companies. They provide a network of cables, theoretically any home can be connected to. In a regular phone call the speaker talks into the microphone which converts the voice into an electronic signal, which is transmitted on a cable network to the listener's end, where the loudspeaker of the receiver converts the electronic signal to an audible sound signal. Unless the total cable length exceeds the previously mentioned value, a music performance would be possible but still certain limitations do exist: The participants of a phone call typically cannot know the cable length since the precise path of a cable normally is an unavailable figure. In barely any case a direct point-to-point connection exists so that detours depending on the telephone provider's network structure have to be taken into account. In fact a descriptive example for the wiring and switching processes in the analogue telephone network is given by figure 4.2, where an employee in a switching center of the 1960s manually rewires input and output jacks in order to establish a cable connection between two phone parties. Today these processes are automated.

The conversion of a sound signal to its electronic representation and transmitting it, falls under the domain of analogue signal processing. Analogue signals require an analogue network such as the old phone network, which the author just described. Though this outdated network could have been used for the exchange of musical data, it barely or actually never happened. The main reasons lies in the fact, that it had been designed for the purpose of speech transmission and hence was typically limited to a quality of 4 kHz [86]. This sufficed in terms of understanding a remote speaker, however, in terms of music the quality can be considered as poor. Secondly the issue of latency has a lower significance in a phone conversation, for which delays of up to 400 ms can be considered as still acceptable [88]. Rather than explicitely informing a customer about the delay to a remote speaker, phone companies simply tried to provide delays below this threshold, so that customers hardly developed a conscious

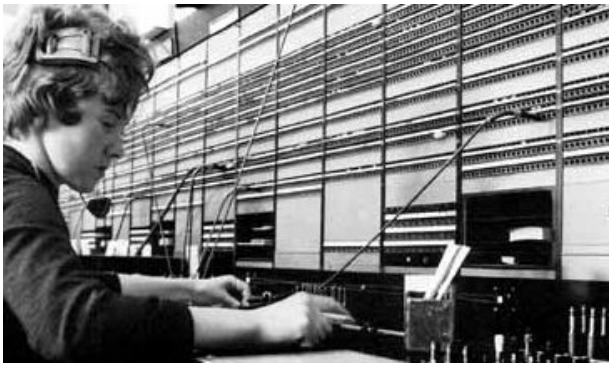


Figure 4.2: Employee in a 1960s phone switching center [6]

perception of the actual latency. In turn the awareness of using such lines for even more unconventional and delay critical applications like music did not exist. Thirdly phone conversations – especially long distance calls – were extremely expensive, so that musicians did not consider this technology as an alternative to the conventional way of music creation with physically moving or traveling to distant players.

4.2 Digital electronic signal path

The previously described analogue signal processing assumes the existence of a time continuous and values continuous signal [117]. Any natural sound signal represents a time continuous signal because it continuously exists at any moment in time. The values of an analogue signal do not underly any limitation in terms of resolution, showing even tiniest changes of a signal's amplitude. Their value and time resolution is infinite – in contradiction to digital signals, which are time and value discrete [151]. They only show values in certain temporal intervals and hence cannot be present at any moment of time. Due to the value discreteness digital signals also contain values in a certain resolution. Again – in contradiction to time continuous signal – this leads to a rougher signal representation. Discreteness obviously implies drawbacks in comparison to continuity, hence one might question why digital technology has substituted analogue technology. In fact there are numerous reasons, which can be summarized with the statement that a digital signal is much easier and safer to handle than an analogue signal. Firstly a digital signal consists of numbers, that can easily be stored on a digital medium such as harddrives or compact discs (CD) [79]. Secondly these numbers can be copied without any loss so that identical signal clones can be created. Furthermore, in terms of signal transmission digital signals do not have to cope with the SNR anymore: An analogue signal's amplitude weakens over transmission time and hence has to be amplified again in order to keep a certain level and not make it fall too close to the noise threshold. Unfortunately, the more amplification stages that are applied, the more noise will be amplified, also in turn

the SNR increases [63]. In contrast, the values of a digital signal are represented by binary digits (bits), which are either zero or one. In case of a signal transmission a receiver only has to distinguish between these two states. Electronically a one is existent, if voltage is on the wire, a zero is existent if no voltage is on the wire. Of course also in in this case, the voltage lowers over transmission time, however, as long as it remains over a predefined level, a one can still be interpretated properly. Once all bits have reached their destination, the signal can be reconstructed accordingly and noiselessly. Another advantage lies in the fact, that digital technology follows the rules of the boolean algebra, even complex circuits can be described and tested with. Even if a signal lacks certain bits by transmission errors, these errors can be corrected by logical boolean calculations and error correction schemes [147].

As musical instruments generate analogue signals and equally our ears respond to analogue input signals only, a signal would require an analogue-to-digital conversion in order to take advantage of digital transmission principles and a following digital-to-analogue conversion in order to bring it back to its audible analogue form. This process includes a number of electronic stages, which are outlined in figure 4.3. In the following subsections the author will explain any of the involved stages respectively and especially in terms of the introduced latency.

4.2.1 Anti aliasing filter

As described in context with the analogue signal path sounds can be recorded with microphones and the analogue output can be used directly without any signal modification in terms of amplification or transmission. In digital signal processing things are more complex: A digital system requires input signals not containing frequencies beyond a desired maximal frequency. The reason for this will be described later on in the context with the Nyquist-Shannon theorem [138]. However, in order to fulfill this precondition a digital system requires to have a so-called anti aliasing filter [119], which functions as a low pass for the input signal. The simplest electrical circuit that provides low-pass filter functionality is a voltage divider, which consists of a resistor in series with a load and a capacitor in parallel with the load. The capacitor exhibits reactance for low-frequencies, causing them to pass the load instead. At higher frequencies the reactance drops, and the capacitor effectively functions as a short circuit. The combination of resistance and capacitance allows to calculate the time constant of the filter $\tau = R \cdot C$ [126], which determines the cutoff frequency f_c in hertz via equation 4.4.

$$f_c = \frac{1}{2\pi \cdot \tau} = \frac{1}{2\pi \cdot R \cdot C} \tag{4.4}$$

The required filter dimensioning depends on the current field of application with the desired maximum frequency. In the current case of audio processing this cutoff frequency lies at 20 kHz for full range audio – the maximum frequency our ears are

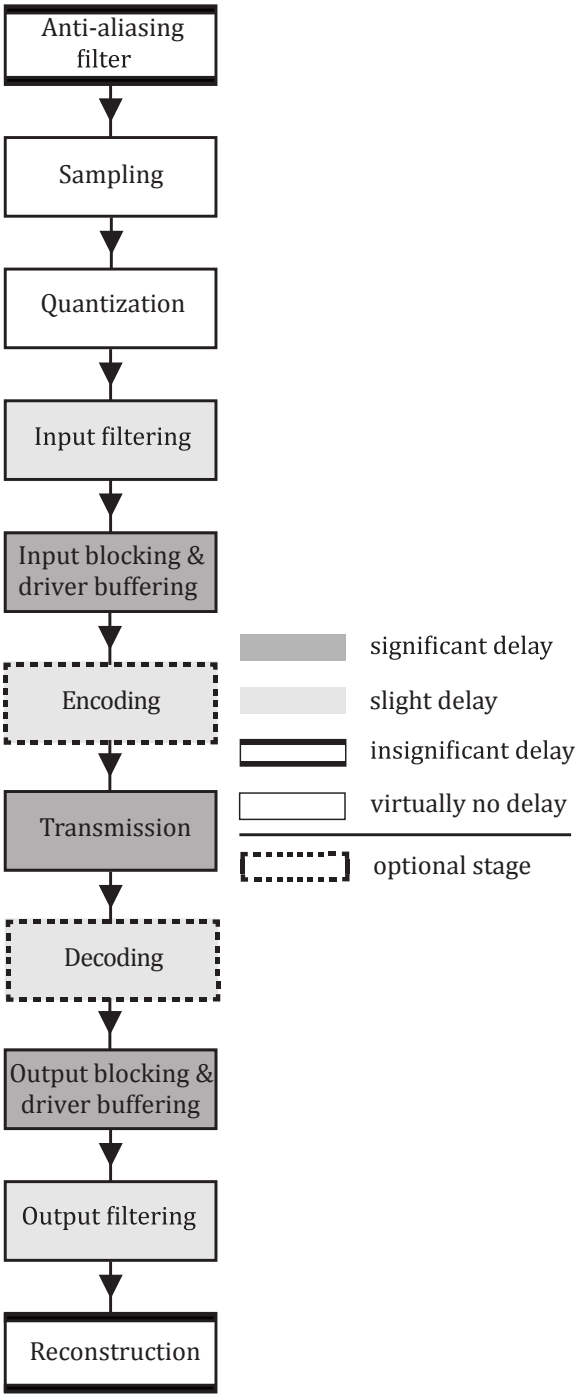


Figure 4.3: Stages of a digital audio signal path

able to work with. Of course a microphone could itself already act as a low pass if it sufficiently attenuates frequencies beyond 20 kHz. Hence, an additional low pass might not be needed but in case we consider a lower maximal frequency of e.g. 16 kHz, 12 kHz or possibly 4 kHz as it is the case for low quality audio, this low pass filter guarantees the required frequency cutoff in any case for any kind of input signal. In fact the anti-aliasing low pass filter introduces a delay according to the time constant τ, which results in 8 μs if the cutoff frequency is set to 20 kHz.

4.2.2 Sampling

The conversion of a time continuous signal $f(t)$ with the time basis T into a time discrete signal $f[n]$ with the time basis $T = T \cdot N$ is termed out as "sampling" [146]. After the sampling process at moments $n \cdot T$ the values of signal $f[n]$ should equal the ones of the time continuous signal $f(t)$. The Dirac pulse (introduced in chapter 2) is predestinated to fulfill such task: Due to its narrow shape the Dirac pulse allows to sample one specific value of a time continuous signal at a specific moment by multiplying both signals. The resulting signal holds the actual sample value at the moment $n \cdot T$ and remains zero otherwise. In that context it is important to mention that the Dirac pulse only represents a theoretical construct: It can practically be described as a rectangular pulse of very small width and a limited amplitude. The latter is determined by a system's operating voltage, however, it is generally associated with the value 1.

In order to retrieve n sample values instead of a single one, we choose a train of Dirac pulses instead, which is a number of Dirac pulses displaced by a fixeded time interval T. This princinple is illustrated in figure 4.4. The train of Diracs $\Delta_T(t)$ is also called the "ideal sampling function" as described in equation 4.5.

$$\Delta_T(t) = \sum_{n=-\infty}^{\infty} \delta(t - n \cdot T) \tag{4.5}$$

The product of the time continuous signal and the train of Diracs results in signal $f_s(t)$ described in equation 4.6:

$$f_s(t) = f(t) \sum_{n=-\infty}^{\infty} \delta(t - nT_s) = \sum_{n=-\infty}^{\infty} f(nT_s) \cdot \delta(t - nT_s) \tag{4.6}$$

In that form the signal still represents a time continuous signal, which is zero apart from the discrete intervals n. In order to loose the dependency of time, it is also possible to list the discrete values in a sequence which in turn finishes the process of time discretization. Mathematically this process is described in equation 4.7:

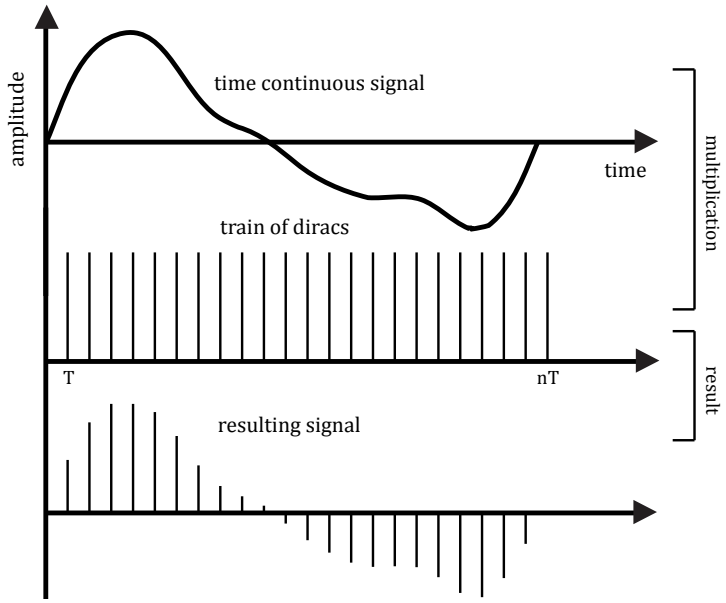


Figure 4.4: The sampling process

$$f[n] = \sum_{n=-\infty}^{\infty} f(nT_s) \cdot \delta(n) \tag{4.7}$$

The train of Dirac pulses is triggered by the positive shoulder of a sound card's rectangular word clock signal [119] in the so-called A/D-converter [119]. Its frequency is also termed out as the sampling frequency. This frequency is determined by Nyquist-Shannon or sampling theorem, which states that a time-continuous, band limited signal with a maximal frequency $f_{\max}$ requires a minimal sampling frequency of $> 2 \cdot f_{\max}$ in order to reconstruct the resulting time discrete signal values back to its original time continuous form. Only if that precondition is given, can the original shape be retrieved – otherwise signal distortions occur. This general problem and interrelationship is more descriptive when transforming the respective signals into the frequency domain as illustrated in figure 4.5: The fourier transformed train of diracs is also made up as a train of Diracs, each separated by the sampling frequency. Furthermore, the graph shows an example of an input signal, whose continuous spectrum linearly decreases until the maximal frequency is reached. In time domain both signals are multiplied but as a multiplication in the time domain corresponds to a convolution in the frequency domain [151], the resulting convolved spectrum shows the input signal's spectrum at each moment a Dirac pulse is present. Hence, the resulting spectrum contains additional spectra shifted by n times the center sampling frequency – a periodic repetition of the spectrum S(f) in 1/T steps. It is also clearly visible that the total spectrum has broadened by factor 2, centered and flipped around

the sampling frequency. As the sample rate increases the gap between two successive spectrum repetitions increases. Decreasing the sample rate leads to the opposite case and can lead to a spectrum overlap if the sample rate does not reside beyond twice the maximal expected signal frequency, that is, when the lower slope of the next spectrum overlaps with the upper part of the current spectrum as clearly visible in figure 4.5. This overlap leads to signal distortions and is described as aliasing [63], which represents the most undesired effect in the sampling process. As the upper audible threshold lies at 20 kHz the minimal required sampling frequency has to be 40,001Hz, however, in the practical application this leads to problems: Due to weak slew rates of analogue filters, the antialiasing input filter cannot guarantee a precise signal frequency cutoff and still introduces frequencies beyond the desired threshold, which in turn violates the Nyquist threshold. This effect can be compensated by shifting the subsequent spectrum further apart, which is realized by choosing a slightly higher sampling frequency. With respect to the CD technology 44.1 kHz is the standard sampling rate, while 48 kHz has become a standard rate for sound studio equipment such as digital recorders or sound cards [79]. The problem of aliasing will be further outlined later on in context with the reconstruction process.

4.2.3 Value discretization

After the sampling process, the generated signal exists in a time discrete form but its sample values are still represented by their current electronic voltage in a continuous form. Thus, as the final step towards digital signal conversion, each sample value requires to be discretized as well. This process also refers to the term “quantization” and applies an approximation of the continuous range by a relatively small set of discrete integer values right after a sample generation. Especially in audio processing this approximation is typically performed in a resolution of 16 Bit, so that the value of a time discrete signal can vary between 0 and $2^{16} - 1 = 65535$. The value of 0 is applied to 0 volts and the highest value of 65535 is applied to the maximal continuous signal voltage.

In that context the two reference levels “decibel volts” (dBV) for consumer applications and “decibels unloaded” (dBu) for professional applications exist. The reference voltage for the decibel volt (0 dBV) is 1 V_{RMS} and the reference voltage for the decibel unloaded (0 dBu) lies at 0.7746 V_{RMS} [79]. The most common nominal level for consumer audio equipment is -10 dBV with a corresponding peak amplitude of approximately 0.1414 V, and the most common nominal level for professional equipment is +4 dBu with a corresponding peak value of approximately 1.736 V [79].

Finally discretization leads to a signal shape illustrated in figure 4.6, in which each sample in assigned with the respective digital value. If the bit resolution was rougher, the “stairstep“ characteristics of the digital signal would be steeper and could possibly not distinguish between two slightly differing values. Hence, it would finally interpret them as equal.

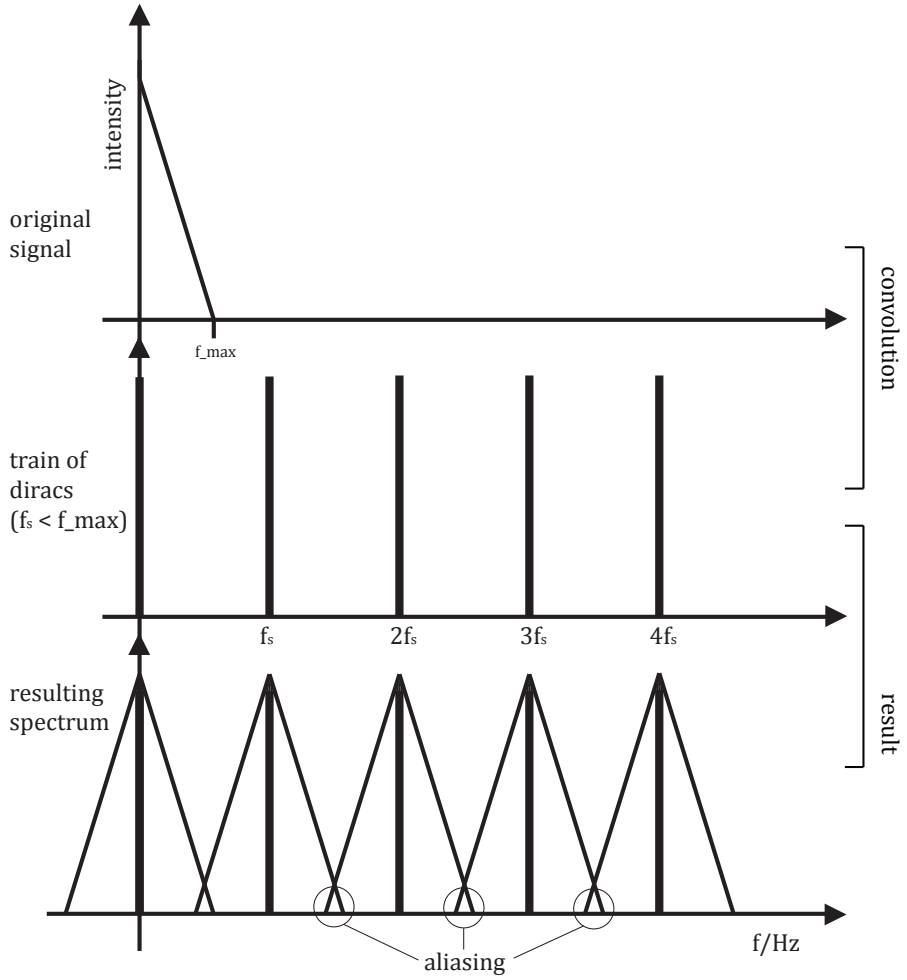


Figure 4.5: The effect of aliasing

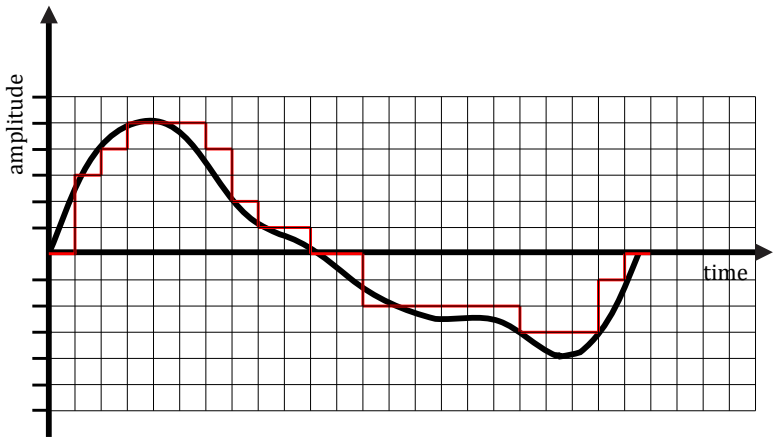


Figure 4.6: Principle of value discretization

The difference between the actual analog value and quantized digital value is called quantization error and leads to quantization noise. As quantization noise is non-linear and signal-dependent, it can be modeled in several different ways. As a special example, which assumes that the quantization error is uniformly distributed and based statistically on a full-scale sine wave of peak amplitude, the SNR can be calculated by equation 4.8. Q represents the bit resolution of the A/D-converter, which in case of a 16 Bit A/D-converter would lead to a maximum SNR of 98.09 dB. If the peak amplitude changes, the value 1.76 would change accordingly. Further details on how to calculate the quantization noise can be found in [119].

$$\mathrm{SNR}_{\mathrm{ADC}} = (1.761 + 6.0206 \cdot Q) \ \mathrm{dB} \tag{4.8}$$

4.2.4 Digital input filtering

Theoretically, after the sound has changed its representation from analogue to digital, the transformation process is finished. Practically electronic devices generally apply additional signal treatment in terms of quality improvement, which in the input stage addresses artefacts due to aliasing. As the data is already present as a set of binary numbers, this task is realized by digital filters, which are of an extremely high precision in comparison to their analog counterparts [151]. Since the analogue anti-aliasing filter suffers from a low slew rate, even a sample rate of 48 kHz might still lead to aliasing. In order to reduce this effect digital filters apply an artificial oversampling [119] by inserting interpolated sample values. Consequently, by calculating a fixeded number of up to 64 values, the subsequent signal's spectra are shifted further apart. In this

higher sampling resolution further digital low-pass filtering is applied to the signal. In the subsequent step the upsampled and filtered signal requires a decimation or downsampling in order to match the original or desired sampling frequency [154].

Rather than applying an artificial oversampling, the so-called Sigma-Delta A/D converters apply a real oversampling by working at a higher sampling frequency. In turn the digital input filter functions as a low pass and a decimator only. Furthermore, in order to reduce a digital signal's quantization noise, the quantization error between the quantized value and the original value is measured and stored. Since the digital filter is designed as a feedback loop, this quantization error is fed back to the input and subtracted. Equation 4.9 represents a mathematical description of such a feedback loop, where y is the final output sample value, x is the input sample value, n is the sample number and $E(x(n-1))$ is the quantization error of the previous sample $y(n-1)$. Hence, the output sample is equal to the input sample minus the quantization error of previous sample [119]. Nowadays the majority of sound cards are equipped with the Sigma-Delta converter technology [154].

$$y(n) = x(n) - E(x(n-1)) \tag{4.9}$$

What a filter precisely does is a vendor and device specific criterion and in practice modern filter design is typically more complex and fine grained. The interested reader can find a detailed overview of noise shaping techniques in [119]. However, digital filters require a number of input samples to work properly – the so-called filter length [138]. After the retrieval of this fixed amount of samples the digital filter generates output samples according to its actual filter architecture. Hence, a digital filter introduces an additional delay depending on the number of required input samples and in turn the higher the sample rate the lower this delay will be as samples are generated faster. As an example, the professional RME Multiface sound card's input filter requires 31 samples, which corresponds to a delay of 0.65 ms at a sample rate of 48 kHz and 0.32 ms at 96 kHz respectively [31].

4.2.5 Device input blocking and driver buffering

Sound cards can provide each generated sample to the next processing stage, however, current PC-based sound systems are not able to process data with that processing speed due to internal scheduling and general architecture: On an abstract level, the time to process n samples must be constant in order to provide a solid audio stream. This is described in equation 4.10, where $t(n)_{\text{processing}}$ represents the total processing time for n samples. $t(n)_{\text{functioncall}}$ is associated with the execution time for n samples with the respective system call. The problem occurs due to the implied function overhead, which increases the execution time with a constant value t_{overhead}. Hence, the lower the number of samples n, the more function calls the system has to perform within a given time frame, and in turn the stronger the impact of the overhead.

Respectively, the higher the number of samples to be processed, the less often the function is called in a given time frame and in turn the overhead decreases.

$$t(n)_{\text{processing}} = t(n)_{\text{functioncall}} + t_{\text{overhead}} \tag{4.10}$$

Furthermore, another source of delay occurs due to the fact that a personal computer is not designed in terms of explicit audio processing. In that context – rather than immediately executing the respective processes – the system's scheduler serves them at a certain moment in time. As a result, it takes an amount of time to transfer an audio sample to the user space of a system. The precise value varies from system to system, however, in any case it resides significantly beyond the interval between two subsequent audio samples. As a result the system has to collect an amount of samples before actually processing them. Hence – rather than sample by sample – sound cards generate audio in blocks of a fixed number of samples. Once the sound card has collected this fixed number of samples, it generates an interrupt and addresses the block of samples via DMA (direct memory access) [46] to the machine. A sample block also corresponds to the term "sample frame" or "audio buffer". The size of each processed block depends on the actual system's performance and is limited to a number of 32 samples as the lowest adjustable value. If we assume audio data to have a 16 Bit resolution – that is one sample consists of 2 bytes – the currently lowest possible block size equals 64 bytes. Without performance optimization of a device's operating system in the form of a special low latency kernel or further technical improvements such as the use of a realtime operating system, current personal computers can handle block sizes of 512 samples/block. More recently – with increasing CPU speeds – devices can work with 128, 64 or even 32 samples/block. In any case, blocking implies a tradeoff between stability and delay: On one hand larger blocks require less computational power so that a more stable system behavior can be achieved – on the other hand the delay increases with larger block sizes since it needs more time to generate an appropriate number of samples. Given a fixed sample rate, the blocking delay is directly related to the block size. Assuming a fixed block size and increasing the sample rate, higher frequencies can be captured, which leads to a higher audio quality. Additionally – due to smaller sampling intervals – it as well leads to proportionally smaller delay times so that altogether the blocking delay depends on the block size and the sample rate. According to this the blocking delay can be calculated with the following equation 4.11.

$$\text{blocking delay} = \frac{\text{block size}}{\text{sample rate}} \tag{4.11}$$

After the sound card has blocked for a certain amount of samples, a filled block or buffer is available to be processed by a digital device or computer application. Practically this means, that the buffer values can now be modified in any way a user desires to. As an example the signal's data could be scaled, distorted, muted, etc.

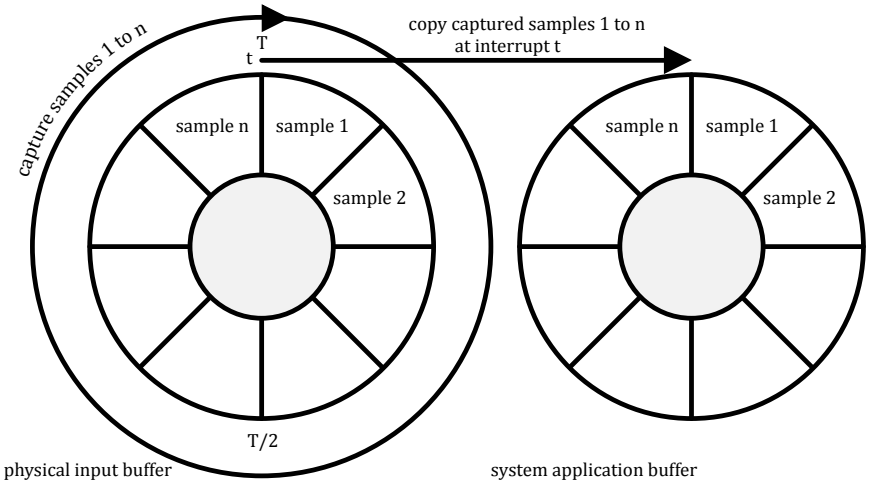


Figure 4.7: Audio capture process

In any case of modification, the existing sample values of the current block need to be changed according to the rules of digital signal processing: In case of a muting all values need to be set to zero, scaling requires all values to be multiplied by a fixed desired factor, while distortion assumes a varying multiplication factor over the appropriate block values. In common computer operating systems (Windows, Mac OS X, Linux), so-called "callback functions" [35] are used by the audio device as a programmer's interface for the retrieval and modification of such audio blocks. Their execution corresponds to a sound card's interrupt and according to the previous equation, this interval depends on the actual sound card settings. As a standard case a sampling rate of 48 kHz with a frame size of 128 samples results in callback events every 2.7 ms. Figure 4.7 shows the principle of the capture process, which is realized as a double buffer system. E.g. a block size of 512 samples corresponds to a blocking delay of 10.8 ms at a sample rate of 48 kHz. The digital application processes the filled buffer's n samples, while at the same time another buffer is filled with the next n samples. This is further outlined in section 4.2.9 and illustrated in figure 4.12. Once the buffer is full, both buffers are swapped and the next period starts in the same way. At the moment t the callback event occurs after all sample values had been captured and made available to the application. The interval T represents the blocking delay. At that state the sound card has just captured audio data, which is now ready to be processed and actually ready to be sent to a desired destination. However, the system has not created any audible feedback yet. The direct playback of a captured signal block also refers to the term "loopback" [30] and would require the buffer to pass the sound card's output process as the next step. This playout process is described later on after the following transmission section. Figure 4.8 graphically shows the previously described relation between block size, sample rate and a sound card's blocking delay.

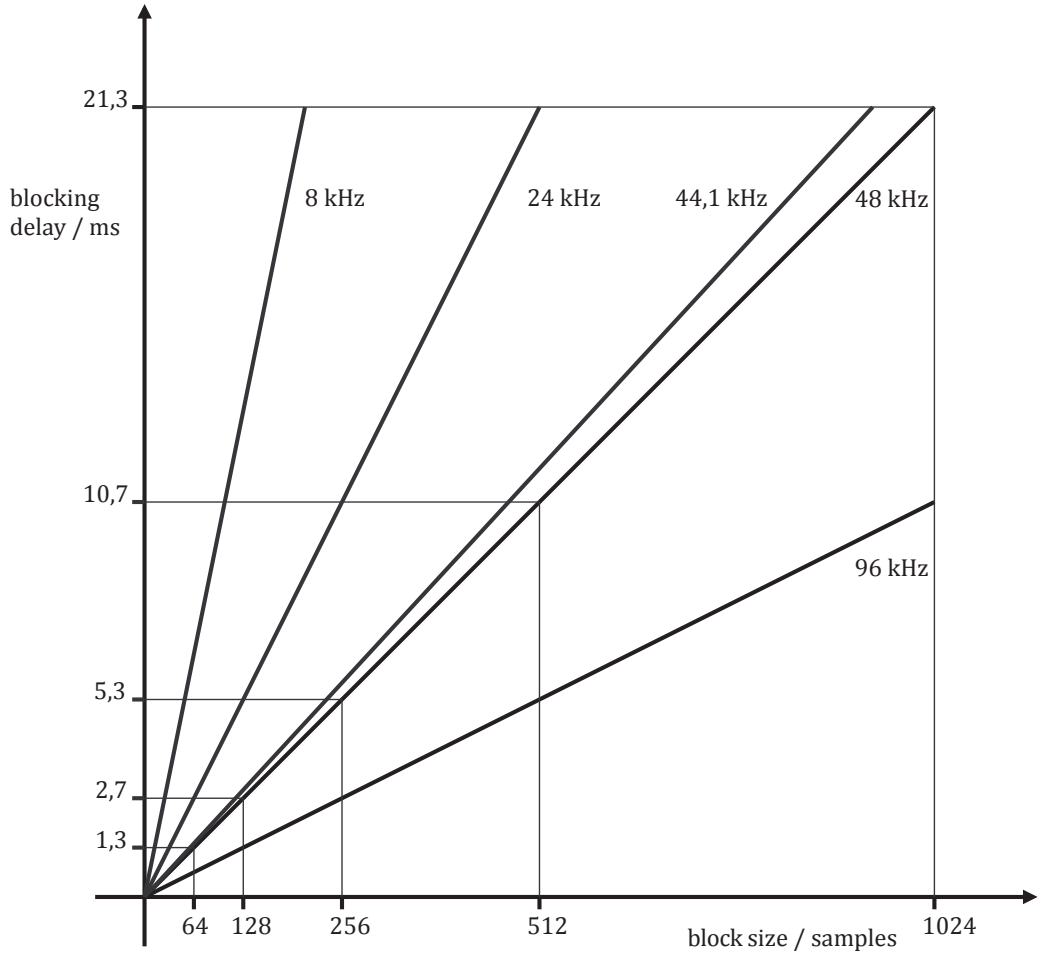


Figure 4.8: Audio blocking times

Nevertheless, one has to be aware that on top of the input filter- and blocking delay, there can be additional audio driver latencies depending on the actual hardware technology and operating system. E.g. in contradiction to PCI bus data transfer with the corresponding fast direct memory access, the Firewire [115] audio driver introduces an additional so-called "safety buffer" [30] of at least 64 samples in order to provide a stable audio capture. Furthermore, the "Core Audio" of the Apple Macintosh operating system OS X e.g. requires the so-called "safety offset" of at least 64 samples up to even 292 samples due to its general architecture, which is described in [85]. In any case, additional driver latencies approach an improvement of the sound card's stability and performance but as this conflicts with the low delay requirements of network music performances it eventually is the user's responsibility to figure the best combination of sound card, operating system and driver.

4.2.6 Encoding (optional)

After the retrieval of an audio block it could theoretically be sent directly across a network. The following section 4.2.7 will outline the requirements of an umcompressed audio stream regarding a network connection. However, if these requirements cannot be met, the audio stream data must be reduced. Hence, signal compression can be applied optionally before transmitting the actual audio block on the network. In that context we have to distinguish between "lossless" and "lossy" compression techniques: Lossless data compression allows the exact original data to be reconstructed from the compressed data. Depending on the mathematical algorithm and the actual content, lossless compression achieves a compression rate of approximately 2 by either generating a statistical model for the input data or mapping the input data to probability bit sequences [140]. This happens in such way that more frequently encountered data will produce shorter output than less frequently encountered data. The latency, introduced by the lossless encoding of an audio block, can be considered as relatively low with values less than 200 μs depending on the machine's processing performance. Hence, in the case of lossless audio compression the final encoding delay $d_{\text{encoding(lossless)}}$ is only made of the encoding computational delay $d_{\text{encodeComp}}$, which a machine requires to encode the current audio block as shown in equation 4.12.

$$d_{\text{encoding(lossless)}} = d_{\text{encodeComp}} \tag{4.12}$$

Despite the possibility of perfectly reconstructing the original signal with a minimal amount of delay, lossless compression suffers from a low compression rate and hence does not necessarily meet the bandwidth requirements of narrow band consumer network connections. An alternative are the so-called lossy data compression techniques, which achieve comparably higher compression rates. The principle of lossy audio compression is based on the phenomenon of psychoacoustics that under certain circumstances related to signal frequency and amplitude, frequencies of a signal are masked by others: Specific signal parts must exhibit a certain intensity in

order not to be masked by other signal parts. The required intensity refers to the term "masking threshold" [140]. In order to achieve compression rates of 4 up to 16, lossless coding forms the quantization noise of the signal according to the masking threshold. Thus certain signal parts are encoded with a higher bit resolution than others [140]: The lower a frequency remains below the masking threshold the less bit resolution is used for the encoding. The tradeoff, however, lies in the fact, that due to the reduced bit resolution lossy audio compression only allows an approximation of the original data to be reconstructed. Furthermore, it introduces a higher delay, which consists of several factors: The final encoding delay is made of the algorithmic delay d_{alg} and the computational delay $d_{\text{encodeComp}}$ as shown in equation 4.13. More detailed, equation 4.14 shows that the algorithmic delay is made of the blocking delay d_{blck} and further codec related delays. The blocking delay corresponds to a block of a fixed number of samples, that a codec requires in order to calculate an encoded output block according to its technical architecture. The blocking delays of conventional and often applied codecs such as AAC [140] or Ogg Vorbis [75] already reside at 1024 samples and a corresponding delay of more than 20 ms at a sample rate of 48 kHz.

$$d_{\text{encoding(lossy)}} = d_{\text{alg}} + d_{\text{encodeComp}} \tag{4.13}$$

$$d_{\text{alg}} = d_{\text{blck}} + d_{\text{others}} \tag{4.14}$$

However, with respect to the actual stage in the digital signal path, an audio block has already been captured and in turn the codec blocking delay has no impact on the overall application encoding latency anymore. Therefore, the codec blocking delay can be subtracted from the sum of coding latencies in this specific application. Finally, it is the computational delay and the codec specific delay, which remain in order to calculate the final lossy application encoding delay $d_{\text{appEncoding(lossy)}}$. This context is described in equation 4.15.

$$d_{\text{appEncoding(lossy)}} = d_{\text{encoding(lossy)}} - d_{\text{blck}} = d_{\text{encodeComp}} + d_{\text{others}} \tag{4.15}$$

According to the name of its index, the delay d_{others} refers to other aspects in context with audio codecs, however, with most codecs this delay is determined as outlined in the following: According to the actual coder structure, the majority of audio codecs convert an audio block from the time domain to the frequency domain. However, a serious shortcoming of the block transform decoding scheme is the introduction of blocking artifacts in the reconstructed signal. Because the blocks are processed independently, quantization errors will produce discontinuities in the signal at the block boundaries. In order to prevent this effect, a principle has been investigated by [107], in which a window of samples from two consecutive blocks undergoes the transformation. The window is then shifted by samples and the next set of transform coefficients is computed. Thus, each window overlaps the last samples of the previous

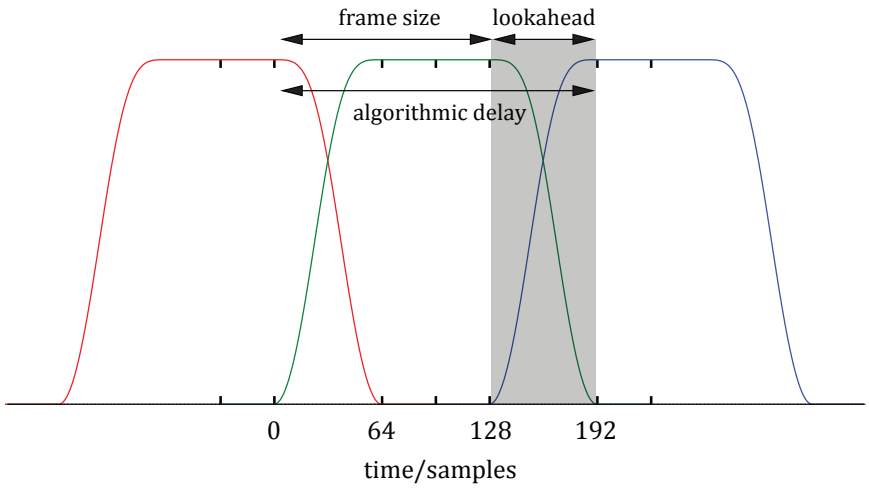


Figure 4.9: Example of a coder working with a frame size of 128 samples and a lookahead of 64 samples [150]

window. The overlap ensures the continuity of the reconstructed samples despite the alteration of transform coefficients due to quantization. The corresponding calculations take a certain amount of time, which represents a part of the computational delay. The drawback of the described overlap processing principle is an increased latency of n overlapping samples, which corresponds to the term "lookahead". In fact, a number of transformation principles in context with various window types exist as stated in [134] and [140]. As an example, figure 4.9 shows a transformation as described in [150]. The audio block size equals 128 samples and the lookahead consists of 64 samples.

If – with respect to figure 4.9 – a distributed system was configured with a sample rate of 48 kHz and a block size of 128 samples, the only additional coder delay would be introduced by the lookahead of 64 samples, which would result in 1.35 ms. Nevertheless, – as mentioned previously – audio codecs are generally designed for larger block sizes, which involves a larger lookahead respectively. Furthermore, one would not want to configure a low delay music system depending on compression codecs with large block sizes such as 1024 samples. Hence, the described unproblematic relation between sound processing and codec blocking is only valid for respective small audio block sizes. In turn, an ideal lossy audio codec exhibits a minimal block size in order to match the low delay restrictions of the distributed music domain. However, since most codecs are either designed for the purpose of offline audio encoding or for VoIP applications, block sizes below 1024 samples are a rather uncommon number. Nevertheless, two compression codecs exist, which were designed for low audio block sizes:

The CELT codec – developed by Jean-Marc Valin – is distributed under an open-source license. CELT stands for "Constrained Energy Lapped Transform". It features full audio bandwidth of 44.1 kHz and 48 kHz at a constant bit rate from 32 kbps to 512 kbps and is meant to support applications where both high quality audio and low delay are desired. CELT is able to work with audio blocks of 256 samples and a lookahead of 128 samples, which in total results in 384 samples and the corresponding algorithmic delay [150]. However, the lowest adjustable delay can be achieved with a block configuration of 64 samples with the respective lookahead of 32 samples. This leads to a total algorithmic delay of 96 samples corresponding to 2 ms of delay for a sample rate of 48 kHz. The subtraction of the blocking delay of 64 samples leads to an encoding delay of only 0.67 ms for the lookahead.

The ULD compression codec ("ultra low delay"), developed by Gerald Schuller at Fraunhofer IDMT Ilmenau (Institute for Digital Media Technologies), supports sampling frequencies from 32 kHz to 48 kHz and is designed to process audio blocksizes of 128 samples with an adjustable constant bit rate (CBR) down to 96 kbps [95]. Due to a lookahead of 128 samples the total algorithmic delay is made of 256 samples. After the subtraction of the blocking delay the additional ULD encoder delay results in 2.7 ms delay for a sample rate of 48 kHz.

In fact data compression implies an advantage in terms of the audio transmission time: With respect to the compression ratio a compressed audio block is proportionally faster transmitted by the involved network gear as described in the following subsection and illustrated in figure 4.11.

4.2.7 Signal transmission

Like analogue signals, digital signals can be transmitted on any kind of cable or radio wave but especially for long distance digital connections, fiber optic lines are used, which transmit the binary data via light impulses. Due to a reduction factor of 0.67 their maximum speed lies 10 % below the speed of copper [64]. In case two musicians were connected with a fiber optic link and if we deny the existence of the previously mentioned sound card delays or additional sources of delay, the propagation delay would only depend on the propagation speed of the fiber optic medium. For an EDAL of 25 ms this would allow a fiber length of 5025 km as calculated in equation 4.16. If a fiber optic cable was run straight across the european continent, it would lead to a maximal one-way delay of about 28 ms as illustrated in figure 4.10.

$$\frac{300,000 \text{ km}}{1,000 \text{ ms}} \cdot 25 \text{ ms} \cdot 0.67 = 5,025 \text{ km} \tag{4.16}$$

Apart from the fact that sound data can be transmitted via fiber optic or copper lines with their appropriate signal speeds of $0.67 \cdot c$ or $0.77 \cdot c$, a further transmission speed limitation has to be taken into account: Digital networks are equipped with

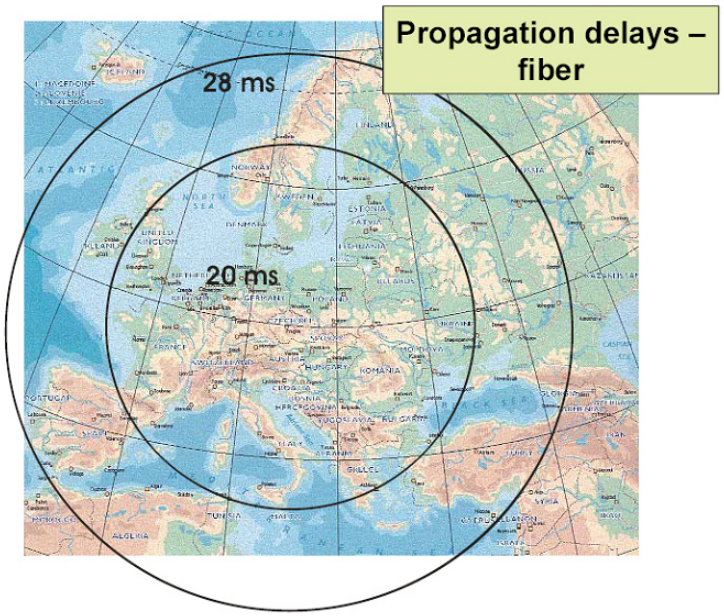


Figure 4.10: Propagation delay of fiber cables

network gear such as switches or routers [99], which basically guide each light impulse through a network in order to make it reach its correct and final destination. These network devices typically expect an input signal to be assigned to an output signal but in contradiction to the pure cable transmission this assignment suffers from further latencies: It is rather the capacity of a network device, which determines the delay for a signal to be mapped from one input port of a network device to an output port and this capacity is typically stated with the amount of bits per second (bps). The core of current networks consists of devices with capacities of several gigabits per second (Gbps). These so-called "backbone connections" [60] are able to serve several millions of users directly or indirectly by linking traffic to or from interconnected sub networks. Device and network capacities depend on the amount of users a network has to serve. The lower the amount of users, the lower the bandwidth capacity a network is generally administrated with. Nowadays current backbone links provide capacities from 10 Gbps to 5 Gbps down to 2.5 Gbps or 1 Gbps, in contradiction a user's endpoint connection is able to achieve about 20 Mbps down to 128 kbps or even 56 kbps in worst case of modem usage [147]. Within these two extremes further links typically range from 10, 34 or 100 Mbps up to 622 Mbps as common figures. In that context the term "wire speed" refers to this hypothetical maximum data transmission rate and in turn determines the theoretical best throughput and performance of a network device. We typically expect an optimized hardware device to operate at wire speed, rather than a software implementation, which might possibly suffer from CPU limitations.

With respect to the sound data the total required bandwidth b_{total} equals the product of the sample rate, the bitdepth and the number of channels, which equals 768 kbps for one audio stream with the standard settings of sample rate = 48 kHz, bitdepth = 16 Bit and a channel number of 1 as calculated in equation 4.17. This pure amount of audio data also refers to the term "payload". As a precondition for a successful data delivery any involved network component has to be able to handle at least this amount of payload data.

$$b_{\text{total}} = 48 \text{ kHz} \cdot 16 \text{ Bit} \cdot 1 = 768 \text{ kbps} \tag{4.17}$$

In terms of latency it is important to distinguish between the propagation delay and the transmission delay: As illustrated in figure 4.11 an audio block of 256 bytes (128 samples · 16 Bit) already requires 16 ms to be fully transmitted on a 128 kbps link – in contradiction to just 25.6 ns on a 10 Gbps backbone connection. This delay, however, does not take the previously introduced signal propagation on the link into account. Hence, when estimating an audio block's total network transfer latency d_{transfer}, it is firstly the sum M of propagation delays $d_{\text{propagation}}$ caused by the total conductor length of each specific propagation medium divided by the respective medium propagation speed and secondly the sum N of packet transmission delays $d_{\text{transmission}}$ caused by the involved network components [97]. This relationship is described by equation 4.18.

$$d_{\text{transfer}} = \sum^{M} d_{\text{propagation}} + \sum^{N} d_{\text{transmission}} \tag{4.18}$$

Considering an audio block decimation or compression (c.f. section 4.2.6) the stated transmission delay values can be reduced by a fixed factor according to the actual compression rate. Figure 4.11 additionally illustrates the transmission delays of a 32 bytes packet, which corresponds to this decimation or compression factor of 8 compared to the original 256 bytes packet.

4.2.8 Decoding (optional)

If a sender has sent an encoded audio block the receiver must decode it in order to retrieve a usable audio signal. In that context the decoder must know the compression scheme the the received block was encoded with. E.g. if the encoder transformed the audio to the frequency domain the decoder has to transform it back to the time domain in order to reconstruct an audible signal. Alternatively – if the encoder has decimated the signal – the decoder has to apply an upsampling by artificially inserting sample values into the received audio block. In fact, numerous encoding schemes exist, however, the decoding latency is only determined by the computational delay

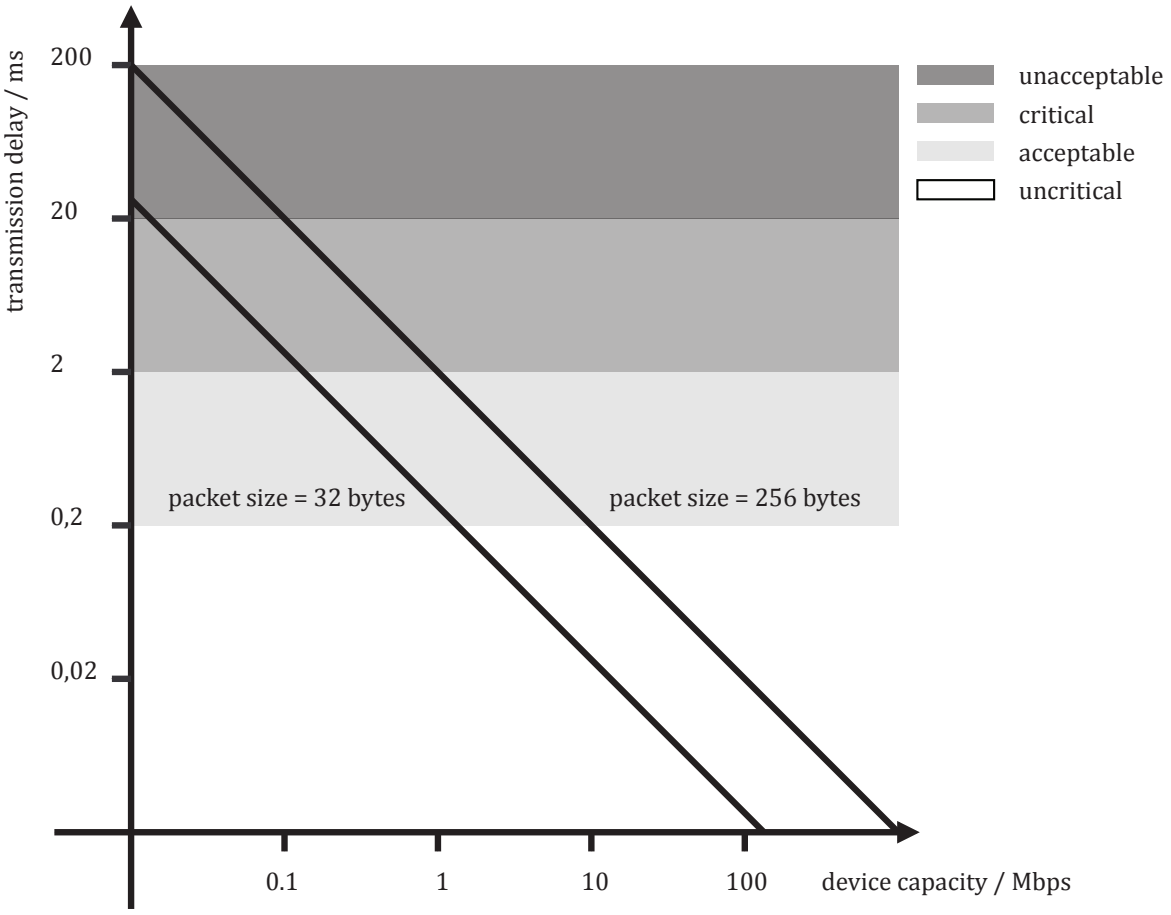


Figure 4.11: Network device transmission latencies

the decoder requires to complete the process. Hence – unlike the encoding – the decoding is not associated with additional algorithmic delays as shown in equation 4.19. However, according to the actual codec, the final computational decoder delay might still introduce latencies beyond 1 ms, which is why the author marks the decoding stage as well with a slight delay.

$$d_{\text{decoding}} = d_{\text{decodeComp}} \tag{4.19}$$

4.2.9 Device output blocking and driver buffering

Once a machine's network interface notices an incoming data packet of a remote sender, this packet will be read and – in order to provide the playback of the local and the remote signal – mixed with the current locally captured audio block. In this mixing process each sample value is summed up and the result is divided by 2, in order to avoid signal clipping due to saturation. Afterwards the resulting block contains both signals, the locally captured block and the received sound block [35]. Equally to the problem described at the audio input stage data cannot be processed sample by sample and hence the sound card's output architecture is as well designed as a double buffer system. While the second playout buffer offers its samples to the physical output, the application buffer holds the samples previously captured by the system's physical input, which also introduces a delay of one audio block. As a result the blocking delay of a sound card appears twice – first by the input double buffer and secondly by the output double buffer [46]. The total audio capture and playout process is illustrated in figure 4.12, while figure 4.13 shows the total sound card delay: Without taking potential network delays into account, the illustration marks the acceptable sound card configurations. In that context the author used an EDAL of 25 ms as a guide value. Configurations, which result in a total delay below 25 ms, are illustrated on a white, close to 25 ms delay on a light gray and beyond 25 ms on a dark grey background. As the user is supposed to figure the lowest possible latency a sound system can achieve, this graph helps to verify the system's ability to suit the requirements of distributed music performances. However, equal to the capture process, users must again be aware of additional driver delays possibly caused by the applied interface technology and operating system and have to add them to the graph's delay values respectively.

4.2.10 Digital output filtering

Besides the application of a digital input filter right after the sampling process, it is also usual and necessary to apply digital output filtering right before the signal transformation back to the time continuous representation [151]. The general principles equal the input filter section and the corresponding delays also depend on the required filter input samples and sample rate [154]. Again the applied technology is

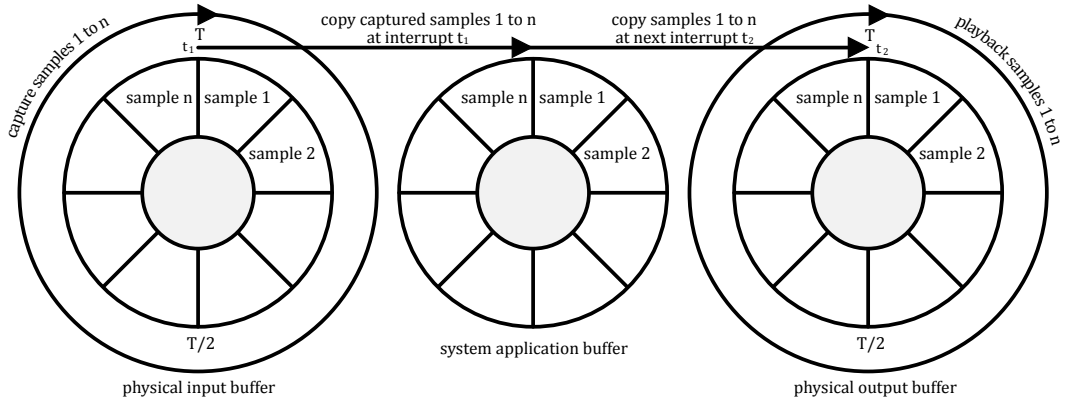


Figure 4.12: Audio playout process

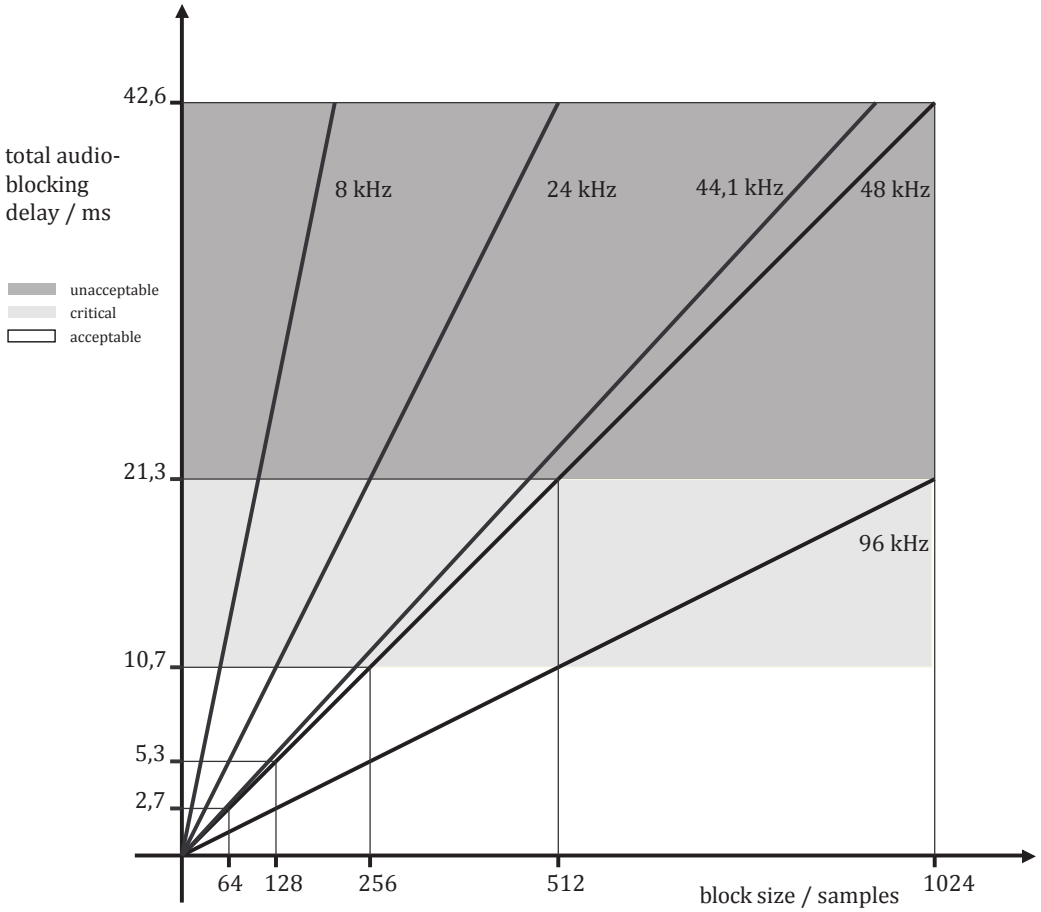


Figure 4.13: Total audio latencies

vendor and device dependent, however, the general purpose of the output filtering is an oversampling and low-pass filtering. It is applied before the final reconstruction process, in which the signal is again filtered with an analogue low-pass in order to eliminate the periodic spectrum repetitions from the signal (c.f. section 4.2.11). The output oversampling in realized by an interpolation and this increases the distance between the spectrum repetitions. Furthermore, a digital low-pass is applied before the signal is fed to the final reconstruction stage. As a result the analogue reconstruction low-pass can – despite a weak slew rate – without any difficulty extract the desired spectrum from the signal. Moreover, most D/A-converters treat the output signal with a "noise shaping" [119] in order to move the quantization noise (c.f. section 4.2.3) outside the audio band [119].

In terms of latency, as an example the RME Multiface sound card's output filter has a length of 30 samples, which corresponds to a delay of 0.62 ms at a sample rate of 48 kHz – 0.31 ms at 96 kHz respectively. Hence, the total input and output filter delay of the Multiface equals 1.27 ms at 48 kHz and 0.63 ms at 96 kHz respectively [31].

4.2.11 Reconstruction

After the signal has reached the final sound card's playout buffer, it has to be transformed back to a value continuous signal by assigning an electric voltage to each binary value. In that context each sample value is assigned a specific voltage, which is held in time at the current value until the next input number is latched [151]. This result already represents an audible time continuous signal but due to the stair steps characteristics, the original shape is distorted and in turn its spectrum contains disruptive higher frequencies. In order to get rid of these distortions, the stair steps characteristics needs to be smoothened and brought back to its original round shape. This procedure is represented by the Whittaker-Shannon interpolation formula 4.20 [102].

$$x(t) = \sum_{n=-\infty}^{\infty} x[n] \cdot \mathrm{sinc}\left(\frac{t - nT}{T}\right) \tag{4.20}$$

This so-called reconstruction is realized by filtering the current analog signal voltage x[n] with an analog time continuous sinc signal, which is illustrated in the upper left part of figure 4.14. In time domain both signals are convolved so that the zero-crossings of the sinc function occur at the sampling instants. Altogether the sum of the shifted and scaled functions finally recovers the original signal by the principle of interpolation. This principle is illustrated in the lower left part of 4.14.

This process can be described more descriptively in the frequency domain: The digitized signal shows its shifted subsequent spectra and the sinc is represented by a

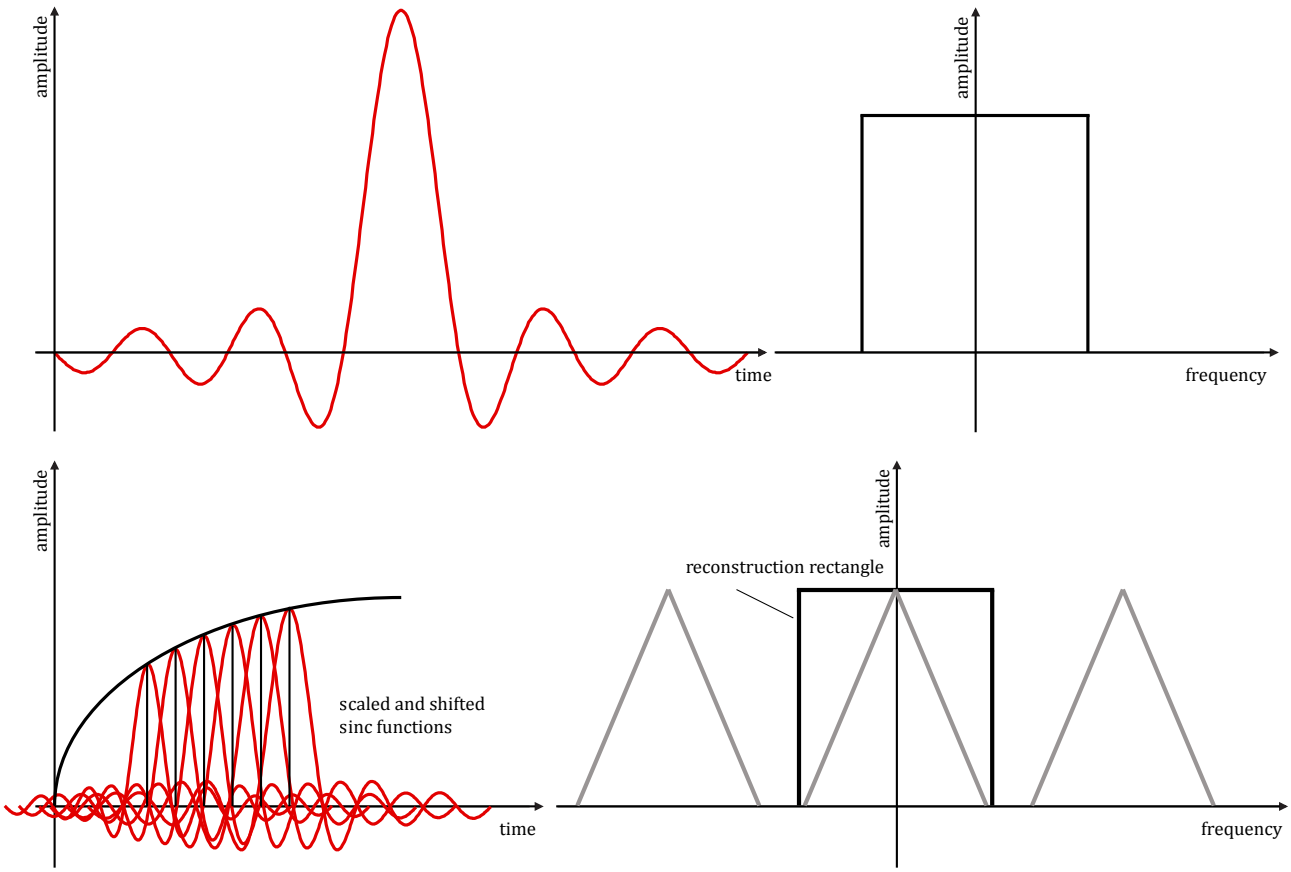


Figure 4.14: The sinc signal in time domain (upper left) and in frequency domain (upper right) and the reconstruction process in time domain (lower left) and in frequency domain (lowe right)

rectangle as illustrated in upper right of figure 4.14. Hence, according to the basic principles of digital signal processing the convolution of the audio and the sinc signal in time domain corresponds to a multiplication of the audio signal's spectrum with a rectangle in the frequency domain [151]. Since it is only the center spectrum and not any of its higher frequent repetitions, which matter in order to retrieve the original signal, the rect signal must suit the respective frequency range. After the multiplication frequencies above and below the desired center spectrum will be suppressed to zero and in turn the signal is successfully reconstructed. This principle is illustrated in the lower right part of figure 4.14. If the subsequent spectra overlapped due to aliasing the rectangle would include lower frequencies of the next higher spectrum and higher frequencies of the previous lower spectrum. As this results in undesired disturbing distortions, the previous stages apply the anti-aliasing filter and an appropriate sampling frequency with values beyond twice the expected frequency plus possibly an artificial oversampling [119]. Nevertheless, the analog reconstruction filter must also fulfill a certain slew rate in order to filter the according desired frequency parts and would otherwise equally introduce aliasing. Altogether it is the combination of the analogue filter's quality (anti-aliasing filter and reconstruction filter), sample rate and digital filtering (interpolation and noise shaping) which determines the final signal quality. Like the anti-aliasing filter also the reconstruction filter introduces a delay of 8 μs for a cutoff frequency of 20 kHz.

4.2.12 Sound card synchronization and wordclock jitter

This section so far described the sound signal path from a sender to a receiver in the digital domain. However, this transmission suffers from a problem related to the physics of digital sound cards: Due to slight inaccuracies of a sound card's quartz frequency, any adjusted sample rate practically underlies a slight deviation above or below this theoretical value. In turn this means that two sound cards normally exhibit slight sample rate differences. Considering a bidirectional audio link between these devices, one sound card would play back a received stream slightly faster with a respective higher pitch than originally intended, while the other card would play back the received stream slightly slower with a respective lower pitch than originally intended. In fact, this effect is hardly noticeable and might be considered as negligible, however, in the professional audio domain, it is mandatory to prevent this effect. In that context an external wordclock signal is used for the synchronization of different digital audio devices as a time reference. It is a rectangular signal of a fixed frequency of e.g. 48 kHz (cycle duration T = 20,83 μs). The high-level amplitude usually ranges between 2 and 5 V and the low-level voltage at 0 V. The positive edges trigger all digital processes, particularly the previously introduced A/D and D/A conversions. By feeding this central wordclock time reference signal to each of the involved sound card's wordclock input, a precise synchronization of all devices can be achieved. This common principle is applied in any professional sound studio, where device interoperability is required. Nevertheless, this principle is limited to locally interconnected

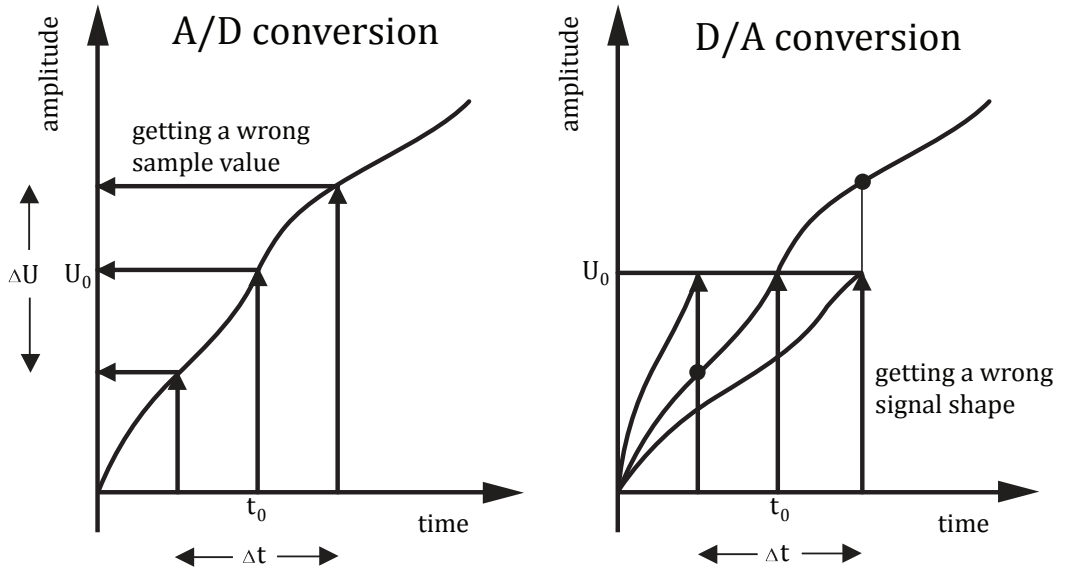


Figure 4.15: The effect of wordclock jitter (left: A/D conversion, right: D/A conversion

devices within reach for a direct cable connection with the reference clock. However, in chapter 7 the author will introduce an new sound card synchronization approach in order to overcome this restriction.

Moreover, the phenomenon of wordclock jitter must be introduced in that context: Wordclock jitter describes the deviation of the real trigger events from their theoretic dedicated positions. This results in problems related to the audio signal conversions: Time deviations of trigger events can cause wrong quantization values in the A/D conversion process or a wrong output voltage in the D/A conversion process. Practically, each conversion process, triggered by a jittery wordclock signal, modifies the audio material and every influence is added one after another to the signal. Furthermore, distortions induced during A/D conversions are irrevocably imprinted into the signal [80]. Both effects are illustrated in figure 4.15.

A measurement by [30] shows in the upper part of figure 4.16 the wordclock jitter of a MADI wordclock [142] in dimensions of up to 80 ns, which at a period duration of 20.83 μs corresponds to a maximum variation of less than 1 %. In that context it is debatable if this has a significant impact on the perceived audible impression. Research carried out by Florian Hernschier in [80] shows that wordclock jitter can be measured and clearly visualized. The values vary significantly from device to device. Bringing the results of numerous measurements and listening test together, it turned out that there seems to be no clear relation between the measurement results and the evaluation of the listening test: Having a low jitter wordclock signal is no guar-

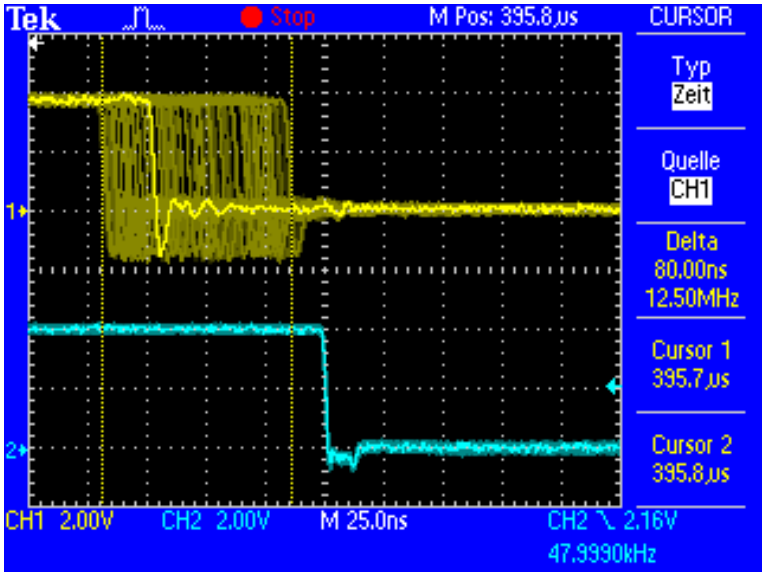


Figure 4.16: Comparison of two sound signals regarding the amount of wordclock jitter [30]

anty for positive sound attributes. Wordclock signals with good measurement results are inconspicuous concerning the listening test as they were less often implicated to neither good nor bad sound characteristics [80]. Nevertheless, sound card vendors generally make an effort to reduce sound card jitter as much as possible. As an example the upper part of figure 4.16 shows a signal with strong wordclock jitter of 80 ns in comparison to wordclock jitter of less than 1 ns after synchronization with the steady clock technology of RME in the lower part of figure 4.16 [20].

4.3 Transmission systems

Network technology is typically more than a simple cable connection between two devices. Apart from the pure delivery of electronic data on the cable, there is further complex logic behind a functioning network. In order to explain these logical structures the author is using the ISO/OSI protocol stack [147], which is a very general and widely accepted network model. It is illustrated in figure 4.17. The model is divided into seven layers, each one representing certain network functionalities. At the very bottom we see the physical layer that represents the direct link to the physical network media. On this layer we face hardware components and physical signals. The data-link layer represents technologies for concurrently accessing the network media. It determines fundamental parameters how and in what sizes data is transmitted on an underlying physical medium and can as well define mechanisms for error detection. The network layer is responsible for addressing packets (host-to-host) and

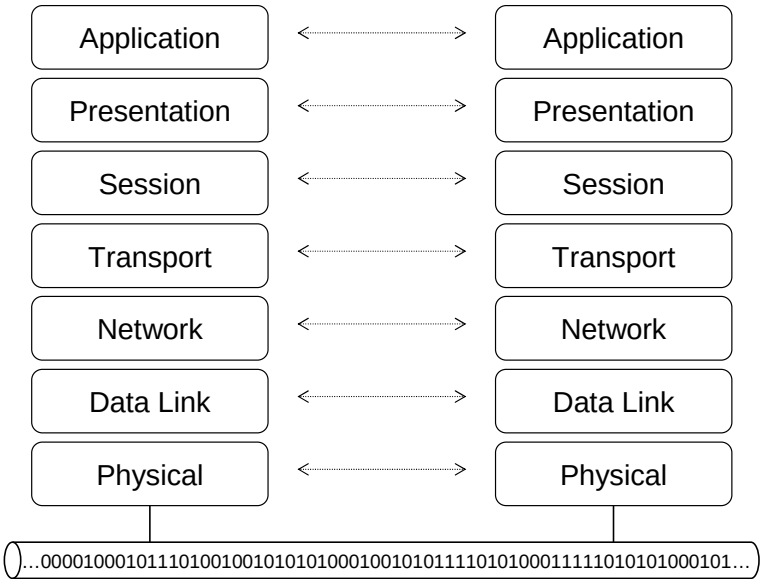


Figure 4.17: ISO/OSI Stack

for directing packets through the network, while the transport layer adds more fine-grained addressing functionalities and communication rules between two processes. Furthermore, the session layer is needed for establishing a stateful communication between two processes. The presentation layer is responsible for adapting varying data representations between the communication parties and finally the application layer is providing an interface for the actual network application. In figure 4.17 we see two instances of this protocol stack, representing two communicating hosts in the network. Through this protocol stack both host are able to interchange data: Each layer can virtually communicate with its remote counterpart (horizontal arrows), but physically the data is always sent via the underlying layers, finally traveling bitwise through the physical media. After the remote side has received the data on the physical layer it is passed upwards again [95].

In the following the author will describe the ISO/OSI stack with respect to a distributed music system and its requirements: The fist step before considering a music session a user would need to set the relevant audio parameters such as sample rate, block size, channel numbers and buffer sizes. According to these settings an application has to record blocks of sound card data and prepare them for the transmission. The corresponding layers are the application, presentation and session layer. Since the strict delay boundaries require the fastest possible way of message transport, the transport layer should transmit data as direct and quick as possible, while the network layer should address it accordingly. Possible protocol choices for the Transport and network layer will be discussed later in context with current commonly available network technologies. Below the network layer, a network can consists of various architectures for the data-link and the physical layer. Typically both layers form an

entity so that a specific data-link layer technology is generally bound to a specific cable and connector defined for the physical layer. As the applied technology in the lowest two layers determines the basic transmission characteristics and rules in a network of interconnected devices, it has the most fundamental significance concerning the fulfillment of the low delay restrictions of a distributed music system. As all layers have a hierarchical dependency a higher layer cannot be able to speed up possible delays in a lower layer and hence it is the physical and the data-ink layer, which characterize the basic transmission behavior of a network [147]. A pure and single cable connection between two devices represents this most direct and fastest case of device interconnection as already described in the context with analogue networks in one of the previous sections. However, the advantage of this simple and clear connection implies significant drawbacks: With an increasing number of network hosts a network's complexity rises which in turn complicates the administration of n explicitly used lines and increases the probability of a line defect. Furthermore the explicit use of a single line can be considered as inefficient, since theoretically a transport medium can be used for the propagation of more than a single signal at the same time. The principle behind this is called multiplexing [116], which will be covered in the following.

Whenever a sender or a receiver consists of more than one entity, data has to be multiplexed. Generally four kinds of multiplexing exist – space division multiplexing (SDM) [152], frequency division multiplexing (FDM) [156] or the corresponding wavelength division multiplexing (WDM) [149] and time division multiplexing (TDM) [116]. The simplest approach is the space division multiplexing which collects n physical transmission channels and bundles them to a single main line so that signals processing happens in a parallel manner via space separated paths: In the domain of audio engineering the multicore cable collects n cables into one single main cable or analogue multitrack devices write unmodified n audio tracks onto one tape [152]. In telecommunication the so-called crossbar switch in telephone switching centers automatically switches a digital signal stream to the right destination and in turn establishes an exclusive link between two sides. SDM provides better ease of use but has no advantage in terms of line efficiency. By FDM a connection is separated into different frequency bands, which are finally used as separate transmission channels for numerous signals. Each signal in modulated according to the desired frequency band and decoded at the receiving end. This allows simultaneous transmission on a single cable. The most famous applied field of FDM is the transmission of radio waves though the air, where each radio program has its own carrier frequency. As numerous destinations might want to receive different radio programs, each radio receiver can be tuned according to the desired carrier frequency. Similarly WDM follows the same approach but appears in context with fiberoptic lines: Rather than frequency modulation it uses different wavelengths (or colors) as carrier for the respective information channels on a fiber. Finally, the major multiplexing technology in terms of digital communication is the time division multiplexing (TDM), which transmits a number of signals in a time shifted and interleaved manner on the same transport

medium [116]. TDM can be broken down into the synchronous (STDM) and the asynchronous (ATDM) mode. For STDM the time domain is divided into several recurrent timeslots of fixeded length – one for each virtual channel – the respective data block or STDM frame is transmitted in. After the last channel has sent the data in its respective timeslot the cycle starts again with a new frame, starting with the second data block of the first channel. Due to this synchronous clockbased approach STDM also refers to the term "static multiplexing". In contradiction to STDM where network clients are served in their respective timeslot even if no transmitting data is available [116] an ATDM network does not carry any data in case of a transmission break or interruption. This asynchronous and non-clock based approach refers to the term "statistical multiplexing". Based on the the appropriate multiplexing techniques current networks can be divided into three categories – asynchronous, synchronous and isochronous networks – as outlined in the following subsections.

4.3.1 Asychronous networks

Asynchronous networks are based on ATDM, which implies that signal transmission happens randomly depending on stochastic host transmission activities. Furthermore, any network host is allowed to send a desired amount of data in a blockwise restricted manner, that is, if a sequence of bytes exceeds a certain upper size limit, this sequence will be split into chunks of that maximal size. Such chunks of data are also considered as packets, blocks or frames and their maximal size is called the maximum transfer unit (MTU) [147]. This uncomplicated approach has significant drawbacks: As senders can individually decide about the amount of packets and their size, they might possibly flood a network with too much data and would in turn exceed its capacity. This is especially the case if network administrators underprovide certain network segments, so that a link's capacity does not suffice to carry each endpoint's maximum data throughput at the same time. Secondly asynchronous networks suffer from an effect called "network jitter" [147], which describes a delay variation of the delivered data blocks. This effect occurs due to the fact that certain data blocks cannot be transmitted before a network component has completed the transmission of a previous data block. This results in additional waiting times, which increase with the amount of so-called "cross traffic" of additional network hosts. The less transmission capacity a network component has the higher the amount of network jitter. Especially in context with the small subsequent data blocks of low-delayed audio streams this leads to significant problems: The larger a cross traffic data packet is the longer it blocks the transmission medium. According to this waiting time the subsequent audio block will be delivered later [60]. However, as the playout buffer of a sound card expects data in constant time intervals depending on the actual settings (c.f. section 4.2.9), sudden increases of the network delay would violate this precondition of constant and errorfree audio stream playback. This leads to a so-called "drop out" in the received audio stream. Due to this problem asynchronous networks are mainly designed as non-real-time-communication systems, which do not make strong

demands on the network in terms of delay or jitter. The most famous example of an asynchronous network technology is the Ethernet [128], which is the most significant and currently most often applied local area network (LAN) technology: Ethernet typically works with a MTU of 1,500 bytes and its current transmission capacities range from a minimum of 10 Mbps to a maximum of 10 Gbps. Assuming a 10 Mbps Ethernet network component, a 1,500 byte packet would block the medium for 1.35 ms. With a 100 Mbps connection the transmission time for a 1,500 byte packet is 10 times lower at 0.13 ms respectively. The same asynchronous principle is valid for wireless local area networks (IEEE 802.11) with their current maximal transmission capacities up to 54 Mbps [125].

4.3.2 Synchronous networks

In a synchronous network hosts do not have such operational freedom as they have in an asynchronous network. Due to the clock driven approach of STDM [116] each host is served at its reserved fixed timeslot in terms of sending and receiving a fixed amount of data, which size depends on the given network specification. As synchronous networks are specialized, purpose-built systems, this specification depends on the field, which determines the requirements regarding the clock resolution, slot design and data sizes. E. g. in the synchronous telephone network, endpoint devices work with a sample rate of 8 kHz and in turn the network clock's frequency equals 8 kHz. As a sample rate of 8 kHz results in a sample generation every 125 μs, the total duration of a respective STDM frame matches this value in order to transmit the signal in a "sample by sample" manner – rather than packet by packet as in the asynchronous case. This principle is illustrated in figure 4.18 for a 32 channel STDM frame. It is the network's transmission capacity, which technically determines the number of channels and in turn the number of served hosts for one frame. The most famous example of a synchronous network architecture is the Synchronous Digital Hierarchy (SDH) [147], which is used in almost any telephone communication backbone all over the world and nowadays is mainly applied in combination with ISDN (Integrated services digital network) in the top layer [92]. The SDH transmission capacity depends on regional aspects of how many clients to be served and in that context telecommunication providers use so-called synchronous transport modules (STM): The available voice data transmission capacities range minimally from 48.960 Mbps for a STM-0 up to maximally 153.94 Gbps for a STM-102. Since a sample rate of 8 kHz and a resolution of 8 Bit results in 64 kbps for each phone connection, such transport modules can serve a corresponding minimal number of 765 up to a maximum number of 2,405,376 hosts with a guaranteed calculated throughput. With respect to the example frame in figure 4.18 the total frame size would result in $32 \cdot 8$ Bit = 256 Bit.

The most significant advantage of the synchronous approach is that due the fixed time slots and equal data sizes the problem of network jitter does not occur: Each transmission has the same duration and is processed in constant time intervals [116].

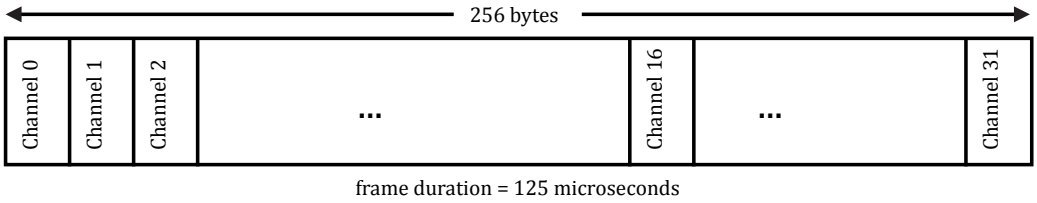


Figure 4.18: Principle of a synchronous TDM frame with 8 kHz sampling rate and 8 Bit resolution

Altogether, the general principle of STDM makes synchronous network the predestinated choice in terms of realtime communication, unlike the asynchronous approach forms the basis of data networks. Anyway, synchronous networks are also supposed to provide asynchronous data delivery: Theoretically a 64 kbps telephone link can be used for the transmission of any kind of data and in this context a special service of the ISDN specification is the so-called channel bundling [141], which allows to combine two 64 kbps channels to one single 128 kbps link. However, since in a synchronous network data is constantly sent even if senders do not have any data available, it does not readily accommodate bursty asynchronous traffic patterns. The network is permanently under full traffic load, which in that context can be considered as inefficient. Hence, equally as carrying synchronous data on an asynchronous network, vice versa, carrying asynchronous data on a synchronous network is not an ideal solution in terms of a mixed services network [58]. Furthermore, synchronous networks lack certain flexibility as the static and hierarchical technical conventions such as the network clock, sample rate and bit resolution do not allow to apply any individual modification of the technical setup if it was required.

Plesiochronous networks

The translation of the greek work “plesiochronous” says “almost synchronous” and according to this term a plesiochronous network works like a synchronous network with a less stable clock behavior [74]. This instability occurs due to the use of more than one central clock as it is the case in a synchronous network. Practically any clock’s frequency can be precisely calibrated but as numerous clocks are not directly linked to each other it is likely that sudden speed drifts occur. The PDH (Plesiochronous Digital Hierarchy) [74] as the predecessor of SDH worked with the same STDM principle but as each multiplexing unit had its own clock, it was likely that the network as a whole was not in perfect sync. This drawback was eliminated later with SDH by using one central clock.

4.3.3 Isochronous networks

An isochronous network can be described as a compromise between a synchronous and an asynchronous network as it features aspects of both of them. Identically with an asynchronous network the network link is idle as long as no data is actually transmitted. Furthermore, hosts are not restricted to certain time slots and fixeded data sizes so that an isochronous network can identically be used like an asynchronous data network. However, in order to reduce the effect of network jitter and in turn equally provide decent conditions for synchronous realtime traffic it introduces the concept of cells: Any packet of data is split into a number of equally small sized cells, which require significantly less time to be transmitted [147]. Cells are generally not broken down to a single byte or sample as it can be the case in a synchronous networks, but sizes between 8 bytes and 100 bytes are common and can be considered as extremely small – especially if the corresponding full packet size might consist of 1,500 bytes or possibly more. Furthermore, an isochronous communication system is intended to deliver quantifiable performance, which is realized in a service agreement on the connection between communicating network nodes [60]. This service agreement specifies criteria such as bandwidth, delivery delay and delay variation on a special protocol sublayer. As a result isochronous transports are capable of carrying a wide variety of traffic and thus represent a versatile and flexible networking concept. One example is the USB (universal serial bus) [62] as a local device connector. However, in terms of a wide area network with numerous hosts the best known isochronous concept is the asynchronous transfer mode [58] (ATM): ATM splits the data into cells of 53 bytes, which require 28 times less transmission time than full Ethernet frames of 1,500 bytes. At the beginning of the 1990s ATM used to be declared as the major and most advanced network technology due to reasons outlined in the next section. Altogether the idea of compromising between synchronous reliability and asynchronous flexibility is smart but also implies the drawback of larger complexity than is the case with asynchronous network components such as Ethernet [60].

4.4 The Internet

Until the late 1960s wide area networks had significance only in terms of telecommunication mainly for voice based services [76]. On top of that providers offered additional services such as Telex for text based communication or Telefax as a remote copy service. These services have been used and improved constantly over the years. Asynchronous data networks played no significant role in that context as they were mainly intended for data retrieval in local area networks of companies. Hence, in the past we could clearly distinguish between synchronous telecommunication networks and asynchronous industrial data networks. Although in the first decade of the 21st century this separation still exists, it is not as strict as it used to be: McLuhan states in [105], that presumably every single medium will be combined and finally

integrated into a single multi purpose medium. In that context customers more and more asked for multimedia and communication services for audio and video on one hand and at the same time for services in data retrieval and non-realtime communication on the other hand. In order to be prepared for the new mixture of real time and data services, telephone providers consequently had high hopes in a new broadband ISDN (B-ISDN) [141], which preferably applied the isochronous ATM in its data-link layer. Things, however, did not develop as anticipated: In the meantime people became more and more attracted to a network, which was originally intended as an information pool for military data exchange on an asynchronous wide area basis. This network – originally called the Arpanet [76] – later became the today's well known Internet. Rather than military data, the Internet today offers an enormous amount of data hosted on globally interconnected machines and can already be considered as the major commonly available source of information. As this ranges from e.g. pure scientific data to online newspapers or from stock data to pure entertainment or arts, the Internet nowadays is able to offer almost any kind of desired information. Besides using websites [147] on the world wide web (www) [147] as a pull data source, the Internet as well offers persons the possibility to push one's own content onto the network. Furthermore, the email [135] service has become a globally established non-realtime communication service in terms of fast, flexible and cheap message interchange. Apart from that further services such as online banking, flight reservation, partner search or chat systems became a common and integral part of modern life. In that context the modern term Web 2.0 [24] is supposed to describe and cover the new sociocultural aspects and future possibilities the Internet can offer.

Since the Internet has such an impact on our daily life and since modern personal computers can already provide high quality multimedia content, the idea of a network that covers any desired service, has become attractive to users and the industry as well. Hence, despite the drawbacks of asynchrony, in 1992 first services offering Internet radio came to life, followed by Internet telephony (VoIP) [38] and videoconferencing [87]. These services have been improved constantly and almost 20 years later they have become extremely reliable and a generally accepted form of realtime communication [89]. The famous VoIP tool "Skype" [10] represents a good example for this tendency. The common approach to compensate the effect of network jitter is the application of a jitter buffer at a receiver's end: By storing up an amount of audio packets in the network queue, the audio process can still provide a solid playback in case of late arriving packets. The drawback of this principle is a higher latency as packets are not processed right after the reception [94]. Another aspect that made the Internet become famous and successful is its availability in form of affordable flatrates. A flatrate offers users a constant 24 hour service at a fixed price per month and in turn allows as much network activity as desired. This is especially attractive in case of VoIP Internet telephony where users do not have to pay an extra bill for their telephone calls anymore. There is no doubt that the Internet as an asynchronous medium is not the predestinated choice for the carriage of realtime data and especially not for low-delayed audiostreams with their according low block sizes.

However, due to its large distribution and its strong sociocultural establishment it is already considered as the main multi-purpose future network – in contradiction to the synchronous telephone network, which is rather considered as outdated. Hence, further investigations in distributed music systems will use the Internet as the prime transmission system of choice. In the following the author will describe its basic fundamental principles with special regard to aspects of latency and jitter.

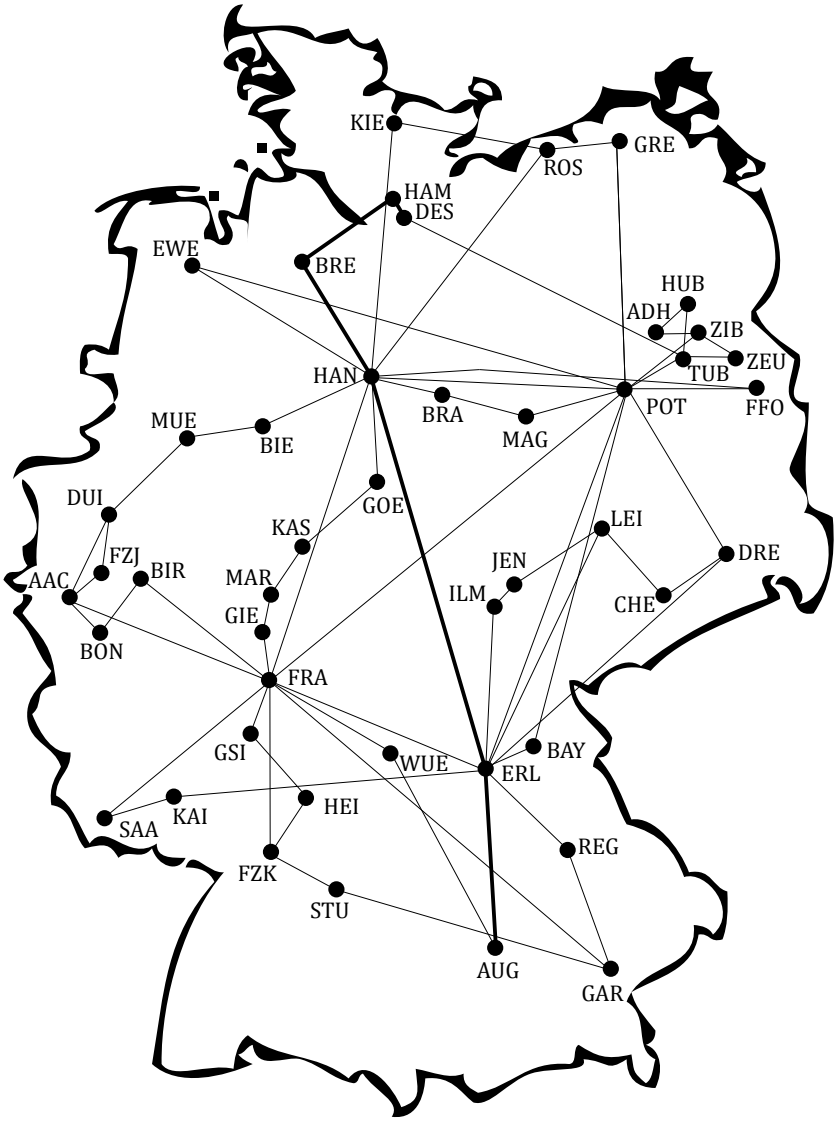
With respect to the previously introduced ISO/OSI model it is actually the network layer, which identifies network traffic as Internet traffic by using the so-called Internet protocol (IP) [99]. Whenever data is encapsulated in an IP packet, it is clear that a sender uses the Internet and its respective addressing conventions, which includes an IP address and a port [147]. Hence, any network technology can be applied to the physical and the data-link layer if the IP protocol is used in the network layer. However, in order to fully describe and address a valid IP data packet, additional layers above the network layer have to be taken in to account. In that context the "Internet protocol suite" has been introduced as an Internet-specific stack model [70]. In contradiction to the ISO/OSI model it encapsulates the application layer, the presentation layer and the session layer to one single application layer [70] and defines a set of Internet-specific protocols. On the transport layer the Internet protocol stack offers the choice between either the Transmission Control Protocol (TCP) [99] or the User Datagram Protocol (UDP) [36]. The former works with network acknowledgements in order to confirm the arrival of sent data packets at the destination. While this principle on one hand improves the reliability of the message transport, it on the other hand introduces higher delay times. In comparison, UDP simply sends the data from the sender to the receiver without guaranteeing reliable transmission and in turn provides the quickest signal delivery [36]. Hence it is the predestinated choice for the transmission of audio data. The highest application layer generally provides services of higher complexity based on the underlying UDP or TCP protocol. An example of such a service is the hypertext transmission protocol (HTTP) [147] or the file transfer protocol (FTP) [147] which are both based on TCP and applied in context with webbrowsing and file retrieval. As for the generation of low-delayed audio data no special protocol exists, a custom solution on the application layer is required, able to generate audio blocks corresponding to the actual user settings such as sample rate, block size, sample resolution and channel numbers.

As in any network, also in an IP network hosts can exchange data with other hosts on the same network but not with hosts in foreign networks unless special packet forwarding schemes are applied. Since the Internet in fact consists of numerous interconnected IP networks, these networks require an instance, which is able to provide the interchange of packets between them. Such instances are called gateways or routers [128]. They forward an appropriate data packet from a specific gateway of the source network to the gateway of the destination network which finally routes it to the desired endpoint. Routing is a complex process, which happens either statically by predefined paths – set manually by network administrators – or dynamically by

applying special routing algorithms [128]. The latter establish network paths according aspects such as link utilization or special service level agreements (SLA) [147]. Since a path through the Internet depends on the physical location of available networks with their respective gateways, a direct link between two hosts is unlikely to expect. Instead the signal's route typically shows detours as it always has to pass specific network nodes as parts of their route. A good example is the routing of traffic in the german research backbone DFN (deutsches Forschungsnetz): As illustrated in figure 4.19 almost any german university or research institute represents a DFN node [14]. Some nodes share a direct link, as node FRA (Frankfurt) does with node POT (Potsdam) and in this case the delay between the nodes (or also called hops) is introduced by one single connection. However, if we look at hop DES (Deutsches Elektronen-Synchrotron Hamburg) in the north of Germany and hop AUG (Augsburg) in the south, packets would need to travel via hops HAM (Hamburg), BRE (Bremen), HAN (Hanover) and ERL (Erlangen) with their corresponding route segment and detour delays or via other possible hops combinations depending on the currently applied packet routing scheme.

As the DFN is part of the global Internet and the european research backbone GEANT2 [5] in particular there is one hop in the network, which is designed especially for the routing of international traffic. This gateway with the appropriate high bandwidth network gear is located in the city of Frankfurt. Hence, international traffic to or from any DFN hop will always be routed via Frankfurt, which has a drawback in terms of latency: Two hops, which do not belong to the same country but share a physically close location close to their country's border, typically don't share a close network route: Figure 4.20 shows the GEANT2 network topology with its respective gateways. A descriptive example for the routing delay problem is the present in the case with the city of Lübeck in the north of Germany and the city of Copenhagen/Denmark. Both locations are separated by a physical distance of approximately 400 km in case the route would go straight north. However, due to the GEANT2 routing, the practical route involves a larger number of hops and results in a length of about 1,400 km as packets move south to Frankfurt first and then up north to Sweden and finally south-west to Copenhagen. An even more descriptive example of a routing detour is the path between Turkey and Israel as depicted figure 4.20.

Technically routing data – in our case one block of audio data – assumes a valid packet structure a router can identify and forward it according to its routing table. This in turn requires a certain data structure which is termed out as header [60], which contains metadata about the protocols assigned to to the packet. Upon reception such header must be removed from the packet in order to extract the packet audio data. In fact, the process of adding and removing a packet header implies a certain number of CPU cycles and hence introduces a delay. However, with the current machine clock speeds of several gigahertz, this latency value significantly falls below 1 μs, and in turn the author considers it as insignificant. The first header which is added to the



AAC	Aachen
ADH	Berlin
AUG	Augsburg
BAY	Bayreuth
BIE	Bielefeld
BIR	Köln
BON	Bonn
BRA	Braunschweig
BRE	Bremen
CHE	Chemnitz
DES	Hamburg Desy
DRE	Dresden
DUI	Essen
ERL	Erlangen
EWE	Oldenburg
FFO	Frankfurt/Oder
FRA	Frankfurt/Main
FZJ	Juelich
FZK	Karlsruhe
GAR	Garching
GIE	Giessen
GOE	Göttingen
GRE	Greifswald
GSI	Darmstadt
HAM	Hamburg
HAN	Hannover
HEI	Heidelberg
HUB	Berlin
ILM	Ilmenau
JEN	Jena
KAI	Kaiserslautern
KAS	Kassel
KIE	Kiel
LEI	Leipzig
MAG	Magdeburg
MAR	Marburg
MUE	Muenster
POT	Potsdam
REG	Regensburg
ROS	Rostock
SAA	Saarbrücken
STU	Stuttgart
TUB	TU Berlin
WUE	Würzburg
ZEU	Zeuthen
ZIB	ZIB Berlin

Figure 4.19: The X-Win research backbone with an example route between hops DES (Desy Hamburg) and AUG (Augsburg)

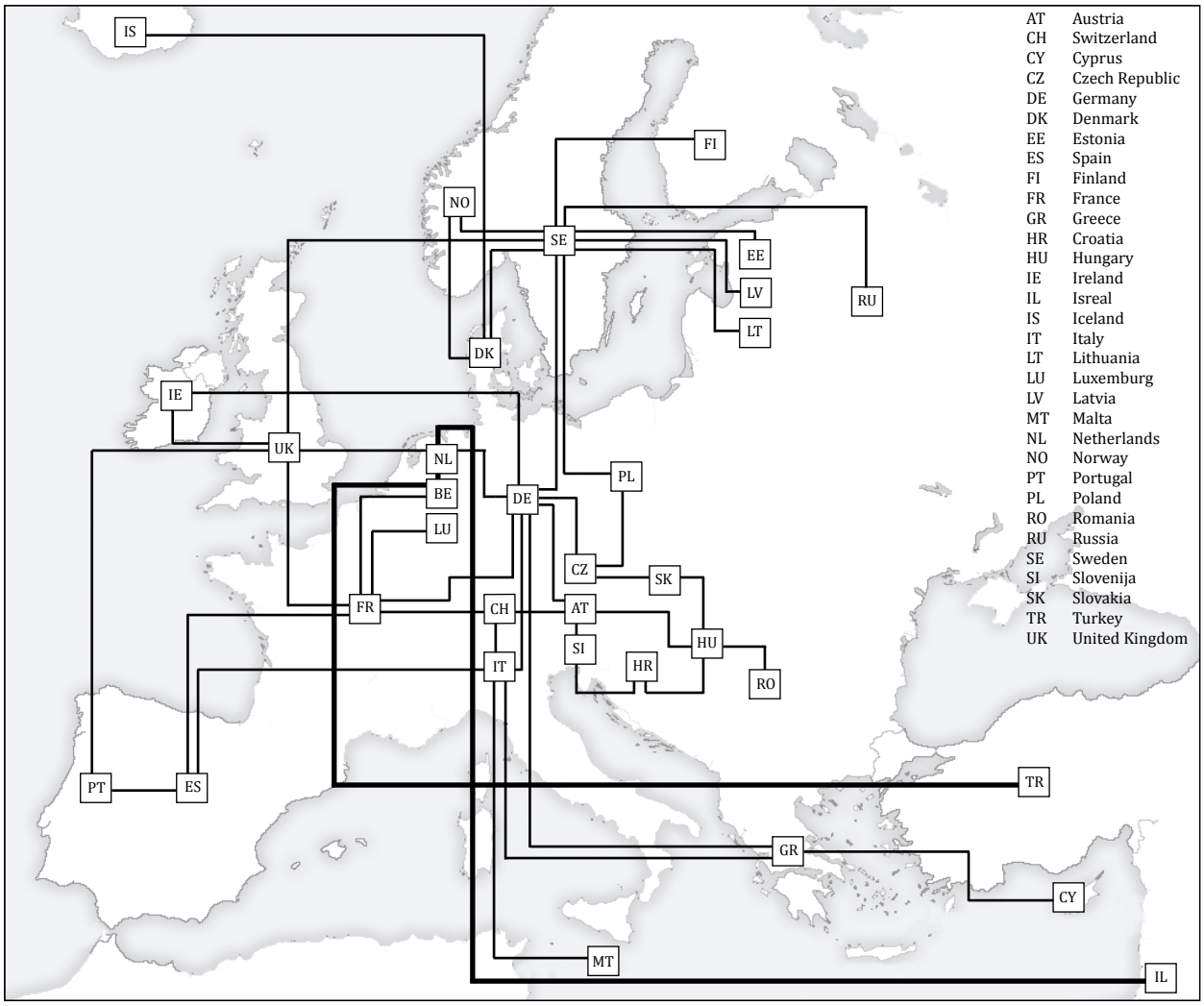


Figure 4.20: The european GEANT research backbone with an example routing detour between Turkey and Israel

audio payload, is a UDP header [60] – rather than a TCP header – as suggested on the transport layer in the Internet protocol stack. It has a fixed length of 8 bytes.

Since the Internet presumes the use of the IP protocol, the network layer offers no other choice than that. Otherwise a packet would not be considered as Internet traffic. Hence, an additional IP header [99] has to be added to the packet. It has a fixed length of 20 bytes. Finally, the resulting packet size of such a valid IP packet consists of 28 bytes on top of the adjusted payload size of minimal 128 bytes for a 64 sample-packet.

Although this data structure already describes a routable IP packet, it still lacks the physical information of how to actually reach a router's input port. This information is provided on the data-link layer and depends on the applied technology. Since IP networks define the network layer as their basic reference layer, one can freely choose, which technology to be applied on the lower two layers as long as the respective gateway is able to function with this technology. The status of the major data-link layer technology for the Internet was fought since the 1990s between competitors like Frame Relay, FDDI, Token Ring, ATM or Ethernet. For a few years it seemed that due to its isochronous character again ATM would suit the mixture of data and real time demands best but finally it turned out that ten years later the asynchronous Ethernet had won this competition [128]. Nowadays almost any PC is equipped with an Ethernet adapter and routers or switches are Ethernet-based, so that a machine's Ethernet card would add its appropriate header to an IP packet. This header has a fixed total length of 26 bytes [128].

After attaching the three headers to the packet, it is sent, routed and received by a remote destination. Upon reception the headers are removed in the reverse order in order to extract the audio payload. Finally, the received audio block is stored in the remote side's jitter buffer, which ends the Internet related signal path. As a summary each respective stage is illustrated in figure 4.21.

Apart from the audio payload, the valid routable packet consists of a UDP, an IP and an Ethernet header finally adding up 54 bytes of header information, which is termed out as "overhead" [147]. Since this overhead occurs for every sent packet, it is directly linked to the audio settings specified on the application layer: As a reduction of the block size or an increase of the sample rate increases the amount of generated audio buffers, in turn the number of sent packets with the corresponding overhead increases. A minimization of the audio blocksize of $1/x$ leads to a protocol overhead multiplied by x. This relationship is further outlined and illustrated in the following subsection. In fact some ATM switches might still be available. However, since most current networks – especially the majority of fiber optic backbone connections – are already Ethernet-based ATM must be considered as outdated [60]. The main reason why ATM lost this competition lies in increasing transmission bandwidths: At times, when link capacities ranged at a dimension of only a few hundred kbps, the jitter introduced by asynchronous IP data packet sizes was too high in order to support real

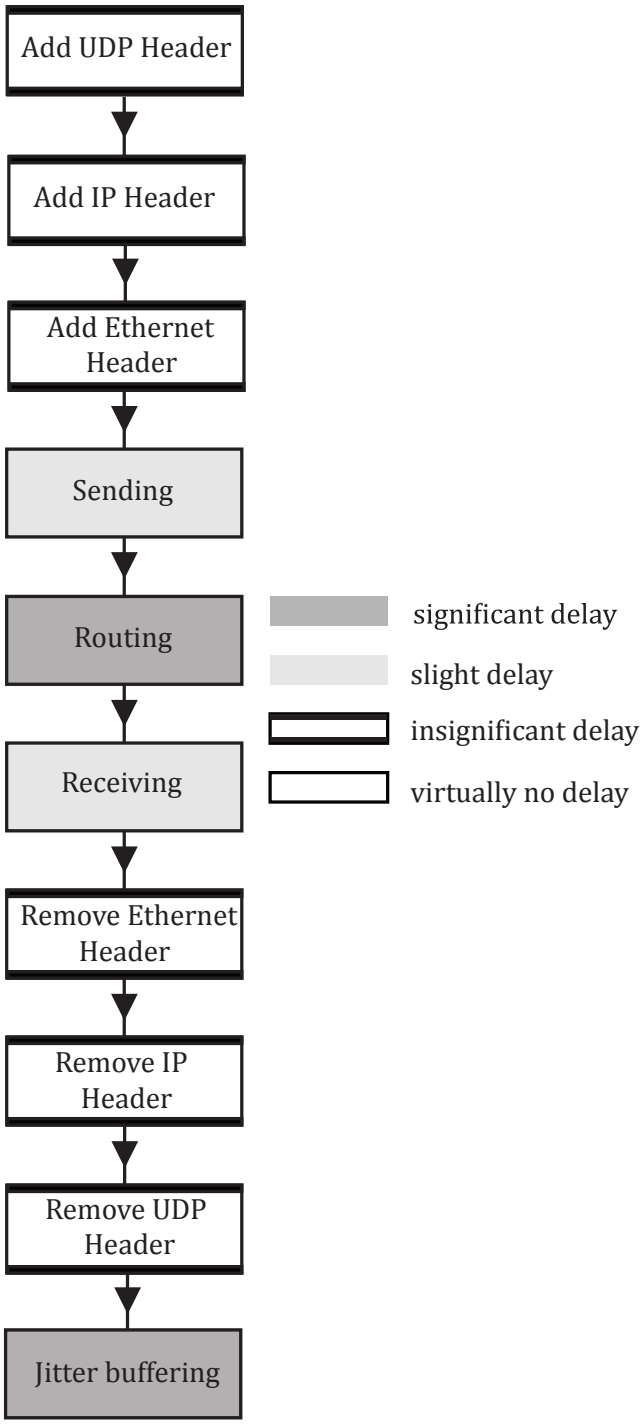


Figure 4.21: Processing stages in the Internet

time traffic demands so that ATM's small cell switching and integrated service level agreements appeared to be a convenient solution to overcome this problem. However, when transmission capacities increased to unexpected dimensions of thousands of Mbps, the transmission duration of packets decreased significantly below a tenth of a millisecond. As a result the transmission of realtime traffic on an asynchronous network of variable data block sizes became an accepted approach and in turn made the comparably complicated ATM cell switching become more and more obsolete.

Digital subscriber line (DSL)

Unlike clients of universities or research companies with fiber optic backbone connections, subscribers from home had no access to a fast Internet backbone before the year 2,000. At that time it was common to use the telephone line as a dial-up connection [147] to an Internet provider, which finally gave access to the next Internet gateway. Dial-up connections use so-called modems (modulator/demodulator), which transformed the digital impulses into an audible analogue signal for the telephone line, so that each dial-up Internet connection technically equalled a regular phone call, for which the line became busy and not usable for further calls at the same time. In terms of transmission capacity modems were able to achieve a maximum throughout of 56 kbps [147], which sufficed for email services and simple text browsing but turned out to be extremely slow in terms of multimedia content such as audio or video. As a result new technologies were required in order to provide fast and convenient home Internet access. Apart from "Internet via powerline" or "Internet via cable TV" the most famous and most often applied solution is the Digital subscriber line (DSL) [60], which uses the higher frequencies of the twisted pair telephone cable in order to access the next Internet gateway.

Digital subscriber line technology was originally implemented as part of the ISDN specification. The central device in DSL technology is a DSLAM (Digital subscriber line access multiplexer), which allows a single twisted pair to carry data and analog voice of the circuit switched telephone network at the same time. The DSLAM applies the frequency division multiplexing in such a way that it shifts the Internet traffic to frequencies beyond the voice traffic frequencies so that they cannot interfere each other anymore [60]. At the customer's endpoint a so-called splitter separates both signals, after which the phone signal is fed to the regular telephone and the DSL signal to a DSL modem. As a result a DSL data connection and a conventional phone conversation can be run in parallel without any interferences. The available data rates depend on a line's damping factor, which describes how strong a signal's amplitude lowers when passing a line connection. Older ADSL standards can deliver 8 Mbps to the customer over about 2 km distance of unshielded twisted pair copper wire to the switching centre. The latest standard can deliver up to 24 Mbps, depending on the distance to the DSLAM. Distances greater than 2 km significantly reduce the bandwidth usable on the wires and in turn the data rate. Generally DSL

is offered with asymmetric data delivery and hence refers to the name ADSL (asymmetric DSL), which implies asymmetric transmission capacities for the data up- and download. This asymmetry is based on a business model, which assumes that DSL clients typically need more download capacity than upload capacity. This results in a significant lower bandwidth for the upload than for the download (generally 10 times lower). Typical upload restrictions range between 128 kbps and 1 Mbps for currently available DSL networks. In that context – depending on the actual connection – an audio stream of 768 kbps does not or just slightly fulfill this restriction, which is why bandwidth reduction by decimation or compression has to be taken into consideration. Furthermore, DSL implies a complex technology including a set of specified protocol layers, as outlined in the following: In order to establish a connection on the twisted pair cable between the DSLAM and the DSL modem (the so-called "last mile"), both have to add further sublayers on the data-link layer. Possible protocol stacks for this purpose are IP over ATM, PPPoA, RFC1483 Bridging or PPPoE [23] of which PPPoE has become the most popular and most commonly applied protocol stack. As illustrated in figure 4.22 the author uses a PPPoE example stack in context with an ATM connection between the DSL modem and the DSLAM. In fact most DSL providers still use ATM as their backbone technology in order to reduce the effect of network jitter [147], however, it is rather uncommon to apply it on the last mile. Nevertheless – according to [23] – this network configuration is generally applied by Belgian DSL providers. Although it represents the worst case scenario in terms of protocol overhead the author consciously uses this example setup in order to outline a significant problem in context with low-delayed audio streams: The example stack consists of six additional protocols: The PPP, PPPoE and MAC layers, which introduce a fixed overhead of 26 bytes, and ATM with two related protocols RFC1483B and AAL5, which altogether generate an overhead of 70 bytes. This adds up in a total PPPoE-stack overhead of 96 bytes [23], which generates an enormous overhead when decreasing the audio block size as illustrated in figure 4.23. An audio stream of 128 samples/block already introduces a fixed overhead of 374 kbps which already exceeds certain link's upload limitations. As a consequence the transmission of low-delayed audio streams is restricted to DSL lines with a sufficient upload bandwidth capacity.

In terms of latency it is important to know that DSL providers normally apply an interleaving [59] for the transmitted data blocks between the DSL endpoint modem and the DSLAM. Interleaving increases the twisted pair transmission reliability by preventing burst errors. The tradeoff, however, lies in a significant increase of latency, which ranges at 50 ms or possibly more. Following the requirements of time critical applications (e.g. online games) providers offer the possibility to switch off the interleaving, which might increase the probability of signal errors but generally decreases the latency down to 3 ms as it would as well be desired in context with a distributed music system. Concerning the routing, the first hops after the DSLAM will always be a router which is able to route the DSL backbone traffic into further networks. In case a remote destination peer is part of the same DSL network it will most likely not leave its native backbone, however, depending on actual network load

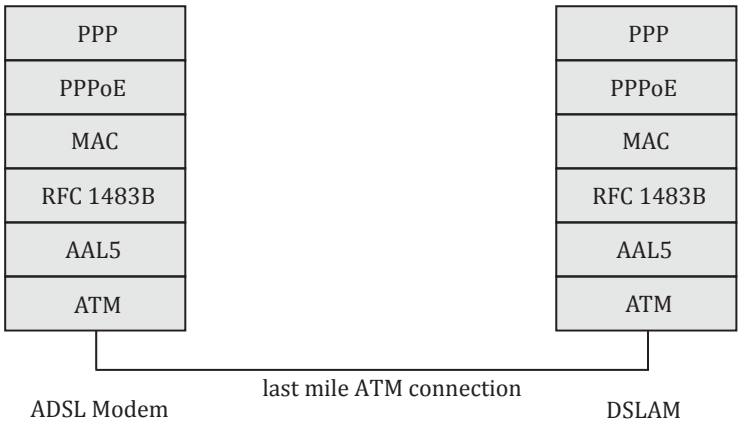


Figure 4.22: A-DSL Modem communicating with the DSLAM via the PPPoE protocol stack on an ATM link [23]

and conditions, DSL providers might use alternative network routing which in turn would increase or decrease the current network latency. This automatically implies that again physically close peers do not necessarily have a close physical network path and the delay might vary significantly depending on the actual situation. This is particularly relative for two closely located peers of different DSL networks: Despite a low physical endpoint distance of several meters, packets would first leave the own provider's backbone and finally enter the foreign provider's backbone. Both provider gateways might be physically totally unrelated, hence packets might possibly pass numerous further hops and possibly make detours of hundreds of kilometers instead.

According to the focus of this elaboration, the previous paragraphs mainly considered the general functionality, latency and bandwidth related aspects of the DSL technology. The interested reader can find more detailed information in [147] and [73].

4.5 Results

Apart from the the natural signal chain, which involves the air as the transmission medium, further signal chains have to be taken into account when approaching a distributed music performance in current digital networks: First the electronic signal chain converts sound waves to electronic waves and processes them in an analogue manner. Secondly, in the digital signal path the analogue electronic signal is converted from its time and value continuous representation to a time and value discrete representation, is sent across a digital network and reconstructed after reception on the receiver's end. Further processing stages occur in the network path depending on the applied transmission technology and its characteristics.

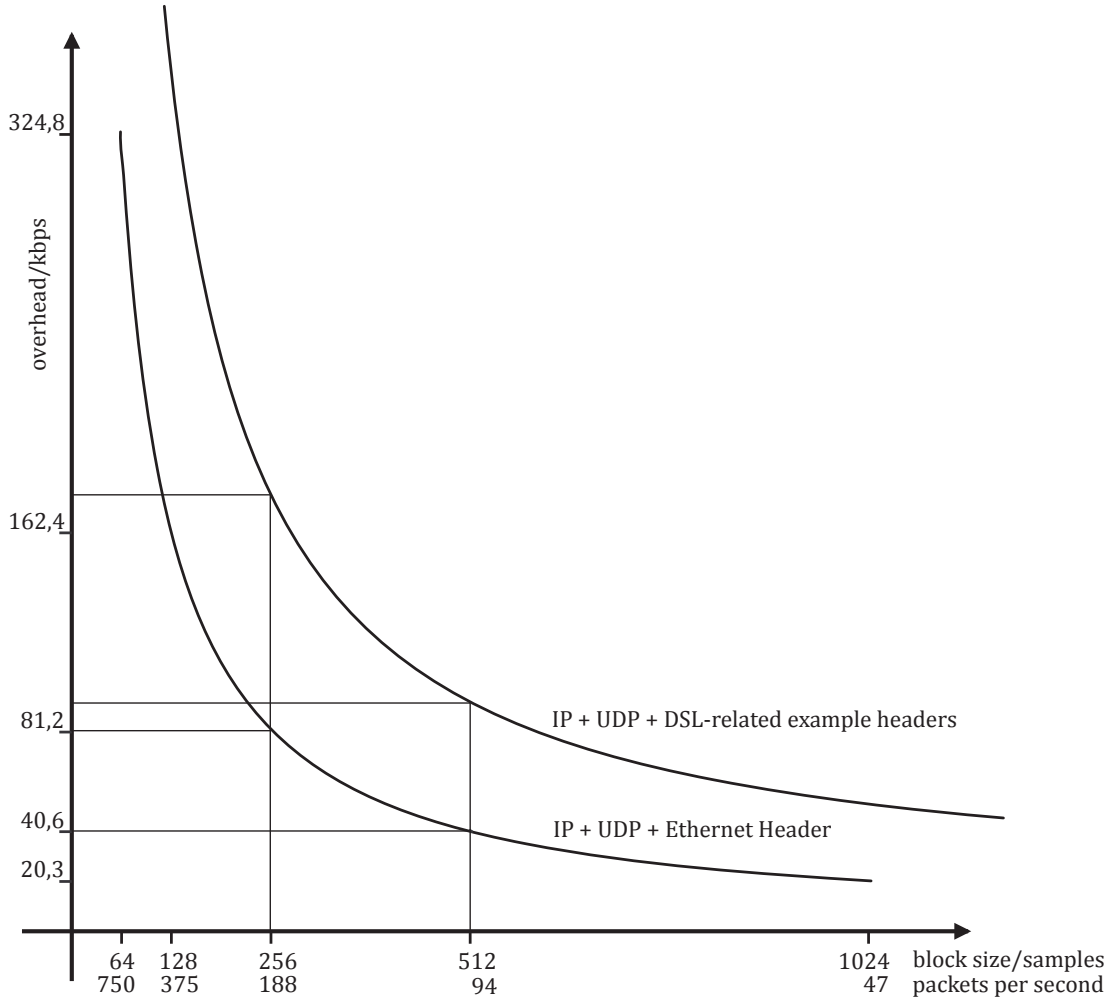


Figure 4.23: Amount of protocol overhead for audio streams

In spite of a relatively large set of signal stages the majority does not introduce a delay, if only slightly or not at all. Within the natural signal chain it is just the physical distance which has to be kept as low as possible due to the air's transmission speed. In the electronic signal path the analogue processing and filtering is not associated with a delay. However, it is mainly the digital signal path, which introduces most of the delay: After passing the anti-aliasing low pass filter, a signal is sampled at a certain sample rate and quantized in a certain bit resolution. The delay of these stages relates to the sample rate and according to the standard of 48 kHz, samples are generated each 20.83 μs [119]. Generally sound cards also apply digital filters to the sampled signal in their input- and output sections. What these filter precisely do is a vendor and device specific criterion but in any case filters require a number of input samples to work properly – the so-called filter length. After the retrieval of this fixed amount of samples the filter generates output samples according to its actual filter architecture. Typically the input and output filters apply an artificial oversampling in order to reduce the effect of aliasing and quantization noise [119]. Digital filters introduce delay depending on their filter length in samples. Hence, when increasing the sample rate, the filter delay decreases proportionally. Nevertheless the major latency appears due to the blocking stages and transmission: Rather than sample by sample, PC-based digital audio workstations process data in a blockwise manner of a preadjusted and fixeded number of samples. Hence, the adjustable sample rate and blocking size determine the delay of the capture process, however, by the double buffer priciple this delay appears on the playout chain as well. In practice most sound cards allow parameters to be set manually and it is generally the user's responsibility to figure the lowest possible delay a system can achieve.

Apart from delays caused by the audio engine of a system, the transmission on a network results in significant delays depending on total distance between two endpoints. Due to the speed of light a realistic music session with the given latency restrictions is only possible within a diameter of approximately 4,000 km but due to cable and routing detours this distance is generally reduced to a maximum of approximately 2,000 km [52]. Furthermore, the question of the ideal music network architecture must be taken into consideration: Theoretically the synchronous telephone network appears to be the predestinated architecture for the transmission of low-delayed audio packets but practically the asynchronous Internet offers various aspects, which finally make it the prime choice for network music performances: Due to constantly increasing bandwidths the device transmission delays increase proportionally and disturbing delay variations decrease respectively. Compared with ISDN and its maximal data capacity of 128 kbps, modern consumer DSL links already offer download capacities of up to 20 Mbps and upload links of 1 Mbps. Rather than low quality and bandwidth limited speech signals of telephone networks, such high bandwidth capacities would already allow even the transmission of full range audio in clear CD quality. However, even if a network connection is not able to deliver uncompressed audio streams, current compression technology is already able to meet the requirements of distributed music in terms of latency and quality. In that context the author introduced two low

delay codecs, which provide high quality sound quality at compression rates of factor 8 and an additional delay below 2 ms.

Furthermore, the Internet offers various of other services such as text messaging, access to world wide databases or unidirectional streaming services, which could be useful features in combination with distributed music in the future. Since these services have already become an integral part of our daily life, the Internet has become an affordable medium, which is generally charged monthly and often accessible in form of a flatrate allowing unlimited use without time restrictions. Although the telephone service has become cheaper over the years as well, especially for foreign and long distance calls, the often applied minute billing model would not achieve a relaxed situation, in which musicians would only concentrate on the artistic outcome of their interaction.

In spite of the Internet's capabilities its ability to transmit realtime data is more complex than it is on a synchronous network: Reducing the block size as low as possible leads to lower delay on one hand but results in a higher amount of packets with a corresponding higher packet overhead on the other. Apart from that and despite increasing bandwidths the most significant problem is still the network jitter, which has a higher impact, the lower the audio block size and in turn the higher the amount of network packets. As a result a distributed music session on the Internet underlies the tradeoff between latency and signal dropouts: The larger the network jitter buffer is adjusted, the lower is the probability for audio dropouts due to late arriving packets – the lower the network jitter buffer is adjusted, the higher is the probability for audio dropouts due to late arriving packets. Altogether it is the sum of numerous parameters, which determines the actual playing condition. In that context it is the quality of the sound card, a system's performance with the applied operating system, the available bandwidth and the amount of additional cross traffic on specific network segments, which together, can be considered of major significance.

Figure 4.24 illustrates any stage an audio signal passes, when being transmitted on the Internet. It sums up the natural, the electronic, the digital and the Internet path to the final total signal path [53].

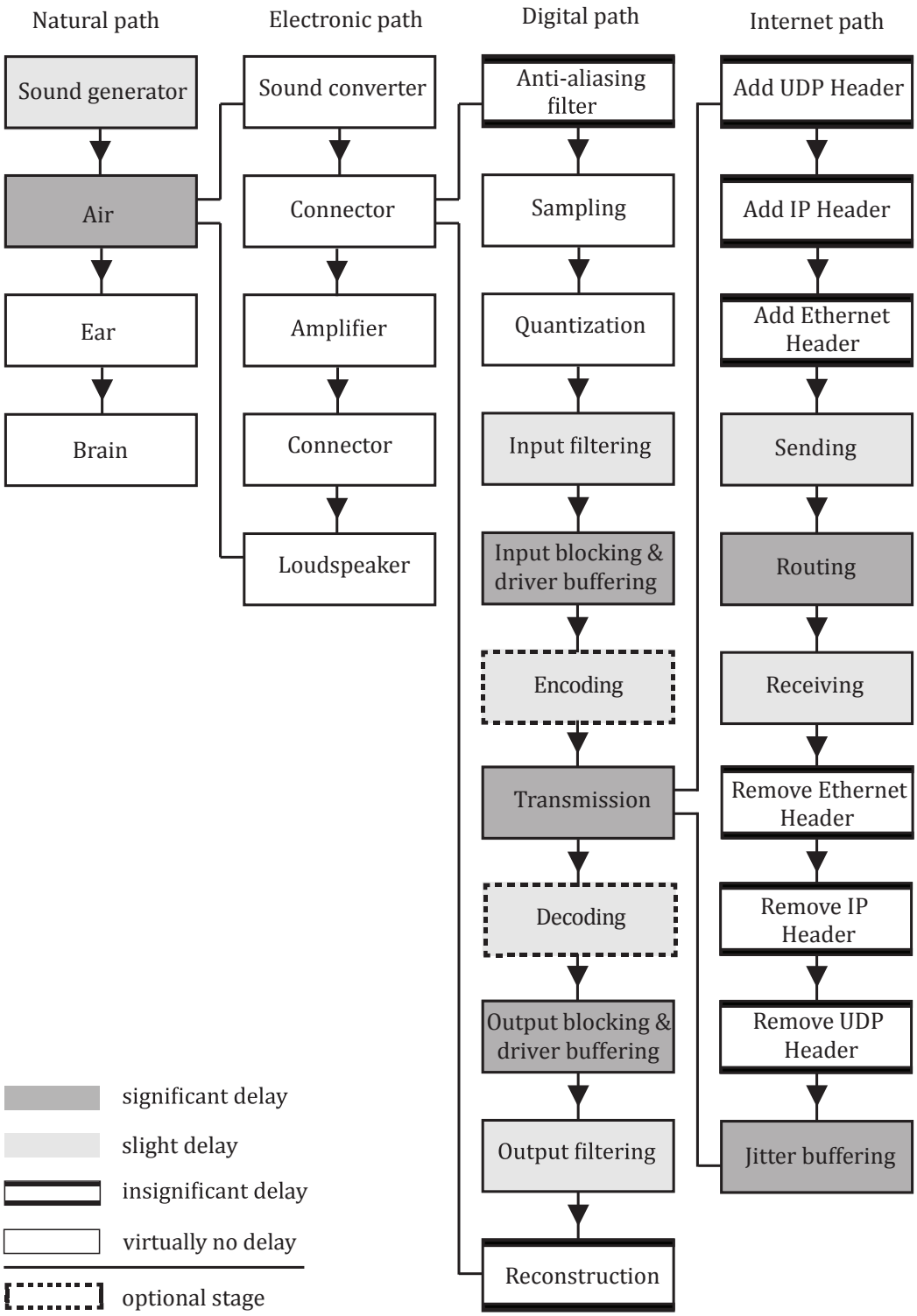


Figure 4.24: Total signal path

Chapter 5

A taxonomy and overview of existing approaches

In the third chapter the author intensively investigated the cognitive aspects of delay-influenced musical interaction and defined the EDAL (ensemble delay acceptance limit) as an individual and speed-dependent measure, below which musical interaction is possible. For delays beyond the EDAL, further terms have been introduced, which describe alternative compromised interaction styles: In combination with an artificially delayed feedback for either one player (SDF = single delayed feedback) or both players (DDF = dual delayed feedback), a second playing style has been evaluated, which improves the situation significantly. The third compromised interaction style consciously separated a solo section from a rhythm section and in turn accepts or tolerates the soloist's artificial "laid back" effect.

As the next step, the author explained any technical aspect in terms of latency in chapter 4 and concluded with the total signal path, an audio signal has to undergo when being transmitted on the Internet. This chapter will bring together the cognitive and the technical results by creating a descriptive categorization of various interaction approaches depending on the expected signal latencies. In this context it examines further situations, in which the EDAL requirements cannot be met. Such a situation can appear when either the physical route length between two peers is too large, strong jitter requires a high amount of audio buffering or various network gear results in additional undesired delays. By assuming that as matter of principle the Internet is not able to meet the EDAL restrictions, a number of projects have already taken compromised approaches into consideration. In that context the related work is embedded into this chapter.

In the following, the author will – as stated in the third chapter – continuously use the terms rhythm and solo section. The rhythm section will consist of drums and bass, the solo section is represented by a saxophone player as a simple example scenario. Based on the actual delay between two players, the author will divide the possibilities of a musical interplay into three main categories A to C. Category A represents the ideal scenario with delays up to the EDAL. Category B approaches to maintain a rhythmical interplay despite latencies beyond the EDAL by applying four different artistic compromises. Category C introduces two further forms of remote musical

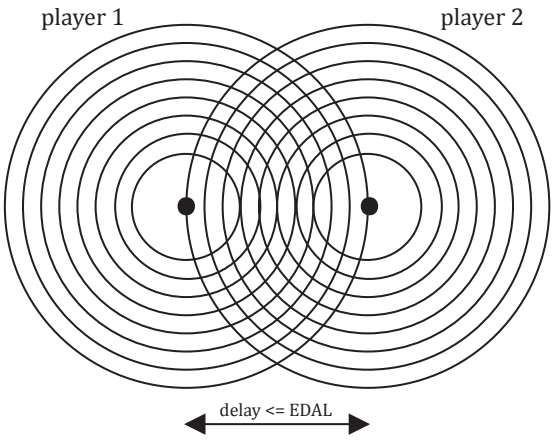


Figure 5.1: Realistic interaction approach

interaction. In this context the currently existing related work is embedded into the appropriate categories.

5.1 Category A – Realistic interaction approach (RIA)

A realistic musical interaction, as if in the same room, assumes a stable one-way latency below the EDAL between two rhythm-based instruments such as drums and bass. In this scenario both instruments' grooves merge and the real musical interplay can happen [55]. From the perceptual point of view the delay appears to be not existent, which is similar to musicians playing with a physical distance of numerous meters in a rehearsing space, where the speed of sound is the limiting time delay factor. The RIA is the only approach professional musicians accept without any compromise since it is the only scenario, which exactly represents the conventional process of creating music in groups or bands. Beyond the EDAL, the groove-building-process cannot be realized by musicians and thus different compromises are required [49]. Figure 5.1 shows that below the EDAL both players are able to play at the same instant and receive each other's signals as if no delay was existent.

Due to technical difficulties in applying the required RIA conditions – as outlined in the fourth chapter –, RIA has so far been considered as an impracticable application. Nevertheless, the Jacktrip software as part of the SoundWIRE research project by Chris Chafe of CCRMA (Center for Computer Research in Music and Acoustics / Stanford University) [56] could theoretically provide music sessions with RIA conditions. However, it rather emphasizes avantgardistic and less beat-driven forms of musical interaction in high delayed international setups and hence cannot be considered as a RIA-optimized entity. This concerns aspects, which will be subject of the

subsequent chapters, where the author's approach for a consequent RIA implementation will be outlined.

5.2 Category B1 – Master slave approach (MSA)

Assuming an attendance to compromise and to step back from musical perfection and ideals, it really is feasible to perform with two rhythm-based instruments such as drums and bass, even when exceeding the EDAL – simply if one of the musicians keeps track of his rhythm and does not listen to the incoming high delayed signal anymore. In that situation the remote side can perfectly play to the incoming signal since the other no longer cares about the response anymore – a change in the musical interaction is happening, which the author terms the "Master/Slave Approach". The first musician takes the master role since he is producing the basic groove, while the remote musician simply relies on it and hence takes the slave role [132]. Of course, the higher the delay, the more difficult the ignorance of the delayed input can be realized by the master, since shorter delays will easier establish a musical connection to the previously played notes. In terms of delay MSA generates no latency and perfect synchronization on the slaves side but on the other hand it delays the slave with the roundtrip delay on the master's side. The slave has a perfect synchronization but musically depends on the master, while the master has musical independency but an unsatisfying sync [49]. Figure 5.2 shows a situation with a delay beyond the EDAL between two players. Due to the high delay the slave has to abstain from playing until the master's signal has arrived, which finally leads to a roundtrip delay on the master's end.

In general the master role is taken by a rhythmic instrument in order to let solo instruments play on its groove in slave mode. An exception can occur when a rhythmic instrument suddenly starts with a solo part. In this case it will require the other instrument to take over the leading rhythmic role, which in turn leads to a switch of roles. MSA can be applied with any system that allows the transmission of realtime data on the Internet. This could be a tool for IP telephony or videoconferencing, which does not put emphasis on low delay signal transmission, but also high speed audio transmitters in an intercontinental setup. In the latter case the main source of latency is the long physical distance. Nevertheless MSA has more to be considered as a theoretical model, which does not provide a real musical interaction. Due to an artistically inconvenient situation it is not likely that musicians would ever use it as a serious musical approach, however, it represents the base for the following two approaches.

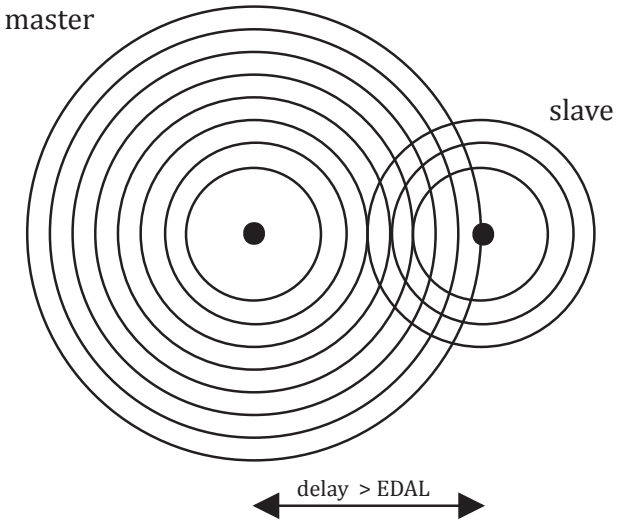


Figure 5.2: Master/slave approach

5.3 Category B2 – Laid back Approach (LBA)

As outlined in section 3.2.2, the "Laid Back Approach" is based on the "laid back" playing manner, which is a common and accepted solo style in jazz music. The Laid Back Approach is similar to the principle of the Master/Slave Approach and is mainly determined by the number of participating instruments and their role. As previously mentioned, two rhythm-based instruments separated by delays beyond the EDAL have to play with MSA but in case of one of the instruments being a solo instrument, the situation changes. Exchanging the drums with a saxophone in the example scenario results in a remote rhythm/solo-constellation in which the bass represents the rhythm instrument and the saxophone the solo instrument. Since the bass no longer has a rhythmic counterpart, it alone takes the responsibility for the groove while the saxophone plays its solo part on it. Equally to MSA the saxophone has a perfect sync on its side and is transmitted back with the roundtrip time but in comparison to MSA this has no disturbing effect on the rhythm instrument in LBA. The saxophone is delayed by the roundtrip delay time, which adds an artificial laid back style on it and hence this playing constellation is no longer considered as problematic. LBA of course does not work for unison music parts, in which both parties have to play exactly on the same beat and at the same time.

The additional delay on the master's end can range between 20 ms up to a maximum of 150 ms (c.f. section 3.2.2). The precise value mainly depends on the bpm of the actual song and the musician's subjective perception [49]. Figure 5.3 is similar to the MSA principle but due to the according lower maximal one-way delay and the

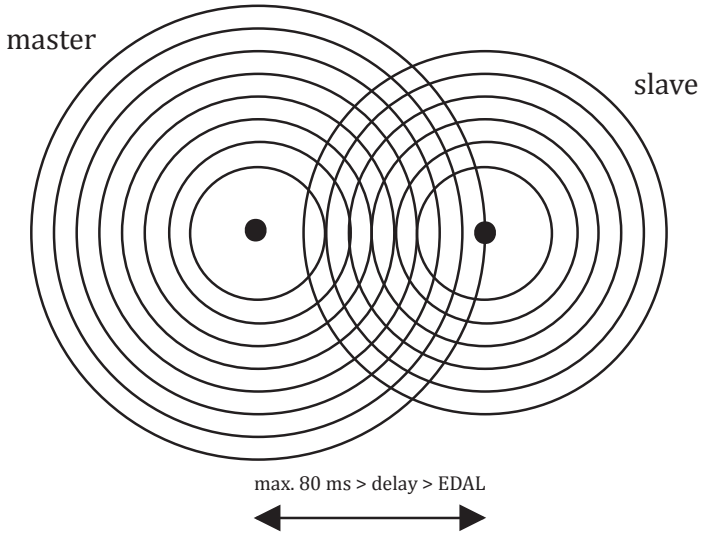


Figure 5.3: Laid back approach

determination of a rhythm and a solo section this situation leads to an artificial "laid back" effect.

LBA can be used when the delay ranges in areas slightly beyond the EDAL threshold up to latencies at which the signal's "laid back" character starts shifting to an "out of time" character and in turn MSA has to be used. Musicians typically do not know of LBA as a serious form of musical interaction and hence it does not represent a popular or commonly applied approach for musical interaction. So far the author is not aware of any practical realization, yet.

5.4 Category B3 – Delayed feedback approach (DFA)

In case the EDAL is exceeded, DFA tries to make musicians feel like playing with the RIA by delaying the player's own signal artificially: By principle delays beyond the EDAL lead to either LBA or MSA, in which the master hears the slave with a delay equal to the roundtrip time while the slave plays in perfect sync. When delaying the playback of the master's signal, both sounds finally have a closer proximity at the master's ear, which improves the problematic delay gap in MSA or reduces the laid back effect in LBA and finally terminates the master/slave separation. Details in context with the DFA can be found in section 3.2.1

The larger the self-delay, the better the synchronization of both signals. The best synchronsization can be reached with a self-delay equal to the roundtrip-time. More descriptively this particular situation is present if one player listens to his own signal

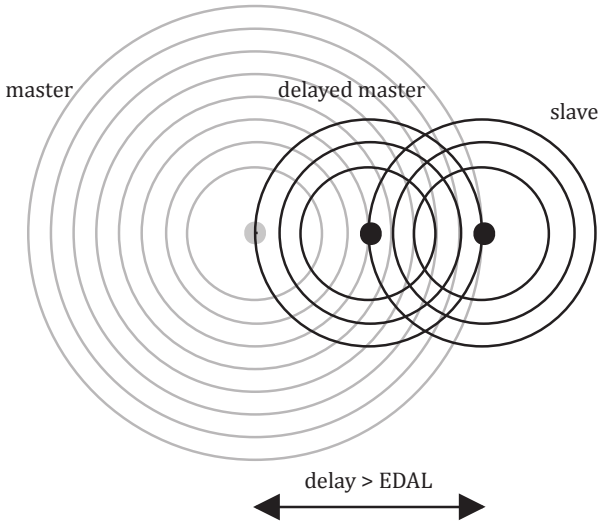


Figure 5.4: Delayed feedback approach with SDF

via a feedback loop on the remote side. As this setup guarantees the elimination of a delay gap between the two sides, rhythmical RIA conditions can be achieved this way if the master consciously plays ahead of time in order to compensate its self-delay. This principle is illustrated in figure 5.4. Since only one player suffers from the artificial self delay, this principle refers to the term SDF (single delayed feedback). A second variation of DFA is present when both instruments play with an artificial self-delay as illustrated in figure 5.5. In this so-called DDF (dual delayed feedback) setup the previously applied roundtrip delay can be provided either symmetrically or asymmetrically to the two players. In either DFA principle the amount of self-delay can be reduced up to a certain limit. This offset value can be calculated according to the equations introduced in section 3.2.1. Nevertheless, although DFA improves the delay situation between two musicians, it is no doubt that a delay of one's own signal typically can be considered as inconvenient and not natural. The larger the delay gets and the louder the instrument's direct noise, the worse the realistic instrument feel and playing conditions.

A system based on DFA is the NcMP project of a research group lead by Prof. Dr. Lars Wolf at the University of Braunschweig, Germany [72]. It uses a central server between two players, which receives either stream, mixes them and sends this mix back to the original sender. This way an asymmetric DDF is applied to the players, however, the route to the additional server results in additional delays, furthermore this principle determines the amount of self delay to the theoretical maximum without the possibility of adjusting a self delay offset. Moreover, an SDF-based audio transmitter has been presented in [45].

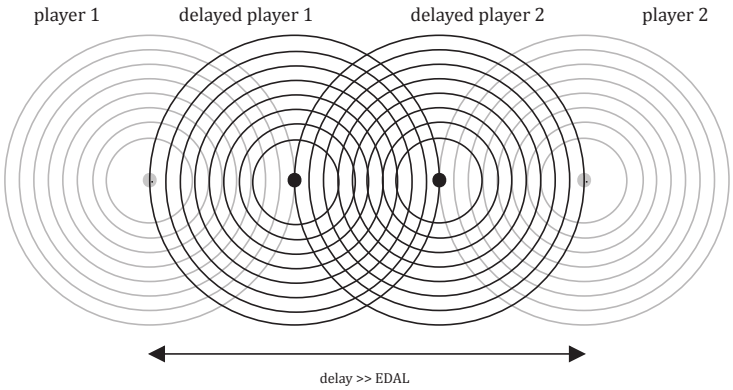


Figure 5.5: Delayed feedback approach with DDF

5.5 Category B4 – Fake time approach (FTA)

As an alternative to the previous approaches the FTA (Fake Time Approach) places an emphasis on musical experimentation and expression and tries to establish a jamming environment under the assumption that per se network latency prevents actual real-time synchronization of the participating musicians. To circumvent this assumption, a communication principle is introduced, in which latency is increased in such a way that participating performers receive each other's output with the delay of exactly one measure. In fact musicians play asynchronously to the music their colleagues have played one measure before. Of course this principle doesn't really allow a realistic interplay but keeps up the vision of such and has a practical identification: In many so-called rock or funk music jam sessions the same basic groove is often played over and over again without any explicit difference to previous measures. In this case FTA might satisfy the musician's needs more than any previous approach apart from RIA as the ideal case scenario. In terms of delay improvement FTA remains an interesting approach, since it increases the delay up to one measure instead of reducing it.

The only representative for the FTA is the Ninjam-Project [1]. Ninjam is a successful client/server based open source software which delays any music stream as described and hence can be applied in any kind of network scenario. Due to this versatility and flexibility Ninjam has already been used in virtual environments such as second life [9].

5.6 Category C1 – Latency accepting approach (LAA)

While previous approaches have attempted to uncover alternative ways to achieve realistic network music performances, the latency accepting approach (LAA) steps

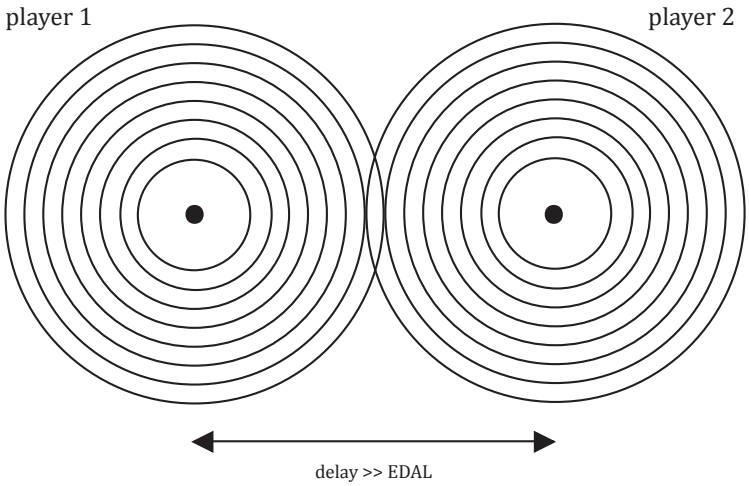


Figure 5.6: Rhythmically unrelated sound sources in LAA mode

back from latency-optimized or compromised solutions and simply accepts delays beyond the EDAL. In principle LAA has no motivation to create conventional music and thus can allow any delay, which is consciously taken it into account. In this scenario musicians play with the delay and use it as an artistic way of expression. LAA is the most avantgardistic approach resulting in a total dissociation of musical conventions and functions with the Internet as the core technology. The latency between the players in figure 5.6 has such a strong dimension that their rhythmical interaction is no longer related.

In terms of new avantgardistic music in LAA, the Quintet.net [8] framework by Georg Haydu [78] fulfills relevant requirements and can be applied under any kind of network condition. Quintet.net transmits MIDI control data and does not necessarily require the user to play a musical instrument, rather the user can play with an electronic input device for the sound generation. Apart from that, various worldwide network sessions with the Jacktrip software [43] as part of the SoundWIRE project have taken place, in which contemporary music is the dominating style of performance [49].

Quintet.net

Quintet.net is an interactive network performance environment invented and developed by composer and computer musician Georg Hajdu. It enables performers at up to five locations to play music over the Internet under the control of a "conductor". The environment, which was programmed with the graphical programming language Max/MSP (developed by company "Cycling '74") [2] consists of five components:

- Server

- Client
- Conductor
- Listener
- Viewer add-on

The basic principle of Quintet.net is illustrated in figure 5.7, where 5 clients, 1 conductor and 1 listener are connected to the central server via UDP links. The players interact over the Internet by sending control streams to the server either using a pitch-tracker, MIDI or simply the computer keyboard. On the server, the streams are copied, processed, and sent back to the clients as well as to the listeners. In addition, a conductor can log onto the server and control the musical outcome by changing settings remotely and sending streams of parameter values as well as giving short commands to the players.

Quintet.net uses a sampler or MIDI for instrumental playback. It also features granular synthesis as well as VST plug-ins (virtual studio interface developed by Steinberg Software) [12] for sound processing and playback, and has additional video and graphical properties, which permit better interaction and control on a symbolical level: The performers along with the audience see the music which the participants produce on screen in "space" notation on five grand staves. In addition video clips and/or live video can be displayed by the viewer add-on and mixed with real-time music notation for an enhanced viewing experience. The conductor can also read musical scores and send parts to the performers, which will be displayed along with the notes produced by the musicians.

The music performed with Quintet.net is typically a combination of composed and improvised elements. The lack of real synchronicity due to the usual delays on the network, necessitates the adaptation of a genuine "Internet" performance style for which John Cage's number pieces could be considered a model: These pieces require certain notes or phrases to be played within "time brackets" [123].

5.7 Category C2 – Remote recording approach (RRA)

This approach uses the Internet connection to replace traditional data transfer mechanisms for audio (e.g. tapes or CDs). Remote musicians can participate in recording sessions by receiving a time-stamped playback of a prerecorded piece and sending back their performed audio signal, the same stamps are attached to. Via this principle the remote signal can can be inserted into the audio mix at the correct position. Time stamping is important to create synchronized audio tracks in the overall mixing result, but the amount of latency of the transmission is irrelevant for the final result. This situation is comparable to the DFA in which the remote musician represents the slave and the playback and recording device represents the master.

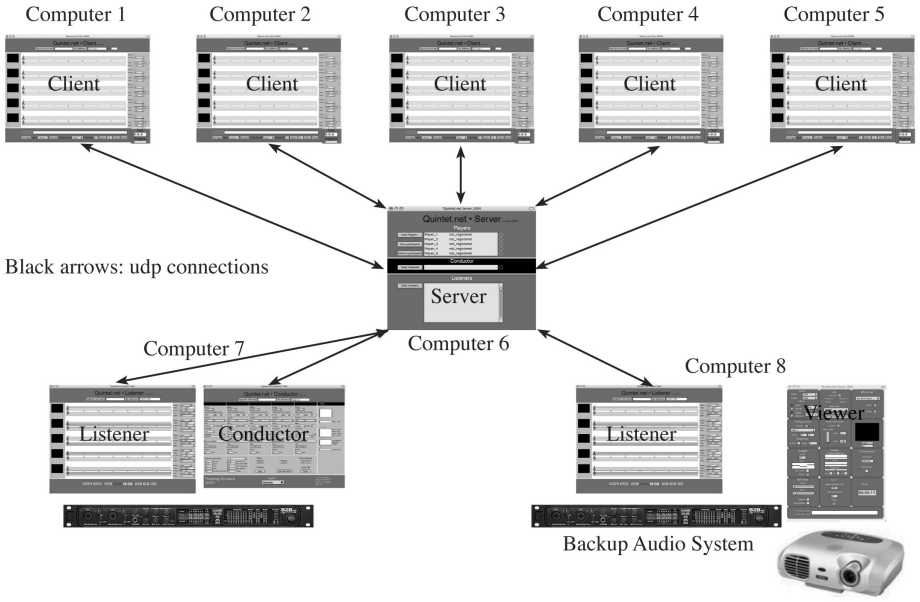


Figure 5.7: Basic principle of Quintet.net [78]

Digital musician link (DML)

The DML [4] is a software which works as a VST plugin and essentially functions like a videoconferencing tool with additional remote recording audio functionality. To collaborate, two logged in DML users agree on a session, one acting as the playback/recorder (A), the other acting as the performer (B). A assigns B a track in his production and starts the recording process. B receives a time stamped stream of the mix that A prepared and plays his specific track to the session, as if situated in a recording booth in A's studio. In a next step B's DML instance assigns each recorded audio block the respective time stamp and sends it back to A, who's DML instance sorts the received data and puts it into the assigned track. The playback of A's mix does not start for A until B's data is received, so for both participants, an environment similar to a recording studio and booth is maintained. The possibility to connect a webcam and a talk back channel rounds up the simulation. Before being able to connect, users need to authorize with a username and password. Depending on what payment model the user has chosen, a connection with low, medium or high quality can be established between two users. Figure 5.8 shows a screenshot of a multitrack recording session with the launched DML plugin.

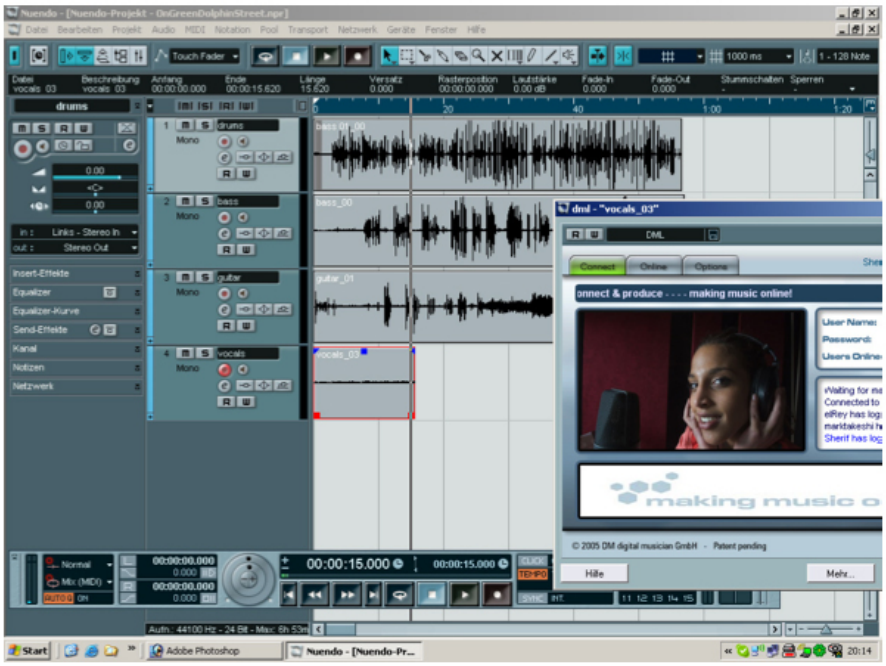


Figure 5.8: Screenshot of a DML-Session

5.8 Results

Depending on an audio device, the physical distance to a remote peer and the amount of network jitter, musicians have to expect signal latencies in different dimensions. According to the actual delay, it is possible to consciously choose a specific interaction category [53]. However, the choice of a category has an influence on the player's perception. In this context five musical aspects matter significantly. In the following table "Sender Sync" and "Receiver Sync" describe the rhythmical synchronisation of the player's signals in a musical interaction. "Own Feel" is an indicator for how realistic playing the own instrument feels for each player. "General feel" generally reflects, how realistic the session feels in comparison to a conventional music session in the same room, while "Unisono play" shows, if musicians will be able to play the same notes at the same time. How well an aspect is fulfilled by each category, is indicated by "+" (good), "o" (average) or "-" (bad). As an exception the following illustrations do not account for the RRA (Remote Recording Approach), since it does not involve simultaneous interplay, and at the same time it cannot be comparatively evaluated.

More generally – apart from audio engineering, network and music skills – the awareness of delay dimensions and their musical consequences is the main basic requirement for a successful network music performance. Based on the technical and cognitive

perfect average bad

	Sender sync	Receiver sync	Own feel	General feel	Unison play
LAA	–	–	+	+	–
FTA	o	o	+	–	–
DFA	o	o	o	o	o
MSA	–	+	+	o	–
LBA	o	+	+	+	o
RIA	+	+	+	+	+

Figure 5.9: Summary of play characteristics of each approach

fundamentals this chapter explains the related categories of delay-influenced musical interaction. Existing remote music technologies, however, are unable to support each of the proposed interaction categories: In fact the MSA could theoretically be realized with any realtime traffic streaming application such as VoIP. Nevertheless, this is not a really acceptable musical compromise and hence it must rather be considered as a theoretical model. The NcMP project exclusively works with an asymmetric DFA. However, it introduces strong self-delays by lacking the possibility of applying a self-delay offset (c.f. chapter 3). Furthermore, the Ninjam project is strictly bound to musical interaction with the FTA and due to its avantgardistic approach the Quintet.net software is restricted to the LAA. Also the Jacktrip software mainly serves music sessions with the LAA, however, it does not compare in the sense that it could theoretically be used in other kinds of musical performances, but this would require further investigation towards a consequent delay optimization. Moreover, the author is not aware of any remote music application which emphasizes the LBA in order to serve a more conventional and realistic playing approach. Finally, it is not surprising that the RIA is not supported by any entity as yet due to the strong EDAL restrictions and the corresponding technical limitations, a successful RIA implementation appears to be extremely challenging. Nevertheless, in the next chapter the author will introduce such an implementation, which is able to support any of the introduced interaction styles including an offset-enabled DFA, LBA or RIA.

Chapter 6

Realization of delay optimized audio networking

In principle a distributed music session has many similarities with a phone conversation and also falls under the category of real time traffic. VoIP has already established itself as a commonly used service for the Internet and one might possibly consider it as a tool for distributed music. However, VoIP makes different demands on a network: As one-way delays of up to 400 ms are considered as still acceptable [88] for a phone conversation, Internet telephony typically applies large playout buffers for audio frames of at least 1024 samples, packet retransmissions or further methods in order to guarantee a solid audio stream playout at the receiver's end. Furthermore, VoIP audio bandwidth requirements generally reside at 8 kHz, which is considered sufficiant in terms of understanding a remote speaker [86].

Unlike this VoIP principle, distributed music has to both approach the reduction of the total network transfer delay to a theoretical minimum and provide an audio quality of a theoretical maximum. In fact, a number of worldwide experiments exist, which consider the Internet a new artistic transport medium. However, as a general tendency, the main focus lies on new artistic forms of interaction based on the LAA (c.f. chapter 5), which consciously takes delays into account as described in the previous chapter. The most significance in that context is the SoundWIRE research project at CCRMA/Stanford, which currently limits its field of application to Internet2-backbone connections. On the other hand, coping with the Internet's delays in terms of the RIA has so far been considered as an impracticable application, and in turn no comprehensive scientific approach towards this direction exists, which meets the time critical requirements of this domain. Hence, in this chapter the author will outline the concept and its implementation, which is able to test an Internet connection in terms of latency and quality and – depending on the situation – allows a realistic musical interaction with the RIA if the appropriate network requirements are fulfilled.

The outcome of section 4.2.7 is that the total network transfer latency depends on transmission capacities and propagation delays. In order to measure the actual delay between two hosts on the Internet, the most common approach is the use of echo request and response messages provided by the "Internet Control Message Protocol"

(ICMP) [122]. The most popular implementation is the "ping" command, which is part of most modern operating systems: By default the ping command sends a sequence of echo request messages to the remote destination. It usually includes the sending time as part of the payload. At the destination the ICMP implementation recognizes this specific type of packet and echoes it back to the sender. Now, the included time stamp gets subtracted from the current system time in order to calculate the roundtrip time (RTT). By measuring the RTT several times an average value can be calculated. Although the ICMP echo response already gives a rough idea of the network latency it does not represent the precise delay conditions of a network music performance as it does not take further relevant aspects of the total signal chain's latency into account.

Besides latency the main problem for low delayed, realtime IP transmission on the Internet is the shared medium access. If an IP packet containing time critical audio data should be sent over the network medium, this sending process might be blocked by other network traffic. Hence, the it must be postponed until exclusive media access is granted. These blocking times result in variations of the end-to-end delay – generally known as network jitter [147]. Since the timing between subsequent audio packets is essential for the provision of a coherent audio stream, jitter can cause a gap in the sequence, which is called drop out [95]. Depending on a sound card's blocking delay, it expects a new audio block with each interrupt.

A common way to reduce the effects of network jitter for VoIP applications is to store up to n received audio blocks in a large network buffer [114]. However, as buffering increases latency, this jitter buffer has to be kept as small as possible for distributed music stream. In the following sections the author will describe the realization of latency optimized audio networking, which implies basic technical aspects as well as desired features in context with the process of the actual musical interplay. Finally, the system will be evaluated in realistic network scenarios, in order to discern, how far the latency of realtime traffic on the Internet can be reduced to a theoretical minimum.

6.1 Concept

As the delay, the effect of jitter and the applied packet buffer sizes depend on the actual audio frame size, the author aimed to develop an audio transmitter, which is capable of sending real audio data of a freely adjustable block size via the UDP/IP [121] [120] protocol to a remote application. At the receiving end the jitter buffer should be freely adjustable according to the actual jitter situation in order to figure the lowest possible values. After passing the jitter buffer, the sound card should be able to play back the incoming stream. If packets are supposed for a delay measurement, the sound card should recognize and reflect them back to the sender. Due to the fact that users typically do not have a clear understanding of delay and network jitter

dimensions, it would be important to achieve an environment, in which they can consciously influence any sound and network parameter in order to compare different settings in terms of quality and latency. In that context, visual feedback information besides the direct audible impression would be useful as this would give a clearer awareness of the current network situation. In the following the author has divided the concept of such a low delay audio streaming application into a network, an audio section and the interface section.

6.1.1 Network concept

In order to send and receive data, an endpoint UDP socket has to be bound to the IP address of the current network interface. In an ideal case it is bound to an external IP address [147] in the public Internet. Hence, the author declared ports for the respective sockets: By default a socket bound to port 4401, sends data to the IP addresses and ports specified as the first and second remote destination. On this port the socket also receives data packets from remote players. In terms of buffering, a FIFO-buffer for each input channel has to be applied, which stores up a desired number of packets before the application actually reads from this buffer. If the endpoint is located behind a network address translator (NAT) [65] it will have to bind to an internal (or private) IP address [147], which cannot be addressed from the public Internet. In this case further actions according to [69] in context with NAT traversal have to be taken into account.

Alongside establishing a real audio connection between two peers, a connection requires testing in terms of throughput, delay and jitter. Hence, an optional application should also be able to generate user specific test data which is sent by another UDP port to the remote side. In order to retrieve this data, the remote side should host a stream reflector or "mirror", which simply mirrors back the incoming data without any buffering or modification. After the sender has retrieved the reflected data, the measured delay time should be shown graphically in a delay gauger. This functionality is considered to work on port 4407. The amount of bytes and the interval times between sent packets should be freely and manually adjustable. As a default setting one packet of 10 bytes is sent each 100 ms. Depending on the actual settings, information about the current actual amount of data in kbps should be available, too. By observing the delay display the user will already get an impression, of how far the current link is able to transmit the data stream, however, a second display can be useful, which explicitly provides information about the remote side's jitter and the local jitter. The less variation that is displayed, the more stable the expected network link.

6.1.2 Audio concept

In terms of audio settings an application needs to support any option explained in the fourth chapter. This implies the choice of the samplerate, the audio block size, the number of audio channels, the bit resolution and the internal sound card buffers. Apart from the setting of these substantial parameters, the user must be provided with the precise audio latency including the delays caused by conversion process and the audio driver. Once the sound card has generated an audio block, it should be parsed to the loopback device in order to be played back locally and, additionally, it should be processed by the network section in order to be sent across a network as a UDP/IP packet. Again, any value should be freely adjustable, which also applies to the volume of the loopback signal and the remote input channels. Similar to volume faders of an audio mixing console, delay faders should exist, which allow the adjustment of an artificially delayed feedback (c.f. chapter 5) for the loopback signal and variable buffer sizes for the incoming audio packets. If this buffer has reached the adjusted number of packets, the sound card starts to read from it in a FIFO-manner.

If a network component is unable to carry an audio payload of 768 kbps, the device would not be able to process the entire audio stream and resultant signal would suffer from strong audio dropouts and distortions. In order to prevent this effect, it is common practice to compress the payload to make it fulfill the current bandwidth requirements. Since audio compression is a complex field with various approaches and numerous advantages, but also drawbacks, regarding a network music performance. However, an effective and fast way of reducing the overall payload size is a reduction of the audio quality. As outlined previously this is typically done by reducing the sampling rate but in turn leads to higher sound card buffering- and playout delays. In order to prevent this increase of delay the principle of decimation and upsampling [117] can be applied, which one hand maintains a fast sampling rate, but on the other, takes only every n*th* sample into consideration. This in turn reduces the sample rate by factor n.

Rather than the pure and unbuffered network delay the real audio delay between two endpoints would be of interest, which includes latencies introduced by the buffering and sound card playback. In that context the author declares the term “DART” as an abbreviation of the “digital audio roundtrip trip time. This value can theoretically be retrieved via the following process: One specific sound block has been marked with a certain signature. This block is sent to a remote destination after a timer is started. After passing the remote end’s signal chain it is finally the callback function, which notices the input of this specific sound buffer packet and sends it back to the sender. At the moment the marked audio block has reentered the origin’s callback thread, the time measurement is stopped and the roundtrip time can be retrieved. As a rough estimate the one-way audio delay is represented by a division of the roundtrip time by 2.

In order to provide better usability, further functionality besides the fundamental audio processing would be desired: If the user likes to record the processed audio stream the application has to be able to write the mixed audio streams into an audio file. Furthermore, a set of test signals such as a sine and an adjustable metronome should allow users to verify a network audio link with a reference audio signal. In that regard it has to be considered, that depending on the actual machine, data might be sent with a different endianess, which results in a different byte order: In the case of a 16 Bit audio resolution machines such as the Power PC Macintosh Comptuter (PPC) first send the so-called most-significant-byte (MSB), which are the first 8 bits of a 16 bits word followed by the second 8 bits or least-significant-byte (LSB). Intel processor-based machines first send the LSB followed by the MSB. Hence, a mechanism is required to reorder the received bytes according to the actual system's architecture in order to make the buffers readable. Without such functionality the signal will be distorted significantly. Finally, any of the mentioned user settings should be stored in a file in terms of a quicker system configuration after a software restart.

6.2 Implementation

There is no doubt that the major requirement for a distributed music system is reliability in terms of high performance, fast audio, and network processing. Hence, one might consider designing a dedicated hardware device but since this work approaches the use of standard personal computers including the integration graphical user interface elements, the author investigated in the optimization of current available and affordable technology towards the defined goals of the previous section. In that context the use of assembly code with the ability to directly access the respective hardware, does theoretically lead to the best performance results. However, since current operating systems such as Linux, Mac OS X or Windows sound and network processing nowadays offer highly optimized C++ application programming interfaces (API), the author considered using them in terms of higher flexibility and ease of use. Furthermore, the author approached a cross platform implementation of a distributed music system in order not to be bound to a specific operating system. Since the Trolltech Qt-Framework [112] provides a comprehensive set of platform independent C++ calls, the author decided to apply it unless certain functionality required the use of a different technology. In the following the author will present those parts of the current implementation, which provide the essential and fundamental functionality for the proposed architecture of a delay optimized audio streamer. The following source-code-extracts provide descriptive explanations in order to outline the fundamental principles, on which further processes of the following chapters will be based.

6.2.1 Audio processing

Various technologies do exist for the processing of audio data with standard PC hardware. The most famous and generally applied solutions are the ALSA (Advanced Linux Sound Architecture) [15] for the Linux operating system, Apple's Core Audio for the OS X operating system [85] and the ASIO (Advanced Steinberg Input Output) for the Windows operating system [22]. Regarding the best achievable audio stability and delay, it is important to directly address the respective driver in order to retrieve the theoretical values stated in the fourth chapter. As each API exhibits significant differences and would in turn lead to completely different and complex audio implementations for each operating system, the author considered the use of an audio wrapper, which provides a common simplified API for all operating systems and calls the underlying API respectively. In that context it is important that – apart from the programming convenience such wrapping provides – it might not lead to any additional delays or performance lacks. The PortAudio I/O library [35] meets these requirements and provides a standard common callback function for the audio processing with the previously mentioned sound architectures. Although of minor practical importance, it additionally supports Windows Direct Sound, Mac OS 9 Sound Manager, Linux OSS, Irix and Beos. Once a PortAudio device is initialized with the `Pa_Initialize())` call, a stream can be started with a set of required parameters, which imply an input-, an outputdevice, the desired sample rate, the block size, and the name of the defined callback function. In order to pass data to and from that callback function, a self defined data pointer can be used. Both calls are shown in figure 6.1(a). Once the stream has started, the callback function is called with each sound card interrupt in order to pass the captured data blocks, held in the `input` buffer, to the system and pass data blocks held in the `output` buffer to the sound card's output. For the user's convenience, the callback function also provides a block counter and a time information as illustrated in figure 6.1(b).

6.2.2 Realtime scheduling

In chapter 4 the author has described the performance and speed of the processes in audio capturing as dependent on the host's performance, the system's architecture, the applied sound card hardware and the respective drivers. However, it is also possible to optimize this performance by choosing appropriate development approaches: In principle, personal computers are designed to run numerous processes in parallel. This includes the video graphics adapter with relevant desktop GUI functionality, various input devices such as keyboard or mouse or any kind of task the operating system or a specific application requires to execute. The advantage of parallel processes, however, has a drawback in terms of low delay audio functionality: The smaller the adjusted audio block size is, the higher the amount of processed audio blocks and the smaller the time interval between subsequent blocks (c.f. section 4.2.5). On the other hand the more parallel processes that are running, the less CPU resources are

```
Pa_Initialize();
Pa_OpenStream(
          &stream, &inputParameters, &outputParameters, sampleRate,
          frameSize,
          paNoFlag,
          ::portAudioCallback, data);
```

(a) Portaudio initialization and audio stream starting call

```
typedef int PaStreamCallback(
                          const void *input,
                          void *output,
                          unsigned long frameCount,
                          const PaStreamCallbackTimeInfo* timeInfo,
                          PaStreamCallbackFlags statusFlags,
                          void *userData );
```

(b) Portaudio callback function parameters

Figure 6.1: Relevant port audio calls

```
#include <sched.h>

int setRealTimePriority() {
  struct sched_param param;
  param.sched_priority = sched_get_priority_max(SCHED_FIFO);
  sched_setscheduler(0,SCHED_FIFO,&param);
}
```

Figure 6.2: Linux realtime scheduling code

available for the time critical audio process. Hence, the lower the audio block size is and the higher the amount of parallel processes are, the lower the probability for the operating system's scheduler to execute the audio process within the required time boundaries. The result is disrupted sound due to audio dropouts. In order to prevent this effect, the audio process must be run with a higher priority, so that the system's scheduler treats them first prior to processing other processes. In order to retrieve this desired prioritization, such processes have to be declared as realtime processes in the program's source code. Apple Macintosh computers with the OSX operating system automatically schedule audio processes with realtime priority. The Linux operating system, however, expects explicit calls within a sound application in order to provide realtime scheduling. Figure 6.2 shows them respectively.

6.2.3 Audio loopback measurement

As outlined in the fourth capter, the conversion and driver latencies can range in significant dimensions and hence require measurement as well in order to retrieve the precise total delay. One could either study the sound card's datasheet and the respective driver properties or measure the offset between the sound card's input signal and the output signal with an oscilloscope. In the former case sound card and driver information data are often not available and could also suffer from imprecise values. The latter method provides absolute certainty about the total loopback delay but requires a rather complicated and expensive setup. Hence, the author applies another method, which works with a cable feedback loop between the physical sound card in- and output. This way a signal playback on the sound card output is directly fed back to the input, so that a loopback measurement can be applied. However, in that context it is problematic to identify the right audio block reference: An audio block, which is marked with a certain digital signature will not change its content when being transmitted digitally, however, after the DA and AD conversion before and after the feedback loops, this precise block content is unlikely to be equal due to reasons such as additional noise, different scaling, or slight signal distortions. Consequently the author uses a special delay measurement technique, consisting of a set of basic sine and cosine waves, which simple shape can be clearly identified even after the respective conversions. This method was originally invented by the European Space Agency (ESA) [17] in order to measure the latency of satellite links. As illustrated in figure 6.3(a), the `loopback::process` function generates five sine and cosine waves at five different frequencies, finally summed up to one single output signal. For each frequency the input signal is multiplied by both the sine and cosine signals, which produces the values for `xa` and `ya`. The `xf` and `yf` are lowpassed versions of `xa` and `ya`. For the loopback measurement the `loopback::process` function is called with every callback in order to pass the calculated signal values to the device output buffer.

Given `xf` and `yf`, the relative phase of the input with respect to the output can be computed for each of the five frequencies by an `atan2(y,x)` in order to finally retrieve the precise loopback delay. This is implemented in the `loopback::resolve` routine as illustrated in figure 6.3(b). It is executed every 250 ms. Moreover, the algorithm contains further special functionality regarding inverted signals and signals errors. However, since this section is supposed to explain the general principle of the algorithm, they are not explicitly outlined at this point. More details in terms of phase-delay-related aspects can be found in [151].

6.2.4 Data transmission

In order to transmit each generated audio block of the callback thread to a remote destination it has to be packed into a UDP/IP packet. This task is realized by a Qt UDP `sendSocket` call, which hosts the audio block as its payload information

```
int loopback::process(size_t len, float *ip, float *op){
  int    i;
  float  vip, vop, a, c, s;
  Freq  *F;

  while (len--){
    vop = 0.0f;    vip = *ip++;
    for (i = 0, F = _freq; i < 5; i++, F++){
      a = 2 * (float) M_PI * (F->p & 65535) / 65536.0;
      F->p += F->f;      c =  cosf(a);  s = -sinf (a);
      vop += F->a * s;      F->xa += s * vip;     F->ya += c * vip;
    }
    *op++ = vop;
    for (i = 0, F = _freq; i < 5; i++, F++){
      F->xf += 1e-3f * (F->xa - F->xf + 1e-20);
      F->yf += 1e-3f * (F->ya - F->yf + 1e-20);
      F->xa = F->ya = 0.0f;
    }
  }
}
```

(a) Loopback signal generator call

```
int loopback::resolve(void){
  int i, k, m;
  double  d, f0, p;
  Freq *F = _freq;

  d = atan2(F->yf, F->xf) / (2 * M_PI);

  f0 = _freq [0].f;  m = 1;
  for (i = 0; i < 4; i++){
    F++;
    p = atan2(F->yf, F->xf) / (2 * M_PI) - d * F->f / f0;
    p -= floor(p);  p *= 8;  k = (int) (floor (p + 0.5));
    d += m * (k & 7);  m *= 8;
  }
  _del = 16 * d;
}
```

(b) Loopback signal resolver call

Figure 6.3: Loopback delay measurement implementation

```
dataPointer->sender->getit(audioBuffer);
dataPointer->waiter->wakeAll();
```

(a) Transmission thread unlock call in audio callback thread

```
void sender::run(){
  while(running){
    blocker->lock();
    waiter->wait(blocker);

    extraBytes = 1;
    audioBuffer[bufferSize] = (char) packetNumber;

    if (send) sendSocket->writeDatagram(audioBuffer,
        bufferSize+extraBytes,*destinationIP, destinationPort);

    if (send2) sendSocket->writeDatagram(audioBuffer,
        bufferSize+extraBytes,*destinationIP2, destinationPort2);

    if (packetNumber < 127) packetNumber++; else packetNumber = 0;

    blocker->unlock();
  }
}
```

(b) Audio block transmission thread

Figure 6.4: Data handover and transmission thread execution triggered by the audio callback thread

and sends it to one or another destination. Since this sending process might possibly block the execution of the audio callback routine and would in turn lead to an unstable system behavior, the author specified a sender thread in an external sender object for this purpose: Within the audio callback function the respective audio block is passed to the sender object followed by a wake up call for its sleeping send thread. This way the audio process and sending process are directly linked without disturbing each other. In order to prevent interference with additional machine processes, this sender thread must also to be implemented as a realtime process. Figure 6.4(a) shows the audio block handover to the sender object and the `wakeAll()` call, which wakes up the sending thread of the sender object displayed in figure 6.4(b). Additionally, with every sender thread execution a packet sequence counter is periodically incremented from 0 to 127. This sequence number is attached to the audio payload for further packet identification in the receiving process.

6.2.5 Data reception

The reception of packets is realized with the signal/slot-mechanism [112], which the Qt framework generally applies in terms of interconnecting and exchanging data with various instances: Whenever the endpoint socket notices an incoming packet, it emits a signal, which results in an execution of a certain predefined slot. This `receiverSlot()` first retrieves the `amountOfBytes` of the received packet, reads the packet from the socket and writes its address to the `input`-character pointer. Unless the maximal FIFO buffer size has been exceeded, the received packet will be processed in terms of analyzing its packet sequence number and writing its audio content to the FIFO buffer. This extraction is realized by dropping the last byte, which contains the added sequence number. If the packets arrive in the desired order the deviation of subsequent packets must equal a value of 1 – otherwise the application notices a packet loss and sets the boolean variable `wrongNumber` to true, so that further processing can be applied. The receiving socket's slot function is shown in figure 6.5(a).

In order to hand the stored audio blocks from the FIFO buffer to the sound card the `pullBuffer` method is bound to the callback function: If the desired buffer size has been reached, one block of the expected size is pulled from the FIFO and written to the `playbackBuffer`. Otherwise the readout process will be postponed until the FIFO holds enough received packets. Once this has happened, the variable `bufferSizeReached` is set to true and the audio callback function will continue reading from the buffer even if the temporary FIFO size changes due to jitter or loss. This functionality is illustrated in figure 6.5(b). However, once the FIFO is empty and hence no packet can be read, the FIFO signals a buffer underrun and `bufferSizeReached` is set to false, so that again the desired number of packets has to be stored before restarting the pull process. Additionally, in case of a FIFO underrun the variable `killnext` is set to true. This prevents the processing of audio blocks, which have caused the underrun due to their late arrival time. Pushing such late packets to the FIFO would be an invalid operation since the correct playback moment has already passed. Hence, this distinction is already made in the receive slot.

6.2.6 Visual indicators

Apart from the audible information the potential, which the distributed music participant expects as the major functionality, the author considers it as useful, to provide as much visible information as possible, in order to comprehensively clarify the playing conditions with the actual network connection. With that regard the Qt framework offers the so-called event loop, which can host various developer defined GUI updates to be addressed with a certain integer identifier. The event loop can be triggered from any associated method or object by posting a respective event with the respective identifier to the main GUI object. As an example the code in figure 6.6 posts

```
void receiver::receiverSlot(){
  amountOfBytes = receiveSocket->pendingDatagramSize();
  receiveSocket->readDatagram(
              input, amountOfBytes, senderIP, senderPort);

  if (!killnext){
    if ( fifo->now_in() < maxBufferValue ){
      bool wrongNumber = false;
      short actNumber = (short) input[amountOfBytes-1];

      if (actNumber - oldNumber != 1){
        cout << "DRIFT: " << actNumber - oldNumber - 1 << endl;
        wrongNumber = true;
      }
      oldNumber = (short) input[amountOfBytes-1];
      if (oldNumber == 127) oldNumber = -1;

      fifo->push(input,amountOfBytes-1);
    }
  }
  killnext = false;
}
```

(a) Receiving network socket's slot function with FIFO push call

```
void receiver::pullBuffer(){
  bool pullBlock = false;
  if (fifo->now_in() >= maxBufferValue) pullBlock = true;
  if (bufferSizeReached) pullBlock = true;
  if (pullBlock){
    datahere = true;
    bufferSizeReached = true;

    playbackBuffer = fifo->pull(playoutBytes);

    if ( fifo->underrun() ){
     killnext = true;
     bufferSizeReached = false;
   }
 }
```

(b) Network buffer check and possible FIFO pull call executed with every audio callback

Figure 6.5: FIFO push and FIFO pull calls

```
if (GUI){
  GUIChangeEvent *ce;
  ce = new GUIChangeEvent(ID,"underrun");
  QCoreApplication::postEvent(mainGUIObject,ce);
} else  cout << "Underrun!" << endl;
```

Figure 6.6: Example of a Qt call for creating and posting a GUI event to the main application's event loop

an event in case of a buffer underrun or – if the user wants to ignore GUI updates – prints the word "Underrun" in the terminal window. Further functionality implies i.e. the application to post a "buffer overrun"-event in case a remote peer sends data too fast or to update the GUI with the IP address and port of an incoming connection.

6.2.7 Delay and jitter measurement

Before establishing a low delay audio connection with a remote participant, as a first step, it is generally recommended to figure out the conditions an actual network connection can offer in terms of delay, jitter and throughput. With that regard the author implemented a traffic generator, which is able to send an adjustable amount of bytes in adjustable intervals to a destination on a predefined measurement port. Every packet, which is received on this port is immediately reflected back to the sender. The author calls this the "network mirror" with the respective "mirror port". Once a packet is sent a timer is started, which measures the time delay until the packet has been reflected back from the network mirror and received back at the sender. Besides the roundtrip latency information the jitter is calculated by the deviation of the previous roundtrip delay and the current roundtrip delay. The result are graphically illustrated in the GUI via the event loop as introduced in section 6.2.6. Additionally the network mirror itself measures the delay between subsequent packets and adds this information to the packet before reflecting it back to the sender. In turn the sender can use this for the evaluation of the one-way jitter before the packet had been reflected. With the subtraction of the remote jitter from the roundtrip jitter the sender can finally conclude about the jitter of the return path from the reflector. Figure 6.7 shows the implementation of the `mirrorSlot()`: Since the slot is directly connected to the network's endpoint socket, it is called whenever the interface notices an incoming packet. Then the time gap between the current call and the previous is measured with a predefined `stopWatch` object. The deviation of both values represents the current jitter at the remote side, which is stored in the variable `actualJitter`. After this calculation the measurement is restarted. As the last step the packet is read from the socket, the jitter information added at the end of the block and immediately sent back to the sender, which finally interprets and displays the information to the user.

```
void soundjack::mirrorSlot(){
  int measuredDelay = stopWatch->elapsed();
  actualJitter  = lastDelay - measuredDelay;
  lastDelay = measuredDelay;
  stopWatch->restart();

  int read = mirrorSocket->bytesAvailable();
  mirrorSocket->readDatagram(buffer,read,reflectIP,reflectPort);
  buffer[read-1] = (char) actualJitter;
  mirrorSocket->writeDatagram(buffer,read,*reflectIP,*reflectPort);
}
```

Figure 6.7: Implementation of a network mirror and remote side measurement

6.2.8 Data reduction

As described in the fourth chapter, no network link is able to carry the full data capacity of a captured audio stream, which is particularly the case for the home consumer A-DSL connections with their restricted upload and larger overhead. This problem can be solved by minimizing an appropriate audio block by a certain factor n, which finally reduces the required bandwidth and also the perceived audio quality. This so-called decimation or downsampling [117] removes every n'th sample value, which in turn leads to n times less samples and hence to an n times smaller buffer size. In order to play back such a decimated buffer on the receiving end, the data must be brought back to its original sampling rate by upsampling, which inserts n samples of value 0 between two respective transmitted audio samples.

Both processes – the decimation and the upsampling process – lead to a sample rate conversion and in that context the sampling theorem must be respected: In order to avoid signal distortions due to aliasing the signal must be lowpass filtered to the desired sampling frequency before the decimation process. Furthermore, it must again be lowpass filtered after the upsampling process since the artificial insertion of samples violates the sampling theorem. The filter is either implemented as an FIR (finite impulse response) filter, an IIR (infinite impulse response) or in a combined manner [151]. The decimation of an original audio block and subsequent upsampling is implemented as shown in figure 6.8. Additionally the author implemented the ULD and the CELT codec, which both lead to a data reduction by factor 8 without a significant decrease in quality (c.f. section 4.2.6).

6.2.9 File operations

Besides the processing of audio streaming data, the application hosts file operations for the recoding, the playback of the recoded audio stream and the retrieval of the settings, with which the application was previously configured. In terms of file based

```
int i = 0;
filteredBuffer=lowPassFilter(originalBuffer);
for (int j=0;j<amountOfBytes; j=j+decimationRate*channelNumber){
  for (int k=0; k<channelNumber; k++){
    decimatedBuffer[i+k] = filteredBuffer[j+k];
  }
  i = i + channelNumber;
}
```

(a) Decimation of the original audio block

```
for (int k = 0; k<amountOfBytes; k++) {
  upsampledBuffer[decimationRate*k] = decimatedBuffer[k];
  for (int i=1; i < decimationRate; i++) {
    upsampledBuffer[decimationRate*k+(decimationRate-i)]=0;
  }
}
finalBuffer=lowPassFilter(upsampledBuffer);
```

(b) Upsampling of the decimated audio block

Figure 6.8: The total decimation and upsampling process

audio block recoding and playback, the appropriate calls are again executed in the external sender thread in order to prevent a blocking of the audio callback function. Once the sender thread has received an audio block, it is directly written into a file provided the user specifies this in the GUI. Conversely, the reading of an audio block from a file happens in the same way if the user request this, however, the read buffer cannot be processed within the actual period and hence has to be delayed until the next callback function call executes.

Furthermore, with each successfully started sound card process each parameter specified in the GUI is written to a text file. This implies the sound card settings, destination IP addresses and ports and the adjusted volume fader values. For convenience after a system restart the application can be reconfigured with the stored values and parameters. In that case a button click results in a read operation of the stored data from the configuration file and updates the relevant GUI elements. Moreover, it is possible to specify the outcome of any measurement to be written into a measurement text file. This implies data such as under- or overruns or delay values or delay times as stated in the following evaluation.

6.2.10 Audio roundtrip time and pulse generator

In order to retrieve the real audio roundtrip time between two connected callback functions, the author implemented a pulse generator as an adjustable metronome as a first step. A GUI element allows the user to adjust the desired metronome speed

```
threadSleepTime = 60 / bpmUserInput->value() * 1000;
```

(a) Setting the metronome thread waiting time depending on a manual value adjustment in bpm

```
if (metronomeSound){
  for (int x=0;x<my->playoutBytes;x++){
  audioBuffer[x] = 'x';
}
metronomeSound=false;
```

(b) Click sound generation for one audio block with each metronome thread execution

Figure 6.9: Principle of the metronome implementation

in beats per minute (bpm). According to the calculation stated in figure 6.9(a) the corresponding milliseconds between two metronome beats are calculated. This time is used as the sleep interval of an external thread, which sequently sets the boolean variable `metronomeSound` in the audio callback function to true. Every time this happens, the current audio block is marked with a predefined character. The author chooses the character "x" as shown in figure 6.9(b). This results in a percussive noise for each marked audio block. In case the user has established an audio stream with another location, the remote audio callback function will notice the marked metronome buffer and will send it back. Once the reflected audio block reenters the source audio callback function, the process is completed. Since the sending and receiving moment are known figures, they can be used for the delay measurement, which finally implies the network and the buffering delay.

6.2.11 Byte swap

The byte swap mechanism has to be applied if two machines with different endianess [90] exchange data, so that the least significant and most significant byte are not in the right order and hence have to be swapped. Since the received data block is represented by a set of 1 byte characters, every other byte needs to be swapped with the precedent byte. The author's implementation of such a byte swap algorithm is shown in figure 6.10.

6.2.12 NAT traversal

As a result of the rapidly increasing number of Internet services and endpoints, the network is soon likely to run out of available IPv4 addresses [147]. Hence, the general idea of a NAT is the extension of available Internet peers connected to one single external IP address. In that case the connected peers retrieve a so-called private IP address, which is only valid within the local network behind the NAT. Nevertheless,

```
if (swap){
  char a;
  for (int x=0; x<amountOfBytes-1; x++){
    a = input[x];
    input[x] = input[x+1];
    input[x+1] = a;
    x++;
  }
}
```

Figure 6.10: Implementation of the byte swap algorithm

the NAT routes traffic of such private IP addresses to the external IP address, so that each of them is finally connected to the Internet. This works in particular for conventional web browsing but in terms of UDP audio streaming a significant problem occurs: Machines on the Internet can be addressed and reached by their external IP address and port but since it is the NAT itself, which holds the external address, an internal IP address cannot be reached. In order to avoid this problem, a network administrator has to enable the "port forwarding" of a NAT, which statically routes traffic on specific predefined ports to a specific machine with a predefined private IP address behind the NAT [69].

However, although the static port forwarding reliably forwards data to the desired destinations, users may lack knowledge about NAT administration or might use security restricted networks, which do not allow custom NAT modifications. In that case another mechanism has to be applied, which allows NAT traversal without enabling the previously described static port forwarding. This mechanism takes advantage of the principle how a NAT treats the TCP traffic generated and sent by the private hosts: Since TCP is a connection oriented protocol, which expects so-called acknowledgement packets transmitted back from a receiver, the NAT has to forward these packets to reach the original private endpoint. Hence, it remembers the destination IP address, which previously passed the NAT on the sending process and in turn allows packets to receive from this specific address and will forward them to the original private IP address. In terms of UDP streaming this means that, once a peer behind a NAT is aware of another peer, which is sending data to its external IP NAT address, it simply needs to send packets to this sender's IP address. In turn the NAT would provide the desired forwarding without having established a static forwarding rule. More detailed information related to aspects of NAT and NAT traversal can be found in [69] and [65].

6.3 Evaluation

As a first step – apart from any network functionality – it is mandatory to verify the system in terms of stable audio processing. Normally it is possible to judge about this by listening to a running audio stream and getting an impression of the amount of lost audio blocks in case a system is not able to handle the actual audio block size. However, to verify this audible impression the author additionally forced a command line alert to be printed once such dropout appears. Finally, the author started an audio stream with the reference sample rate of 48 kHz and decreased the audio frame sizes from 256 down to 16 samples. This test was performed with three sound cards of different qualities running with the MacBook Pro host system – the inbuild Intel sound device, an external Edirol UA25 USB sound card and a RME Fireface 400. After the generation of altogether 10 loopback measurements the author plotted the results as shown in figure 6.11. The figure also contains the pure blocking delay as an ideal comparison value and shows the offset caused by the converter and driver latencies. As a special feature the MAC OS X operating system's audio properties contain information about the driver and the converter delay of each card. Apart from a few slight inaccuracies of up to a maximum of 200 μs the author can confirm the correctness of these values.

As a result it is clear that on the Mac OS X evaluation host each of the tested sound cards is able to process a stable audio stream with block sizes down to 32 samples. Below this value the amount of audio dropouts had to be considered as unacceptable. Furthermore, it is obvious, that the converter and filter delays add a significant amount of delay, which results in a final delay of at least twice the theoretical blocking latency. In that context the inbuilt MacBook sound card exhibited the best filter and driver delay of 3.2 ms.

In a next step – in order to figure in how far the current implementation of the network audio streamer fulfills the latency requirements for a RIA music session on the Internet – the author set up a test scenario, any desired network link was supposed to be tested with. In each test one sending and one receiving peer were equipped with the streaming software. The sender transmitted an audio stream to the mirror port of the remote destination. Since the sender addressed the application's mirror port, the receiver immediately reflected the received packets back to their origin. The measurement settings and procedure – based on this setup – are explained in the following:

According to the previous measurements the sound card could have been configured with a minimal block size of 32 samples. This would have generated UDP packets in intervals of 0.675 ms. However, due to aspects related to jitter and overhead (c.f. section 4) the author first decided to use audio frames of 128 and 256 samples. At a sample rate of 48 kHz this resulted in blocking delays of 2.7 ms and 5.4 ms. Hence the total capture and playout delay corresponded to 5.4 ms and 10.8 ms respectively. On top of that value the filter and driver delays of the inbuilt Intel sound card introduced

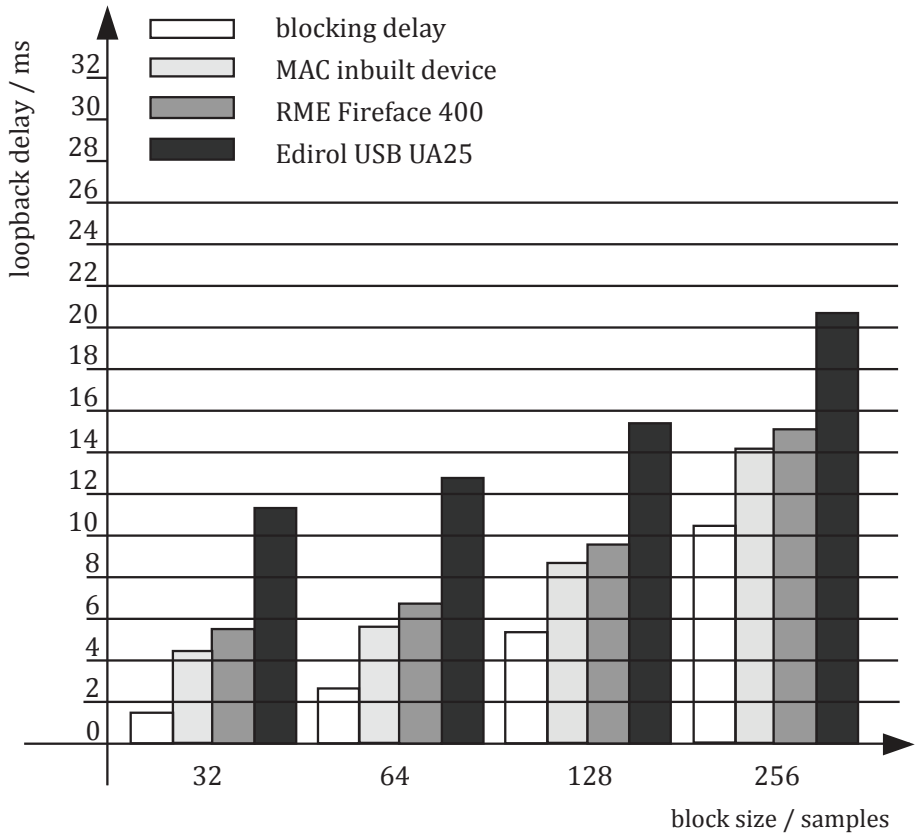


Figure 6.11: Loopback measurement results for 3 sound cards in comparison to the theoretical blocking delay

a fix amount of 3.2 ms, which lead to an overall latency of 8.6 ms for the 128 sample-block configuration and to 14.2 ms for the 256 sample-block configuration. The amount of audio packets for 128 sample blocks was determined to 20,000 and to 10,000 for 256 sample blocks respectively. In either case this resulted in a test duration of 54 seconds. Unlike conventional realtime traffic such as VoIP or videoconferencing, this setup consciously applied a minimal jitter buffer size in order to achieve an adequate signal latency. The current implementation allows the size of the network buffer to be set to one single audio block. Since in this setting the sound card process does not have to wait until a certain network buffer size is reached, it can process an audio block right after its reception. This setting provides a minimal latency, however, it represents a situation, in which, effectively, no real network jitter buffering exists. As a consequence the author applied a network buffer size of two audio blocks in order to guarantee the existence of one audio block between the FIFO's output (audio processing) and the FIFO's input (network interface). Nevertheless, the author has to point out, that even a network FIFO size of 1 single block might provide a certain amount of buffering time, which depends on the sound card's callback moment and the buffer arrival time. However, this time can vary depending on the amount of network jitter, which is why the previously explained solution with 2 blocks was applied.

Since the main goal of this evaluation has been the information retrieval in terms of latency, jitter and audio dropouts, every packet contained an additional time stamp. The roundtrip delay could be measured once the packets arrived back at the sender by subtracting the time stamp from the current system time. Upon reception this duration was divided by 2 in order to retrieve the one-way delay, and this value was written into a textfile. Furthermore, the audio playback of the receiver process was measured in terms of audio dropouts: When a reflected audio block was received in time and could be played out properly, a "0" was written into the text file. In the opposite case, when an audio dropout was observed by the application, a "1" was written into the file. This setup generated delay values and dropout indicator values according to the applied frame size – 10,000 values for packets of 256 samples and 20,000 values for packets of 128 samples. Finally, the measured delay values were plotted into a logarithmic histogram and the dropouts into a linear 54 second time graph. In order to evaluate a worst case network situation, each test was performed at 6 pm in the evening, when a significant amount of Internet traffic had to be expected. Besides this, the author enabled the metronome functionality in order to verify the actual network link by listening to each locally generated metronome beat and its delayed network response.

Altogether tests with 4 remote locations were performed, for which latencies of different dimensions due to their respective route had been expected. Each test was performed with an explicitly used A-DSL endpoint connection with a download capacity of 4 Mbps and an upload capacity of 1 Mbps. In order not to exceed the upload link capacity of only 768 kbps, the author applied a decimation of factor 2 to the sent audio blocks and finally upsampled the received blocks by factor 2. Although this

artificially decreased the sampling rate to 24 kHz, the perceived quality was still three times higher than an 8 kHz VoIP stream and could still be considered as sufficient for music. The order of tests followed the physical distance, starting with another A-DSL peer in the same city at the conservatory of music in Lübeck / Germany (MHS) [19]. The route consisted of 5 hops since packets first travelled to the city of Hamburg and back resulting in a path of approximately 200 km. The results are illustrated in figure 6.12. In terms of network latency this link showed acceptable average result of 8 ms for 128 sample packets and slightly more for 256 sample packets. Hence, the final latency including the audio delay and the delay of 1 additional network buffer frame resulted in 19.3 ms for the 128 sample-block configuration and in 27.4 ms for the 256 sample-block configuration. In terms of signal quality the 256 sample-block configuration did not indicate a single audio dropout within the 54 second test. The 128 sample-block configuration could be considered as stable although two minor glitches in the beginning of the test were measurable.

Secondly, the author measured a test peer at the "Institute for digital media technology" (IDMT) in Ilmenau, Germany. The physical distance between the two cities lies at approximately 500 km. The packets travelled via Berlin, Potdam and Erlangen on altogether 9 hops. As expected, the average latency was higher than in the previous measurement. Figures 6.13 indicate one-way latencies of approximately 16 ms for 128 sample packets and 17 ms for 256 sample packets. In this setup the final latency – including the audio delay and the delay of 1 additional network buffer frame – resulted in 27.3 ms for the 128 sample-block configuration and in 36.4 ms for the 256 sample-block configuration. Regarding the audio dropouts, again the 256 sample-block configuration received each of the reflected buffers within the required time boundaries – in contradiction to the 128 sample configuration, which frequently dropped a number of packets due to jitter in the first half of the measurement. This effect obviously occurs with a certain periodicity.

Finally, the author measured two further test peers in Rambouillet, France and Belfast, Northern Ireland. In Rambouillet the physical endpoint was the youth concert hall "usine chapeaux" (UC), in Belfast it was the "Sonic Arts Research Center" (SARC). If one drew a straight line on a map between Lübeck and Belfast, it would indicate a physical distance of approximately 900 km to Rambouillet and 1,000 km to Belfast. The packet routing to the Rambouillet peer was made of 11 hops and 14 hops to the Belfast destination. The Rambouillet link exhibited one millisecond less delay than the IDMT link and the Belfast link indicated one millisecond more. The delay results are illustrated in figures 6.14 and 6.15 respectively. The Rambouillet link shows a perfect dropout behavior without a single glitch in the 256 sample-block configuration. In comparison, the 128 sample-block configuration results in two stronger dropouts in the first third of the measurement.

Regarding the SARC stream to the city of Belfast, even the 256 sample-block configuration could no longer be considered as stable: Close to block 2,000 a large number of subsequent dropouts appears, followed by 5 less significant and casual errors until

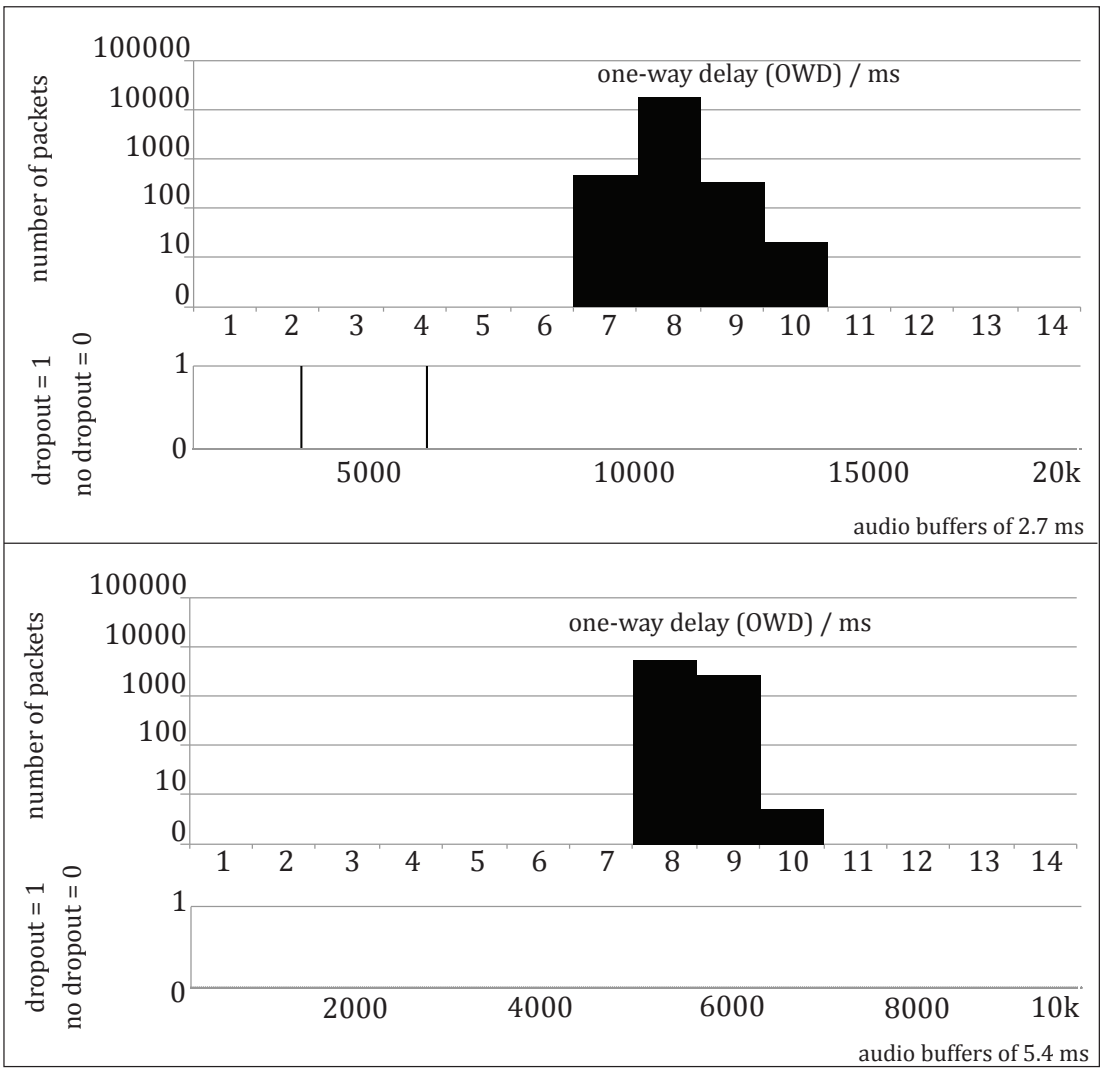


Figure 6.12: Results with MHS peer in Lübeck, Germany: Delay histogram and dropout graph at 128 samples/block (upper) and 256 samples/block (lower)

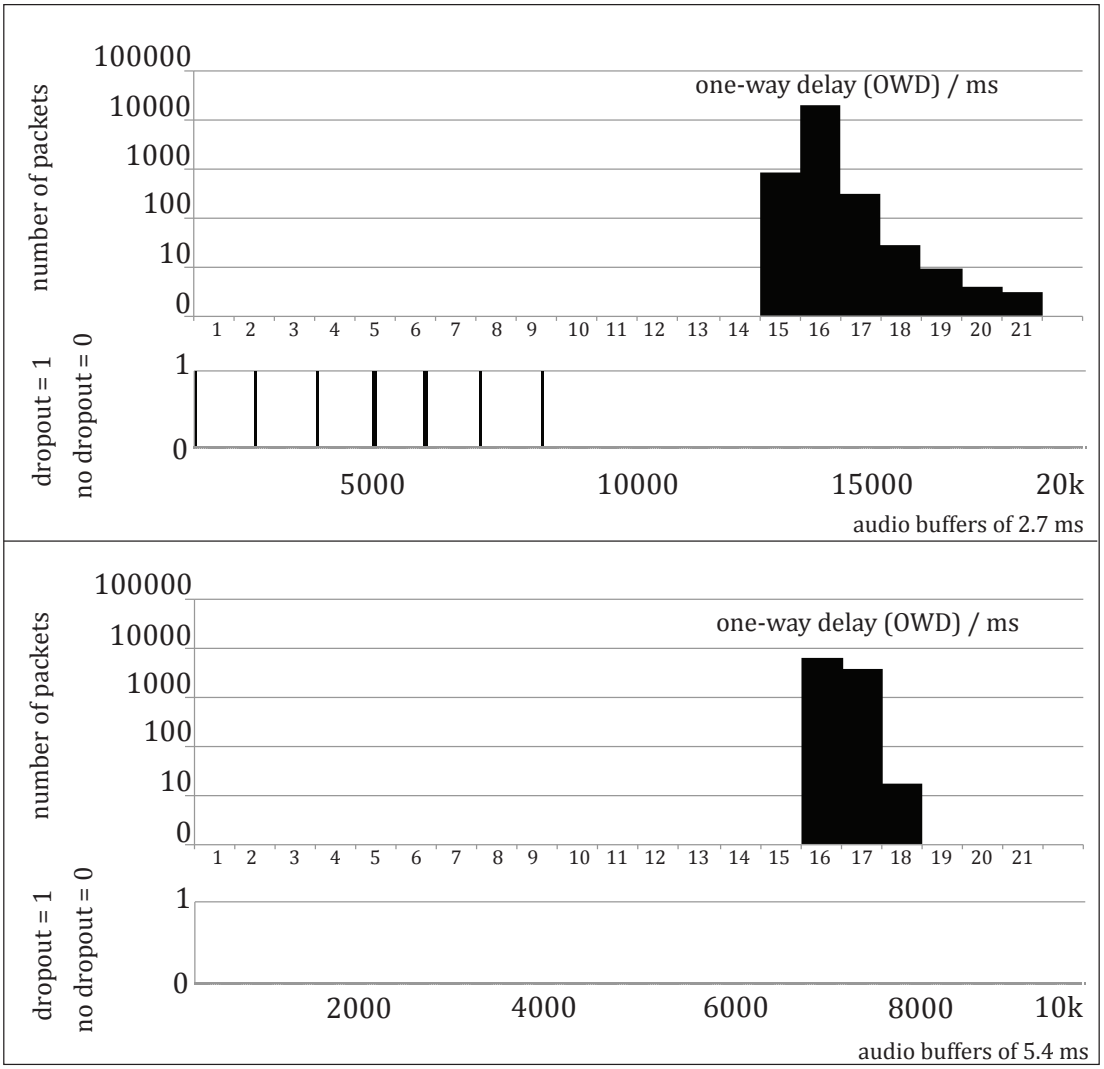


Figure 6.13: Results with IDMT peer in Ilmenau, Germany: Delay histogram and dropout graph at 128 samples/block (upper) and 256 samples/block (lower)

the end of the measurement period. The 128 sample-block configuration, however, leads to an unacceptable result with altogether 60 dropouts of variable occurrence. Furthermore, the author again observed a certain periodicity concerning the moments of dropout appearance.

Additionally the author tested each network link with a 64 sample-block configuration. However, these tests lead to absolutely unacceptable results by indicating dropouts of more than 50 %. Hence, none of these measurements were plotted and illustrated.

6.4 Conclusion

With the current implementation of a delay optimized audio network streamer the author consciously took every technical aspect of chapter 4 into account and developed a system, which allows individual parameter adjustment in order to figure and retrieve the best latency, an audio streaming system can offer depending on the actual network situation. The final block diagram of this system is illustrated in figure 6.16. Regarding the pure audio processing the system is able to treat audio blocks with a size down to 32 samples in an absolutely stable manner. In this ideal low latency case of 32 samples the total blocking delay results in only 1.35 ms (c.f. chapter 4), however, the conversion process and the audio driver add a significant amount of latency. Surprisingly the best delay could be achieved with the inbuilt sound device, which added 3.2 ms on to of the total blocking delay.

Nevertheless, due to the effect of network jitter, it is not possible to send packets of 32 samples with the corresponding blocking interval of 0.7 ms (c.f. chapter 4) on the Internet. As a result the block size, and in turn the respective blocking interval, has to be increased until the best compromise between latency and the number of acceptable audio dropouts is found. Generally a packet size of 256 samples – corresponding to a blocking interval of 5.4 ms – results in an acceptable dropout statistics. Packet sizes 128 samples with the corresponding lower blocking interval of 2.7 ms can be applied but often lead to undesired strong signal interruptions. Below 128 samples the amount of dropouts ranges in intolerable dimensions. Furthermore, due to the high DSL-overhead (c.f. chapter 4) it was not possible to stream full-range audio at 48 kHz on the 800 kbps upload link. Hence, the audio stream had to be decimated by factor 2 resulting in 384 kbps link utilization, which still leads to acceptable audio quality.

Moreover, the author observed a certain dropout periodicity in two measurements, which obviously must have been caused by a periodic cross traffic stream in a specific network segment. Due to the Internet's asynchronous transmission principle there might be situations, in which the lowest possible frame size can be applied but generally the capture block size and/or the respective network buffer size have to be increased. Such idealized conditions can typically be achieved at night, when the overall Internet traffic ranges in such low dimensions that additional crosstraffic barely

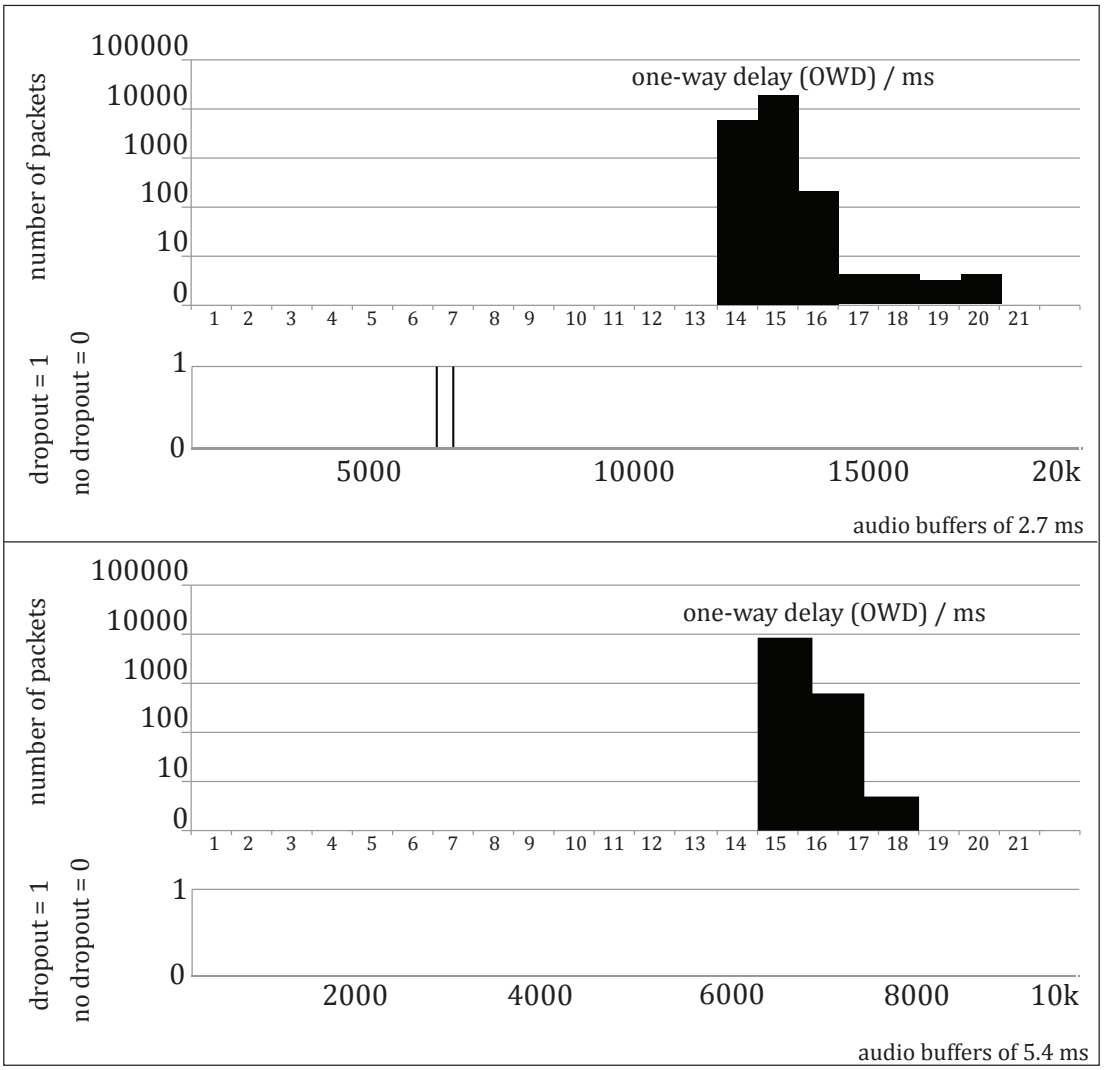


Figure 6.14: Results with UC peer in Rambouillet, France: Delay histogram and dropout graph at 128 samples/block (upper) and 256 samples/block (lower)

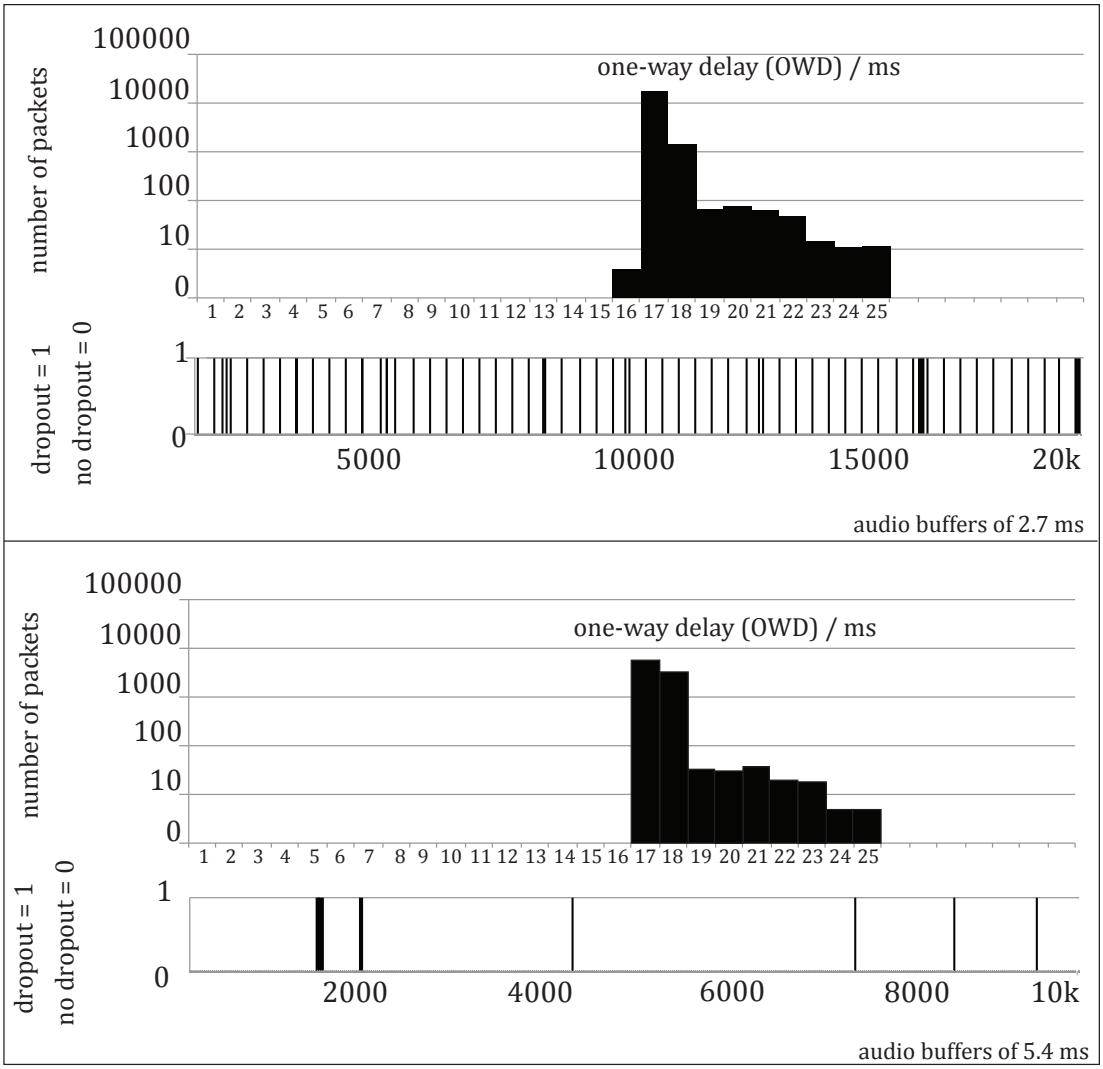


Figure 6.15: Results with SARC peer in Belfast, Northern Ireland: Delay histogram and dropout graph at 128 samples/block (upper) and 256 samples/block (lower)

blocks the respective network link from a sender to a receiver. Indeed the proposed transmission system represents a tradeoff between signal latency and audio dropouts. The higher the adjusted audio block size and/or the adjusted network buffers, the more stability an audio stream will exhibit. However, it will also introduce higher delays. Hence, one has to determine oneself where the individual acceptance threshold for audio dropouts lies and use this as the basis for the appropriate parameter settings.

However – regarding the achieved delay values – the current implementation does not yet fulfill the strong delay EDAL restrictions as figured in chapter 3: Even with the physically shortest route, musicians would have to cope with at least 20 ms of delay, which – depending on the musician and the speed of a tune – can prevent a realistic musical performance with the RIA. For the remaining three national and international connections another 10 ms of delay have to be taken into account, resulting in a total delay of almost 30 ms. This can still allow a musical performance but the probability of an EDAL-overshoot increases significantly. Furthermore, the 128 sample-block configuration may result in an unacceptable amount of audio dropouts. This would in turn force the musicians to increase the block size to 256 samples, which implies another 10 ms of delay. Hence, the current audio streaming implementation still requires further improvements in terms of delay minimization, which will be described in the following chapter.

As a general result we can conclude the following: Describing a communication link on the Internet with the keywords “reliability”, “delay” and “quality” we can see significant differences comparing the author’s approach with the principle of conventional VoIP: The latter assumes a stream playback to be as reliable as possible and in turn as delayed as necessary. Furthermore, it suggests an audio quality as good as necessary. In contradiction, the distributed music principle assumes a delay as small as possible and in turn a stream reliability as solid as necessary, while assuming an optimal quality. Hence, due to these different requirements, it is clearly obvious that distributed music may not be considered to equal conventional VoIP principles and in turn represents a communication category of its own. Nevertheless, according to the desired functionality, the implemented system offers the choice between either principle by respectively adjusting larger or smaller audio block and network buffer sizes.

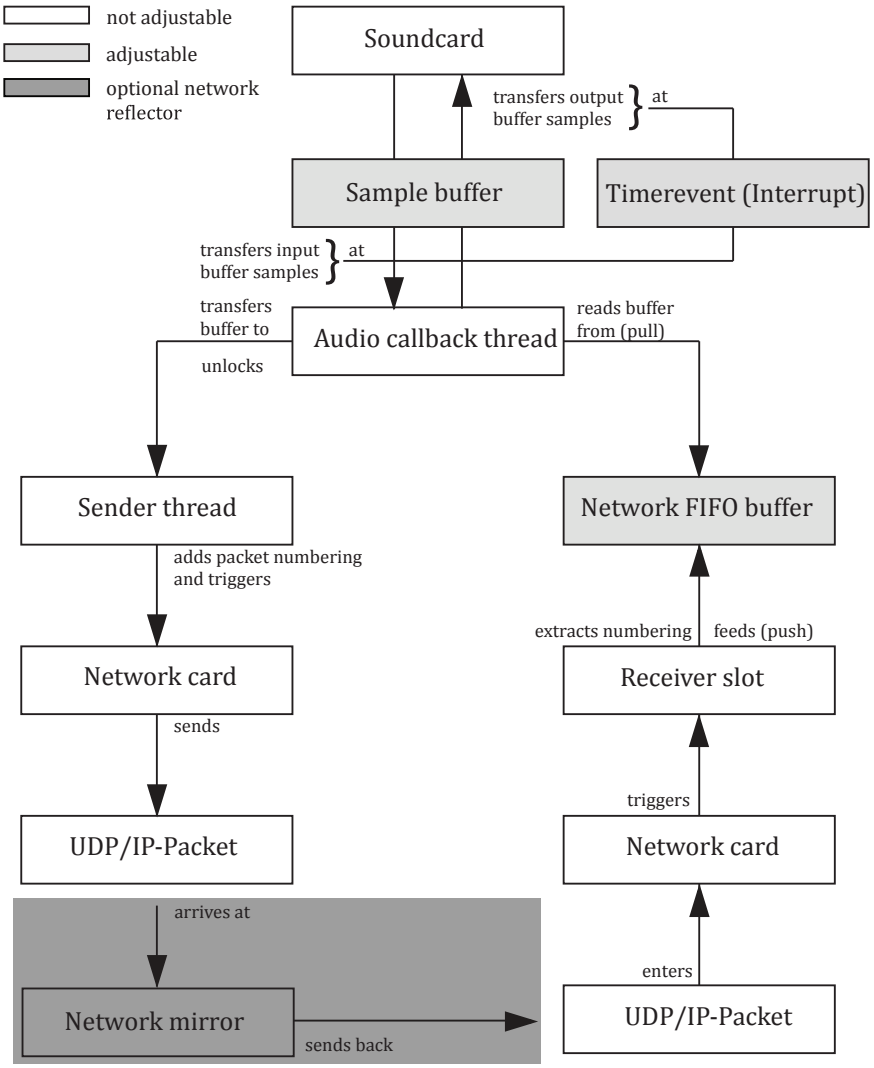


Figure 6.16: Block diagram of the delay optimized audio networking

Chapter 7

New sound- and network engineering approaches

A concert on a stage in front of an audience requires essential knowledge in the domain of audio engineering and acoustic effects in general. Sound engineers typically provide two independent sound setups – the stage sound and the FOH (Front of house) sound [63]: The stage sound is required to deliver ideal conditions for the involved musicians on stage. This implies one's own sound and the volume relation with other musicians. So-called stage monitors are typically placed on the stage floor and emit the signal in a 45 degree angle towards the musicians. Apart from volume level adjustments stage monitors also compensate disturbing delay offsets on larger stages due to the speed of sound. The FOH mix, however, is independent of the stage mix and provides the sound for the audience. Depending on the concert venue and size, it implies complex speaker combinations of subwoofers, midrange speakers, tweeters and the respective larger volume levels. Its volume relations typically differ significantly from the stage sound as it is supposed to present the final outcome of a musical performance to the audience. Nevertheless, both mixes might interfere: Since the FOH speakers normally reside numerous meters in front of the stage, the audience might notice parts of the stage sound as a disturbing reverb. In order to prevent this effect, engineers often apply an artificial delay to the FOH mix [63].

In this sound engineering example a situation is presented, in which a modified technical setup as well as additionally applied technology helps to establish a more convenient situation for the involved musicians, engineers and audience. As a distributed music session represents an even more complex scenario, it consequently shows a large potential for such improvements in different areas. This implies various musical and technical aspects related to latency, the audio and the network engineering. Within the development and evaluation of a low delay audio streaming concept the author discovered a number of new sound and network engineering approaches, which will be presented in the following sections.

7.1 Asymmetric sound- and network processing

According to the results of the previous chapter it became clear, that the author's current implementation of a low-delay audio streamer achieves latencies, whose range is of a significant lower dimension than VoIP or videoconferencing systems. In order to achieve RIA conditions, the maximal distance is typically restricted to a maximum of 1,000 km. However, one has to be aware that with every additional millisecond due to an increased physical distance or increased buffering, the risk of an EDAL overshoot grows. As outlined in chapter 3, this threshold depends on the involved players and the speed of a musical performance. Comparing the range of EDAL values with the measured latencies of the previous chapter, it is clear that the processing latency of the current implementation suffers from an interdependency with the actual network conditions: Since a larger block size corresponds to a larger delay, a logical consequence is a block configuration with the lowest values the system can offer. However, as the network jitter determines the smallest adjustable block size, one cannot freely choose this theoretical lowest value and in turn the latency might increase significantly. At a block size of 256 samples the total audio latency including one network buffer already resides at 11.3 ms. Hence, in combination with the network latency, an EDAL exceedance is likely to happen at any physical distance below 1,000 km.

7.1.1 Concept

The authors's new concept of delay optimized sound card processing terminates the dependency between the audio- and the network block sizes by a slight modification of the sending process: As a preliminary step the user needs to figure the lowest possible block size a sound system can provide and configure it respectively. Secondly, if due to jitter the network requires larger packets to be sent less frequently, the sender would collect up to n desired audio frames into one packet and send this larger packet across the network. Once this packet is passed into the destination's receive buffer, the sound card would read from it. Since the sender artificially stores up n blocks of audio to one single network block the actual receiver's sound card block size is made of $\frac{1}{n}$ of the received block size. Hence it requires n read periods to finally retrieve the full audio block. Assuming a stable system with reliable sound cards on both ends, which provide a solid buffer generation, this asymmetric sound- and network processing configuration would decrease the playout delay by factor n compared with one single block reading period.

7.1.2 Implementation

The asymmetric sound and network processing approach does not affect the playback process of a remote destination, where each buffer content is read with a minimal

possible block size as if it would equal the received network input buffer size. However, the sender has to work in a modified manner by collecting n blocks to one larger block. First of all the blocking factor has to be calculated by dividing the adjusted send block size by the adjusted audio capture frame size as shown in equation 7.1. E.g. in a situation with strong network jitter a large send block of 1024 samples would be applied combined with a minimal audio capture block of 64 samples. This would lead to a blocking factor of 16.

$$\text{blocking factor} = \frac{\text{send block size}}{\text{capture block size}} \tag{7.1}$$

Since the current implementation of an audio network sender is already implemented in a separate thread, it will additionally host the asymetric blocking functionality: Like previously, the network sender thread is unlocked by the audio callback thread. Rather than immediately sending the received audio buffer, it collects n chunks of audio until the incremented chunk counter has reached the calculated blocking factor. With each call the final send buffer's samples are being addressed via a position counter, which is incremented until the final position is reached. Once this is the case, the buffer is handed to the next processing stage and the position and chunk counters are set to zero. If a user adjusts an equal capture and send buffer size, this leads to a blocking factor of 1 and hence does not require the described sample collection process. In this case the audio buffer is directly handed to the next processing stage.

7.1.3 Evaluation

In order to evaluate the proposed principle of asymmetric sound and network block processing, the author firstly examined the measurement results of the previous chapter in terms of latencies, which might lead to an EDAL exceedance: According to the measurements of section 6.3 – if the Rambouillet link is configured with 256 sample-block setting – the total latency results in 35.4 ms. The author repeated this test and plotted the dropout graph as illustrated in the upper part of figure 7.2. This time the Rambouillet link exhibited one single audio dropout close to block number 4,000. Subsequently the author changed the sound card blocking to 64 samples and repeated the measurement on the same link. The results are illustrated in the lower part of figure 7.2. The amount of read audio buffers is 4 times larger than in the previous measurement as one block of received 256 samples had to be read with altogether 4 periods of 64 samples. Despite this modified transmission principle, the figure shows that only one single dropout occurs, just like in the previous measurement. In terms of latency, however, the reading process takes 1.35 ms instead of 5.4 ms.

```
while(running){
  blocker->lock();
  waiter->wait(blocker);

  if (blockingFactor > 1){
    short j=0;
    for (int i=positionCounter; i<positionCounter+playoutBytes; i++){
      sendBuffer[i] = audioBuffer[j];
      j++;
    }
    positionCounter += playoutBytes;
    positionCounter++;

    if (positionCounter >= blockingFactor){
      this->sendBuffer(sendBuffer);
      positionCounter = 0;
      chunkCounter = 0;
    }
  } else this->sendBuffer(audioBuffer);

  blocker->unlock();
}
```

Figure 7.1: Sample collection thread for asymmetric sound card processing

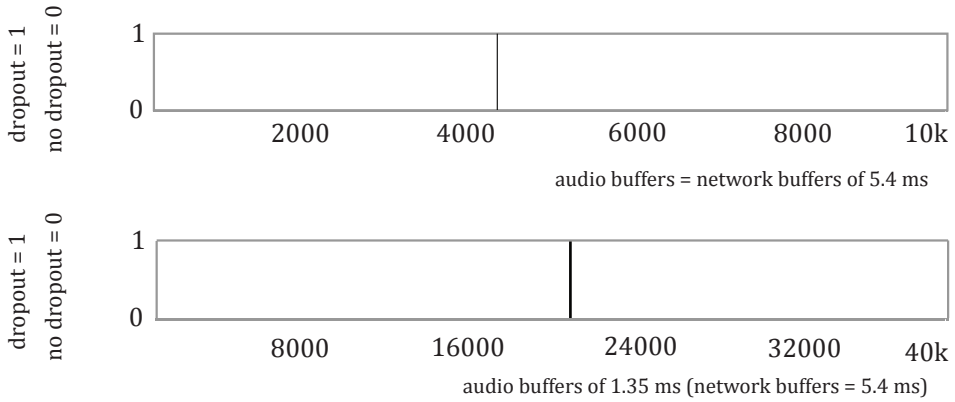


Figure 7.2: Comparison of the regular 256 sample-block read performance and an asymmetric configuration of 256 sample network packets with 64 sample audio blocks (link between and Lübeck and Rambouillet)

7.1.4 Conclusion

The asymmetric sound- and network processing successfully reduces the audio blocking latency to the practical minimum, the actual sound device can work with. However, it still allows the user to adjust an appropriately larger network packet size, which suits the actual network conditions. In turn this principle reduces the capture and playout delay significantly and – according to the actual network conditions – helps to achieve the lowest possible delay: In the current test case the author applied network buffer sizes of 256 samples with an audio blocking of 64 samples. Since the sender has to generate blocks of 256 samples, it has to collect 4 audio blocks of 64 samples first. This does not result in a latency improvement. This actually happens in the receiving process: When the network queue is filled with n blocks of 256 samples, the sound card can process each block 4 times faster as if it was configured with a block size of 256 samples: Instead of a full period of 256 samples to read a whole block, it needs 4 blocks of 64 samples, and this implies that the card generates an audible playback after having processed the first 64 samples. Hence, the read period size could be reduced by factor 4 from 5.4 ms to 1.35 ms. In turn the total latency for the Rambouillet test link could be reduced from 35.4 ms to 31.3 ms.

7.2 Audible ICMP echo responses

As outlined in the previous chapter, the most popular implementation of the ICMP-protocol is the ping-command as a default part of most modern operating systems. It provides information about the roundtrip delay between two hosts. In order to analyze the actual routing between the sender and the destination host, there is a second commonly available monitoring tool called “traceroute”. It makes use of the “Time To Live (TTL)” header field of the IP header [121]. The numeric value of the TTL field is decremented by (at least) one if the packet is forwarded to the next hop. If a value of zero is reached, the packet must be discarded. This mechanism prevents packets from circulating endlessly on the Internet in case of a faulty routing setup. The traceroute command uses the TTL value to reveal the routing hosts on the way between source and destination. It starts sending out an arbitrary UDP packet [120] with a TTL value of 1. The first hop decrements it to 0, drops the packet and sends back an “ICMP time exceeded” message. Hence, the source host can detect the first hop by reading the IP sender address of the ICMP message. Next, the traceroute command repeats this process with a stepwise increased TTL value until all hops on the route are resolved. By combining ping and traceroute the values of RTT and jitter can be measured for any hop of the route.

Theoretically the combination of the ping and traceroute command represents a suitable technique for the retrieval of the actual delay and jitter values in order to determine parameters of a distributed music session. In fact, as well ICMP is a connectionless protocol, however, since audio data is generally transmitted via the UDP

protocol, ICMP measurement results no not necessarily represent the characteristics of a UDP data stream. Furthermore, in case of a network music performance, using ping and traceroute without the actual audio payload would not represent a realistic distributed music scenario. Alternatively one could establish the UDP audio transmission and perform jitter measurements via the ping command in parallel. Nevertheless, a severe drawback to this approach is that the receiver must be up and running, furthermore, it implies that the remote musician must start according services. With additional network components, such as NATs or firewalls, the setup becomes even more complicated and less userfriendly to use [46]. Besides this, the jitter measured via the ping command is only represented as a set of numeric values. These do not necessarily correspond to the perceived quality of the transmitted audio streams, because it depends on several potentially undetermined technical parameters such as packet and buffer sizes.

Nevertheless – despite these facts and drawbacks – the author's motivation is to prove, in how far the ICMP protocol can be used in terms of jitter and delay measurement in order to retrieve representative network parameters in a practical and realistic measurement setup.

7.2.1 Concept

A first significant improvement toward this goal would be to include the audio information in the packets for jitter measurement. These would include the actual audio data combined with a time stamp – analogously to ICMP echo request messages. This approach would require to place a UDP echo service on every hop of the route in order to mirror back the relevant audio stream. This in turn would give the listener an audible impression of the current jitter conditions on this specific part of the route. However, it is usually unpractical to set up new services on foreign Internet routers. In that context the author examines, whether an ICMP implementation, which is present at any Internet router, could behave in the same way as a UDP echo service. This would enable the user to test any hop on a given route without setting up a custom endpoint. In fact, both UDP and ICMP are connectionless protocols, however, they have so far been used for completely different purposes. Hence, each measurement will be performed in direct comparison between both protocols in order to achieve awareness of possible inconsistencies.

In the following the author will show that it is possible to include audio payload data in ICMP request messages and use echo responses as an audible representation of the current jitter conditions on the route to an arbitrary Internet host. As the author will demonstrate, this is a convenient and flexible method not only for jitter measurements but also for a great variety of other audio relevant network parameters. The most important practical benefit in comparison to existing solutions is the possibility to measure without setting up custom endpoint services on the remote side.

```
struct PingICMPPacket128{
  struct icmp icmpHeader;
  struct timeval packetTimeStamp;
  char audioBuffer[128];
};

struct PingICMPPacket256{
  struct icmp icmpHeader;
  struct timeval packetTimeStamp;
  char audioBuffer[256];
};

struct PingICMPPacket512{
  struct icmp icmpHeader;
  struct timeval packetTimeStamp;
  char audioBuffer[512];
};
```

Figure 7.3: Numerous block size-dependend ICMP audio structs

7.2.2 Implementation

The implementation of the audible ping solution is made of an ICMP sender and an ICMP receiver. Like with UDP transmission the callback thread first triggers and feeds the sender thread (see chapter 5) with an audio buffer. If the user has consciously chosen to send this buffer via the ICMP protocol, the sender thread uses further calls of a custom `ICMP` class: This class works with a conventional ICMP struct, however, the main difference is the actual structure of the ICMP packet: Besides the normally used icmp header and a timeval field, the struct additionallty contains an array of characters in order to host the audio buffer. Since the user can freely choose a desired audio frame size, the author implemented numerous ICMP audio structs in order to support any buffer size.

The appropriate sending call `CreateAndSendICMPPacket()` in listing 7.4 first creates an ICMP packet, adds a packet sequence number and a time stamp. Secondly the function copies each byte of the current audio buffer to the packet, creates a checksum and finally sends it to the desired destination with the `sendto` command.

The receiver – as shown in listing 7.5 – works vice versa by listening to the local ICMP endpoint socket. It is as well implemented in a separate thread, which firstly executes the `recvfrom` call and waits until a packet arrives. Once it has read a packet, it interprets this packet as an IP packet in order to retrieve the IP header length. According to this information the new ICMP packet is casted into the correct format. Secondly it observes the received packet in terms of a valid echo reply of a remote destination. If this is the case the current system time is subtracted from the reflected time stamp. This way the roundtrip time (RTT) can

```
int ICMPMirror::CreateAndSendICMPPacket(){
  packetToSend.icmpHeader.icmp_type=ICMP_ECHO;
  packetToSend.icmpHeader.icmp_code=0;
  packetToSend.icmpHeader.icmp_seq=packetSequenceNumber;

  gettimeofday(&packetToSend.packetTimeStamp, NULL);

  for (int i=0;i<bytes;i++) packetToSend.audioBuffer[i]=charbuffer[i];

  packetToSend.icmpHeader.icmp_cksum = 0;
  packetToSend.icmpHeader.icmp_cksum=in_cksum((u_short *)&packetToSend,
                       sizeof( packetToSend ) );

  int sent = sendto(socketConnectionToHost,
                          &packetToSend,
                          sizeof( packetToSend ),
                          0,
                          (struct sockaddr*) &hostAddress,
                          sizeof(hostAddress));
}
```

Figure 7.4: Implementation of the ICMP Sender

be retrieved and finally stored in `roundTripTimeOnPacket`. Finally the function forwards the received audio buffer to the subsequent audio network buffer stage via the `sj->rec->process_input(icmpPacket->audioBuffer)` call.

7.2.3 Evaluation

So far, most tests and measurements for network music performances were typically realized with broadband Internet connections [56] such as Internet2 [7] or GEANT2 [5], where jitter typically occurs in a very low dimension [137] – in contrast to DSL networks with lower bandwidths, where delay variations with higher values are more likely. Hence the author consciously chooses one DSL endpoint for the experiments, in order to examine whether the ICMP protocol behaves in the same way as a custom UDP mirror does. The author set up two test peers – one on a home DSL connection in Lübeck/Germany (16 Mbps downstream / 768 Kbps upstream), a second at SARC (Sonic Arts Research Center) in Belfast/Ireland (Gigabit backbone connection). Both peers were equipped with the distributed music application, which is able to generate low delayed audio stream as UDP packets and ICMP packets at any desired latency and bandwidth. With regard to realistic parameter settings [119] for network music performances, the application was configured with the settings outlined in table 7.1.

These settings normally result in a fixed audio packet size of 512 bytes but due to a decimation factor of 8 the block size results in 64 bytes in order to fulfill the DSL

```
int ICMPMirror::WaitAndPrintICMPs(int socketConnectionToHost){
  numberOfBytesReceived = recvfrom(socketConnectionToHost,
                                   pingReplyBuffer,
                                   sizeof(pingReplyBuffer),
                                   0,
                                   (struct sockaddr *) &remoteHost,
                                   sockLength);

  packetInterpetedAsIPPacket = (struct ip*) pingReplyBuffer;
  ipHeaderLength = packetInterpetedAsIPPacket->ip_hl << 2;
  icmpPacket = (struct PingICMPPacket*)(pingReplyBuffer+ipHeaderLength);

  if (icmpPacket->icmpHeader.icmp_type == ICMP_ECHOREPLY){
      gettimeofday(&currentTime, NULL);
      timersub(&currentTime,
                    &(icmpPacket->packetTimeStamp),
                    &roundTripTimeOnPacket);

      roundTripTimeInMS =
        (double) (roundTripTimeOnPacket.sec*1000 +
        (((double) roundTripTimeOnPacket.usec) / 1000));

      sj->rec->process_input(icmpPacket->audioBuffer);
    }
}
```

Figure 7.5: Implementation of the ICMP Receiver

Samplerate	48 kHz
Frame size	256 Samples
Decimation Factor	8
Bitdepth	16 Bit
Audio Channels	1
Network Buffersize	1
Audio Blocking Delay	5.4 ms
Audio Packet Size	64 Bytes

Table 7.1: Audio settings for the UDP/ICMP streaming measurement

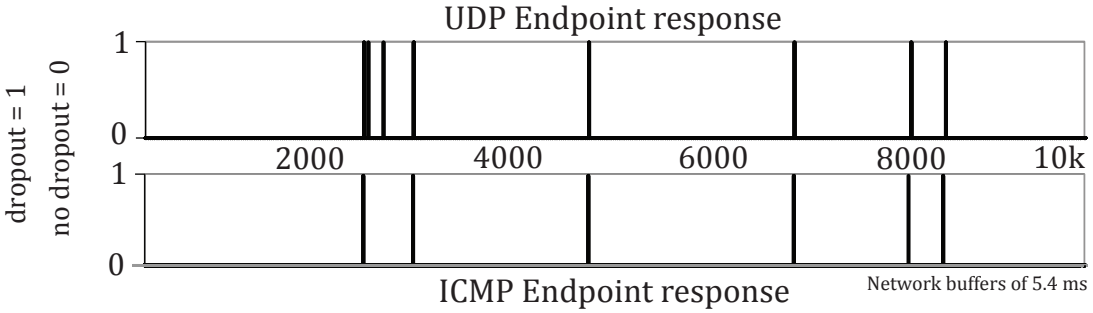


Figure 7.6: Similarity between ICMP and UDP endpoint monitoring

upload restrictions [46]. These audio blocks are sent every 5.2 ms for the sending side and due to only one network buffer, the streaming application passed any received audio buffer directly into the sound card on the receiving end [46]. This small network buffer doesn‘t introduce additional delay and would automatically indicate delay variances in the network stream through the corresponding audio dropouts. Since special emphasis on this effect is desired.

Based on these settings, one UDP and one ICMP stream were sent in parallel to the Belfast machine, which reflected both of them back to the Lübeck machine. The reflection of the ICMP packets was achieved by using an echo request which mirrors an incoming packet back to its sender. Once the streams arrived back at the sending host, the dropouts resulting from network jitter or packet loss were measured. Like in the previous measurements, for packets, which arrived within the required time boundaries led to a reliable audio playback, a “0” was written into a file. When an audio underrun was discovered, a “1” was written into the file. The file was later used as the source for the generation a dropout graph. One minute of streaming audio was generated, which equaled to a number of about 10,000 audio packets. The stream was sent, reflected, received back and measured with the dropout results for each packet shown in the following graphs. Whenever a dropout appeared the graph shows a black line. Due to the relatively large number of packets a single dropout line appears quite thin and the line gets darker as the number of dropouts augments.

The aim of the first measurement was to find out how far an ICMP audio stream corresponds to a reference UDP audio stream in terms of network in general and especially in terms of network jitter. The measurement was performed on Sunday March 9th 2008 at 2:00 am to achieve a low number of dropouts for better visualization and comparison.

The graph in figure 7.6 shows that the behavior of the ICMP stream almost equals the one of the UDP stream. There is a slight variation in the amount of dropouts but the times of dropouts are precisely the same. As an exception to this rule, the UDP response graph shows an additional dropout line at the second position from the left. The author concluded that this small inconsistency was caused by slight variances

hop number	hop IP address
1	89.246.3.71
2	213.30.195.181
3	213.248.77.105
4	80.91.251.81
5	80.91.254.219
6	80.91.252.10
7	213.248.104.154
8	146.97.33.17
9	146.97.42.182
10	195.194.169.250
11	143.117.0.9
12	143.117.254.33
13	143.117.255.162
14	143.117.78.184

Table 7.2: Hops between Lüebeck and Belfast

of the network load situation. Physically the sending events of the UDP and ICMP packets are not exactly simultaneous but rather occur in a sequential order. Therefore, slight variances are plausible. Despite these slight inconsistencies, the results show clearly that the ICMP behavior closely relates to the UDP dropouts. Hence, in order to measure potential sources of dropouts the ICMP echo request could be reflected by any router on the way to the Belfast destination. As a first step the author resolved each host by a traceroute, which indicated the hops on the route shown in table 7.2.

As previously mentioned, the existence of a host on the Internet is commonly checked with ICMP echo responses, typically implemented by the ping command. However, in terms of special network security, network administrators sometimes block ICMP echo requests on some hosts. If this is the case, neither an ICMP request or audio stream would function in the described way. Therefore, as a second step the author had to find, which of the involved hops had ICMP echo response enabled. By applying the ping command, it turned out that except for the hop numbers 11,12 and 13 all other hops had ICMP echo responses enabled. The third step involved measuring the hops with the same one minute streaming test as with ICMP data in parallel to the endpoint in Belfast. Hop 14 was used as the reference UDP mirror stream. The author started with the closest hop, the DSL router itself, the testmachine was connected to. Due to no additional cross traffic on the router, figure 7.7 shows that between the cable linked endpoint and the DSL router no dropouts appeared. At the same time, the UDP reference stream showed three dropouts caused by jitter on further involved hops.

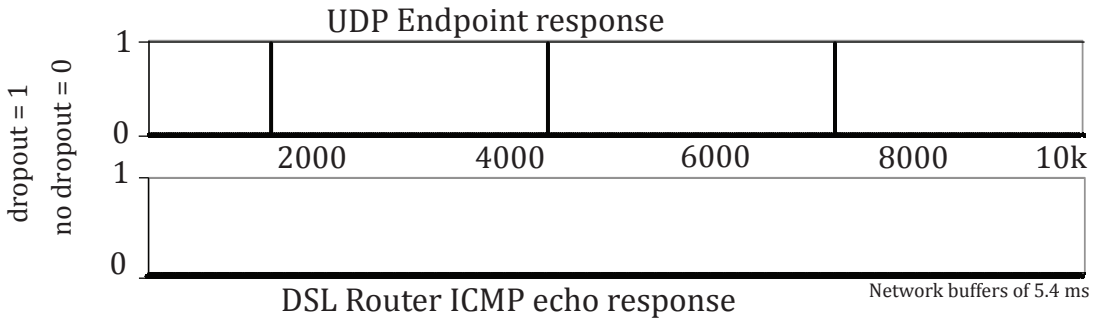


Figure 7.7: ICMP monitoring of hop 1 (DSL Router) in comparison with UDP endpoint monitoring. No dropouts occur on hop 1.

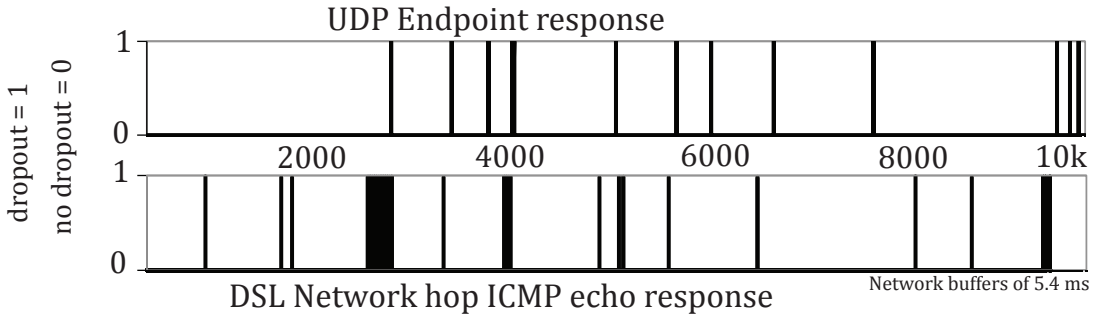


Figure 7.8: Inconsistent measurement results for hop 2

Afterwards hop 2 and 10 were measured respectively. Even though most of the ICMP echo responses were received as expected, in certain intervals for the majority of hops the author experienced some further inconsistencies regarding the dropout behavior. These inconsistencies resulted in strong bursty jitter and additional dropouts for the ICMP stream. The author could observe this behavior even on hop 2, which is illustrated in figure 7.8. This was apparently caused by the ICMP implementation on the reflecting router, which obviously had problems responding to larger amounts of ICMP echo requests.

For hops 6 and 9 the resulting graphs in figure 7.9 and figure 7.10 show the expected and desired behavior. The UDP reference stream indicates dropouts at the same time with similar values. Additionally the author can observe that the results in figure 7.10 indicate more dropouts for the ICMP stream than in figure 7.9 due to a larger hop count. The results of the remaining hops exhibited the same expected dropout characteristics respectively.

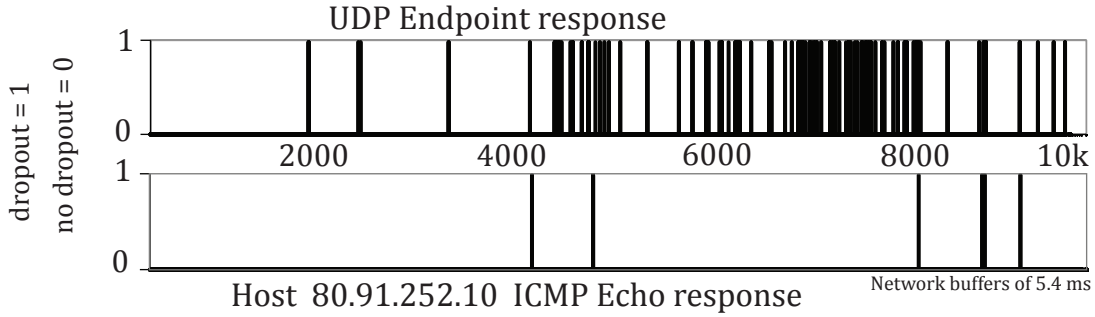


Figure 7.9: Consistent measurement results of hop 6

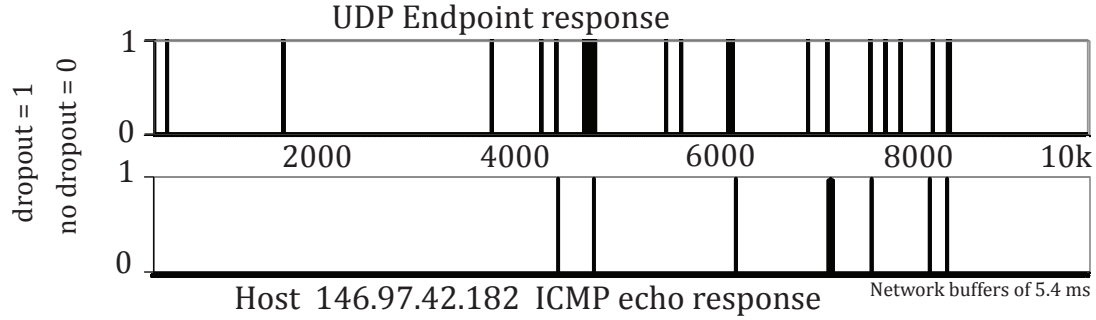


Figure 7.10: Consistent measurement results of hop 9

7.2.4 Conclusion

Though the ICMP protocol is mainly intended for network measurement and normally delivers technical information such as delays and error messages in a numerical representation, it can also be used for the transportation of audio data. The direct comparison shows, that in terms of packet delivery time and dropout behavior in the example setup, an ICMP stream behaves almost precisely in the same way as a UDP stream does. However, in theory ICMP traffic might be routed completely differently to UDP traffic. This practical experience suggests that this approach works in most test cases.

In terms of low delay audio streams, network jitter represents the most significant problem. It can casually appear on any involved hop but unfortunately it has so far been impossible to determine the actual location. The principle of audible ICMP echo responses now offers the possibility to use such a mechanism to reflect audio streams and in turn retrieve information about the current jitter for the specific network segment. In that context it became clearly audible and visible that in case a delay variance occurs on a specific hop, it also appears on the endpoint hop. Though jitter is a casual phenomenon, which only appears with a certain probability depending on the actual network conditions, it became obvious that the jitter increases with the number of hops. Each hop represents a network segment with its own probability for network jitter, so that the sum of probabilities results in the endpoint jitter probability. However, in some cases, the measurements are not coherent to what the author so far stated in this experiment. Firstly the values of dropouts differ significantly. Secondly dropouts appear in the ICMP stream which do not exist in the UDP stream. These exceptions can be explained by the fact, that each stream packet travels the route twice – to the destination and back – and since the network situation might have changed slightly after a packet has reached the reflector, different values can be expected. Due to this effect, there is no need to expect a total correlation of both ICMP and UDP streams, in order to prove the efficiency of ICMP audio streams. Altogether the audible ICMP echo responses are a very practical and convenient approach for monitoring and debugging specific network segments in the context of network music performance.

7.3 DFA reflector

Considering alternative compromised concepts of real musical interaction one could apply MSA, LBA or LAA by simply changing the musical approach. Here no technical aspect has to be taken into account. However, applying DFA represents a more complex case as it implies the principle of a self delay. On one hand it would require additional delay processors and on the other hand one typically has no awareness about the precise values, or at least the dimensions, the self delay has to be adjusted with. In fact, an ideal case represents a situation, in which one musician listens to his

own signal created via the audible feedback of the speakers and microphones of the remote side, while the local signal is muted to zero. However, this setup generates issues related to the involved audio engineering: Since the delayed musician listens to the same signal as the remote musicians, an independent volume control cannot be achieved. Due to different environmental conditions, these levels might be totally different and in turn at least one side might end up in an unsatisfying situation. Additionally, due to different room impulse responses the delayed musician may not want to receive his own signal with such reverbation. As a consequence this loop could alternatively be generated via a manually adjusted aux channel [66] at the remote side's mixing console. Although this improves the described situation compared with a feedback made of speakers and microphone, it requires a more complicated audio setup, more noise due to the respective analogue/digital conversion and furthermore results in a limited flexibility of the delayed player. In this situation one depends on the remote side's sound engineers and hence cannot independently and spontaneous change the desired levels.

7.3.1 Concept

In order to approach the problems related to the establishment of delayed feedback sessions, the new concept should firstly imply an automatic software delay generation without additional external devices. Furthermore, a new solution should produce the same precise delay values as a real feedback loop does but should equally provide the delayed player with the ability to influence its delays independently and spontaneously without influencing the remote side's signal. As any signal treatment after the sound card playout process automatically involves the remote side's sound engineer's settings and actions, a new approach requires respective signal treatment within the software and thus before the analogue signal conversion. With respect to the author's current implementation of a low-latency audio streaming application, the last instance before the sound card playout is represented by the callback function, at which the respective audio buffer is available for any kind of signal treatment or modification. As the callback function hosts the currently played signal to be sent to a remote destination, and equally the currently received audio buffer of an external host, it provides the desired functionality for the approached goal: One one hand the callback function can assign the received buffer to the standard sound card playout as required but can additionally mix this buffer with the outgoing signal before transmitting it to the remote destination. This way the remote player receives the sum of signals, while his own signal is generated by a true and fully digitally processed signal loop.

Although the real feedback loop provides an ideal case scenario, the new cognitive findings of chapter 3 have to be taken into consideration, which furthermore allow a minimization of the roundtrip delayed feedback with a certain offset. Hence, after the establishment of to this real signal loop generated by the previously explained mixing process, the user should have the ability to manually modify his own delay. As the

true feedback loop is bound to the physical conditions, an additional method has to be applied. In that context the user should be able to delay his own local playout by an individually adjustable number of buffers and in turn create a locally produced delay. In a first step the user should adjust this delay in order to match the latency of the true feedback loop. Based on this match in latency settings, he can subsequently lower the local delay according to the maximal offset.

7.3.2 Implementation

In order to achieve an independent volume setting, the remote's callback function needs to know the level, at which the signal has to be mixed. As this level does not match the manually adjusted playout level at the remote side, the transmission of an additional volume parameter is required. Hence, besides the packet sequencing, the transmitted buffer needs to reserve another byte to host the respective values from 0 to 10 – equally to the remaining volume faders. A 0 indicates no mixing and hence no feedback for the remote's side, values between 1 and 9 will reflect the signal with a respective damping of 10 % to 90 %. A value of 10 will lead to an unmodified signal reflection. Although this volume parameter requires to be updated as quickly as possible, it does not necessarily need to be contained in every audio block. As the transmission with every first block of a packet sequence would lead to additional volume update delays of less than 500 ms, such solution would suit the requirements and would, in turn, save transmission bandwidth. The method `sender::settingsSender()` (as illustrated in figure 7.11(a)) is called with each first packet of the numbered sequence, that is packet number 0. Here the byte addressed with the reflect position of the settings-array is assigned with a number depending on the adjusted volume setting from 0 to 10. In another stage this settings-array is finally attached to the sent audio buffer and transmitted.

As the receiver is aware of the fact that packet number 0 contains this addition, it reads the respective parameter from the received packet and assigns it to the `reflectVolume` variable as shown in figure 7.11(b). Finally, within the callback function, the received audio block is added to the currently captured block with the current `reflectVolume`. This process is illustrated in figure 7.11(c). The resulting `sendBuffer` is sent across the network and the feedback loop is completed.

7.3.3 Evaluation

In order to evaluate the DFA reflector solution, two machines were connected to a local area network (LAN) processing a bidirectional audio stream of 1 channel 48 kHz at 16 Bit resolution. The audio block size was adjusted to 128 samples with the corresponding delay of 2.7 ms. One machine had the DFA reflector enabled, which in turn muted the local signal until it was reflected back by the remote side's callback function. Altogether 54 seconds of audio was sent and lead to a total amount of 20,000

```
void sender::settingsSender(){
  switch(sendReflectVolume){
    case 0:  settings[reflectPosition] = 'A';
    break;
    case 1:  settings[reflectPosition] = 'B';
    break;
    ...
    case 10: settings[reflectPosition] = 'K';
    break;
  }
}
```

(a) Settings sender function

```
void receiver::changeSettings(){
  switch(inputPacket[reflectPosition]){
    case 'A': reflectVolume = 0;break;
    case 'B': reflectVolume = 1;break;
    ...
    case 'K': reflectVolume = 10;break;
  }
}
```

(b) Setting the reflect volume in the receiver

```
for ( int i=0; i<playoutBytes; i++ ) {
  if (datahere){
    my->playbackBuffer[i] += my->receivedBuffer[i] * volume;
    float reflectValue = my->reflectVolume / 10;

    if (reflectValue > 0){
      sendBuffer[i] += my->receivedBuffer[i] * reflectValue;
    }
  }
}
```

(c) Receiving and forwarding of reflector data in the audio callback function

Figure 7.11: Sending, receiving and forwarding of reflector data

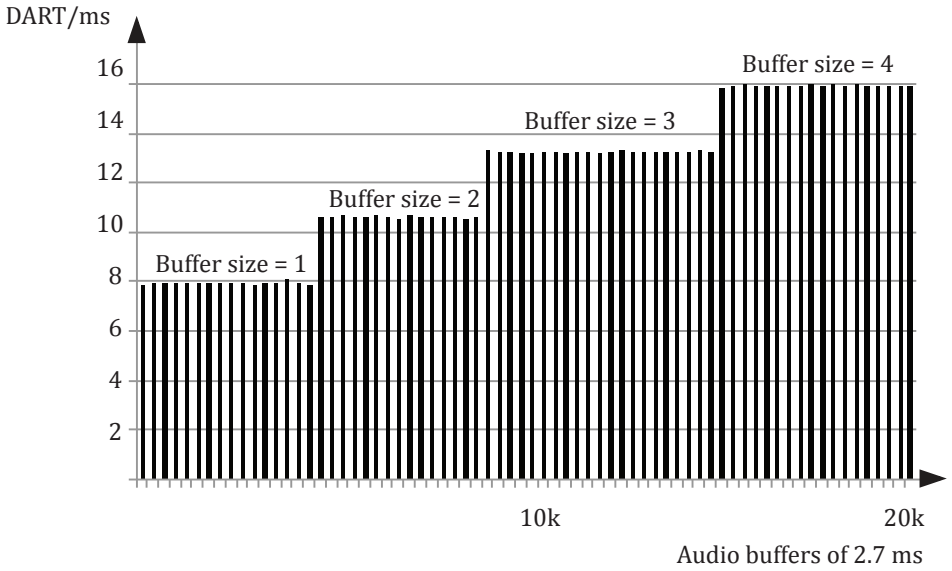


Figure 7.12: Evaluation of the DFA Reflector principle: Results of the "Digital Audio Roundtrip Time" (DART) in a LAN with a stepwise increased network buffer on the remote host

packets. The roundtrip time for each packet was written into a file and finally plotted into the graph as shown in figure 7.12. At the beginning each side's network buffer was adjusted to the minimal value of 1 buffer, which would result in a theoretical delay of $2 \cdot 2.7$ ms $= 5.4$ ms. However, the graph shows a value of 8 ms, which corresponds in a total amount of of 3 network buffers. The additional delay of 1 buffer occurs due to the fact that the local sound card has to wait one period, until it can process the previously sent buffer. Hence, even if no network buffering was applied, this additional delay of 1 buffer would have to be taken into account. Furthermore, the author used the term "Digital Audio Roundtrip Time" since the processing explicitly happens in the digital domain. After approximately 13 s the remote side's network buffer was manually increased by one buffer and lead to the corresponding delay increase of 2.7 ms for every additional network buffer respectively. The final value of 16 ms corresponds to 1 network buffer at the local side and 4 buffers at the remote side.

In addition to this laboratory measurement, the author applied the DFA reflector within a workshop at the Artech conference in Porto, Portugal [34]. First the author established a private test session with the city of Lübeck, Germany. In Porto the author played bass guitar connected to a 10 Mbps link. In Lübeck a grand piano player and the drummer were connected with a 512 kbps A-DSL connection. In terms of the audio engineering the author instructed the remote side in Lübeck to remain passively and not to modify any levels or settings. Their task was just to

play and let the author apply any audio engineering. The buffer adjustment for a stable audio stream resulted in a total audio latency of approximately 40 ms, which lead to disturbing slow-down effects. Since the drums and the grand piano generated too much direct noise the author decided to apply the DFA reflector for his electric bass guitar. As a result the author had to play with a self delay of 80 ms, while the remote side played under normal conditions without any artificial latency. Apart from the inconvenience of the 80 ms self delay it was obvious that the DFA reflector worked in the expected way: The remote callback mixed the incoming bass guitar signal and the local signal with the manually adjusted level and in turn lead to the desired signal reflection. After the author got used to the delayed feedback and after the ideal volume adjustment the remote players did not remark any difference to a regular band rehearsal in the same room.

This test was followed by a demonstration in front of an audience of 200 persons: Again the author applied the DFA reflector for himself connected to an acoustic guitar player and a singer in Rambouillet, France. The total one-way latency resulted in 25 ms and hence the total roundtrip delay resided at 50 ms. In the second set the author performed with a piano player and a saxophone player in Belfast, Northern Ireland. In this case the total one-way latency was 30 ms resulting in a roundtrip delay of 60 ms. In both cases the performance could be considered as extremely good with a stable and tight rhythmical interaction.

7.3.4 Conclusion

The DFA reflector solution represents a convenient solution for applying an automatic delayed feedback for one player if the overall latency is too high for RIA conditions. Since both callback functions are directly interconnected, any change of the signal delay is automatically taken into account and hence results in a solid signal delivery with the precise roundtrip delay. Since the DFA reflector solution allows the player to adjust any levels by himself without influencing the remote side's settings, a maximal convenience and flexibility for each of the involved players can be achieved without any action of the local or remote sound engineer.

7.4 External sound card synchronization

As described in the fourth chapter, sound cards can be clocked with a set of fixed sample frequencies. The reference for these frequencies is generated by the card's internal quartz, which feeds an attached PLL (phase locked loop). The PLL output's quare signal refers to the term wordclock, which triggers the respective conversion processes. Two sound cards, however, inherently suffer from a slight wordclock drift. This can be eliminated by feeding one card's external wordclock input with the other card's clock. Since, due to numerous reasons it is not possible to transmit the clock in

wide area networks (WAN), exact synchronization requires a direct cable connection. Hence, the author investigates a new solution, which provides precise remote sound card synchronization via a novel frequency comparison and adjustment method.

7.4.1 Problem and conventional approaches

Even if two interconnected sound cards are configured with the same sampling frequency of e.g. 48 kHz, they do not run in precise synchrony: Due to slight frequency drifts of the cards' quartzes, practically one card runs slightly faster and in turn it sends more data then the slower card expects. Vice versa, the slower card sends less data than the faster card expects [114]. Hence, in certain intervals – depending on the amount of clockdrift – this results in a buffer underrun for the faster card and a buffer overrun for the slower card. This so-called wordclock drift can theoretically be eliminated by feeding one card's external wordclock input with the other card's clock signal, which consists of a square pulse with the local quartz's frequency. This solution, however, assumes a direct cable link between the cards since the square PLL trigger pulses have to occur in precise intervals of only 20.83 μs with respect to the example frequency of 48 kHz [80]. Hence, in the synchronous telephone network, which works with a sample rate of only 8 kHz instead of 48 kHz, a third instance – the network's 8 kHz clock – is used as a time reference for both sound cards in order to provide the desired sound card synchronization. In asynchronous wide area networks such as the Internet, however, such central instance does not exist and it is furthermore not possible to reliably transmit one card's wordclock signal on this network: Due to the asynchronous transmission principle, the sent data packets underly the effect of network jitter. Moreover, even the transmission of an 8 kHz wordclock with the respective packet transmission intervals of 125 μs would by far exceed the expected amount of packets the network was designed for: Each packet is associated with a so-called overhead, which results in an additional number of transmitted bytes and would in turn lead to a significant bandwidth increase. As a consequence the conventional VoIP principle avoids a precise sound card synchronization and applies a different approach: First in order to prevent the jitter and overhead problem, it sends blocks of a fixed number of collected samples (e.g. 1024 samples per block) and buffers them in a network buffer instead of transmitting and receiving the data in a sample-by-sample manner. Secondly, in order to overcome the clockdrift problem, it artificially increases the amount of sample values in the faster machine's network buffer and reduces them in the slower machine's network buffer by a resampling process [114].

Again the domain of distributed music represents a special application in the field of Internet realtime traffic: It requires to deliver audio streams with a minimum of delay at a maximum quality. Hence, the network buffer has to be adjusted as low as possible. In that context the wordclock drift has a significant impact as it would more frequently lead to a buffer underrun and overrun respectively. Moreover, the

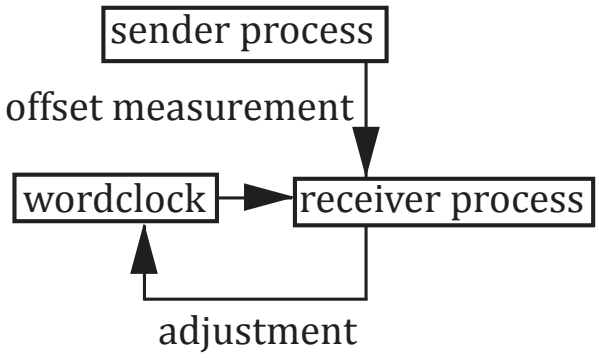


Figure 7.13: Basic principle of the clockdrift adjustment

previously described resampling process has significant drawbacks in this special low delay case: The artificial insertion or deletion of single sample values in the network buffer leads to a slight quality impairment and furthermore breaks the strict block based audio processing principle: As a worst case example with a network buffer of one single audio block, a reduction of the actual sample block by a number of samples would lead to insufficient data for the actual sound card process. Vice versa, an insertion of samples would immediately lead to a buffer overrun. As a consequence the resampling principle does not match the low-delay and high-quality requirements of the distributed music domain: On one hand it assumes a certain network buffer size and in turn prevents to work with minimal network buffer latencies, while on the other hand the conscious sample buffer modification leads to a decreased signal quality. Hence, a solution is desired, which provides a remote sound card synchronization, while keeping up with a strict block based audio processing.

7.4.2 Concept

Rather than applying the common sample buffer modification, the author's concept works with an input signal timing analysis and compares it with the timing of the local sound card. With every sound card callback a time measurement is started and stopped, once the network interface notices an incoming packet. Due to the previously described PLL drift, the measured time gap increases or decreases depending on the amount of PLL inaccuracy between the two clocks. Finally, according to this measurement, the author slightly adjusts the local sound card's wordclock until the time gap remains constant at a desired value. This adjustment is currently realized with a frequency generator as the local wordclock, whose frequency is controlled by the application via an RS232 interface. This adaption process is illustrated in figure 7.13. It is performed continuously in order to address the problem of clockdrift caused by changing quartz temperatures etc..

This principle, however, can only be applied if the network link exhibits a low amount of network jitter. Otherwise – due to the inherent packet delivery variations – a time measurement between two subsequently received packets would rather reflect the amount of network jitter than the actual clock drift and could in turn not be considered as a useful number. Hence, the measuring algorithm takes this issue into account and eliminates it as described in the following section.

7.4.3 Realization

The described sample rate adjustment assumes a correctly applied comparison measurement between the local sound card process and the remote sound card process. Due to the fact that a distributed system cannot hold one and the same clock, this comparison measurement must be processed on a single host. In turn the author declares one host as the "master" and one host as the "slave". The master sends an audio stream to the slave, where the comparison measurement is realized and the local sound card will be adjusted accordingly. The measurement reference is the slave's audio callback function, which is triggered in constant intervals each time the sound card has captured an input buffer and expects to write data to the card's output buffer. The interval depends on the adjusted sample rate and block size. With each callback a time measurement is started via a conventional "getTimeofDay" system call, which stores the system's current time with microsecond accuracy. Another "getTimeofDay" measurement is applied at the slave's network socket. This socket is attached to a function, which is triggered each time the socket receives a network audio packet from the master and hence represents the master's callback intervals. Subtracting the audio callback measurement value from the network socket measurement value, finally describes the actual offset between the two trigger moments. If this offset remains equal over time we can conclude that both sound cards run in precise synchrony at precisely the same frequency. However, if this time gap increases over time, the local sound card runs faster than the remote card – if it decreases over time, the local sound card runs slower than the remote card. In that context the offset can take a value between zero and the maximal sound card interval as e.g. 5.4 ms, which corresponds to a sample rate of 48 kHz and a block size of 256 samples. Once the minimal or respective maximal extreme values are reached, the measurement would immediately exhibit the opposite extreme value in order to continuously repeat the same progression. If the measurement results were plotted on a graph, it would show the sawtooth characteristics of a frequency, which rises or falls depending on the amount of drift between the two cards. Figure 7.14 shows this characteristics of a measurement performed on a LAN between the author's reference setup – a MacBook Pro with an RME Fireface 400 sound card [30] – and a PC with an onboard sound card. The audio stream was sent with a sample rate of 48 kHz and a block size of 256 samples.

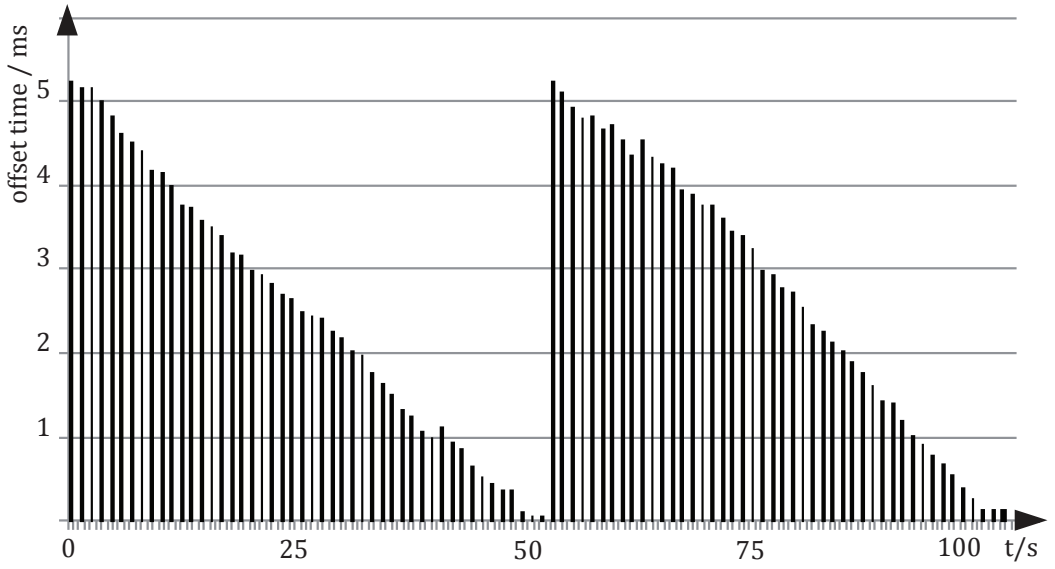


Figure 7.14: Measured clockdrift sawtooth characteristics in a LAN without frequency adjustment

According to the measured frequency drift the author applies the local sound card's sample rate adjustment in such a way that the local wordclock is first initialized with the default frequency of precisely 48 kHz. This is realized via a USB to RS232 converter, which sets the frequency of a Hameg frequency generator. The output of the frequency generator is determined to a square pulse signal of 4 V_{pp} as the reference wordclock signal. Subsequently, the author observes the measured offset in fixed intervals of 1 second, which he considers as sufficiently long enough in order to indicate the amount of drift. If the offset between two subsequent values decreases, the local sample rate is stepwise increased until the offset remains constant. If the offset increases, the local card must be sped down by stepwise decreasing the local wordclock frequency. This monitoring and adjustment process happens continuously since environmental changes such as temperature variations can also lead to a modified wordclock frequency over time.

Nevertheless, this principle only works in local area networks (LAN), which either do not, or just slightly, suffer from the effect of network jitter, which describes the delay variation of packets on an asynchronous network link. Such variations can range in dimensions of milliseconds [147]. Since the measurement determines timing offsets with a maximal range up to 500 μs, the jitter would result in useless values. In order to avoid this effect the author applies a time stamp to each audio packet, which is extracted by the receiver and compared with its local time. This happens for a certain amount of packets in order to determine the average packet delivery time. Based around this calculated average transmission time the author applies a total acceptance range of 1 ms. Only if a packet is received within this average

transmission range, we can conclude that it has hardly suffered from a delay variance on their path from the master to the slave. Furthermore, the author takes advantage of the fact that the intended packet arrival interval without jitter is a known number: As described previously, with a samplerate of 48 kHz and a block size of 256 samples, the network socket would notice a packet arrival each 5.4 ms. Hence, the author applies a second measurement in parallel, which retrieves the "inter packet arrival time". If this time equals or is about the intended blocking interval, we can conclude that both packets arrive in the correct interval time.

If these two preconditions are fulfilled the current network packet can be used to process a valid offset measurement. Otherwise the measurement will be postponed until the next valid network packet has arrived. In case of strong network jitter, the author additionally calculates an average offset value of the last number of measurements in order to eliminate further error potential. Finally, in parallel with the retrieval of a measured number, the wordclock frequency is adjusted higher or lower with a resolution of 0.25 Hz until the offset remains constant.

7.4.4 Evaluation

The measurement of the previous section showed the effect of clockdrift between two 48 kHz sound cards configured with a block size of 256 samples on a LAN. The result is the expected sawtooth characteristics. Figure 7.15 illustrates the wordclock drift after applying the author's frequency adjustment algorithm: At the beginning the graph exhibits the same characteristics, however, with an increasing number of measured values, it approaches a constant value, which finally indicates the desired precise wordclock synchronization. After this steady value had been reached, the frequency generator showed an adjusted frequency of 48,003.9 Hz and makes the author conclude that the clockdrift ranged at 3.9 Hz.

In order to evaluate the author's solution in a WAN, the same experiment was performed between two peers in Lübeck/Germany (peer A: A-DSL link, peer B: University backbone link) – both equipped with the distributed music application [50]. The route between them consisted of 15 hops and a total length of about 1,200 km. Since the author decided the DSL-peer sound card to be the reference or "master" card, he applied the proposed solution on the university "slave" peer. First the measurement was performed without the described jitter compensation technique. The result is illustrated in figure 7.16, in which the effect of the network jitter is clearly visible: Although one could guess the sawtooth shape, the graph exhibits strong measurement errors due to the high amount of delay variation [51].

In the next step the author measured the actual clock drift with the described jitter compensation: The measurement lead to the graph illustrated in figure 7.17. Subsequently, the author turned the wordclock adjuster on and plotted the values respectively. The results are displayed in figure 7.18, which – apart from a rougher

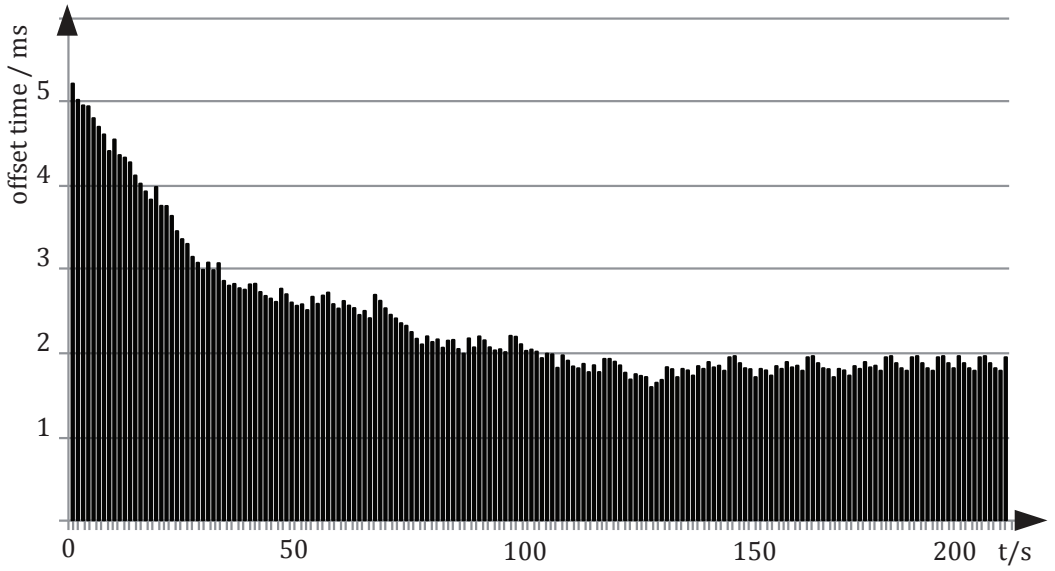


Figure 7.15: Adjusted clockdrift characteristics in a LAN

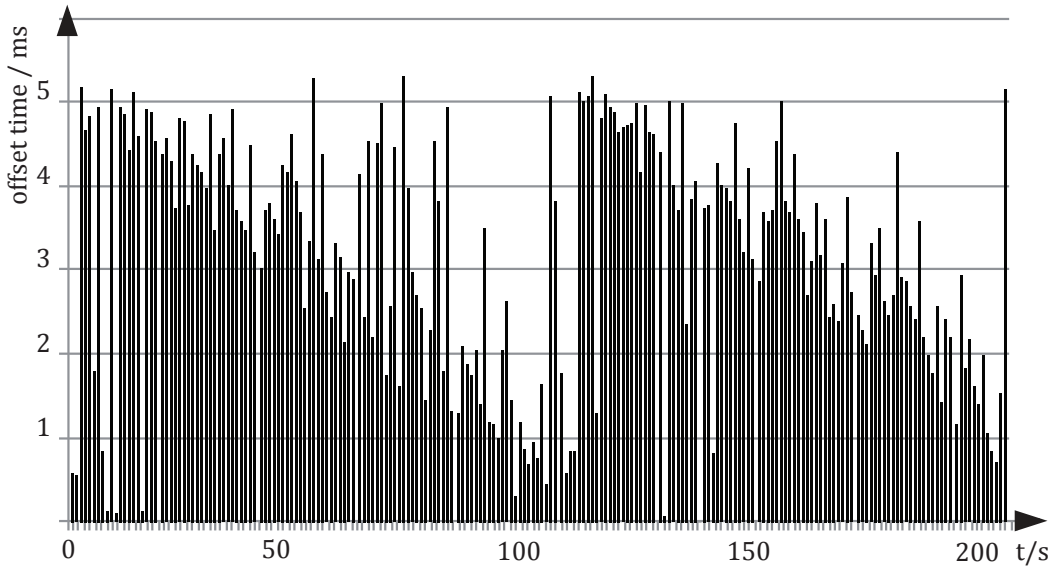


Figure 7.16: WAN measurement without jitter compensation

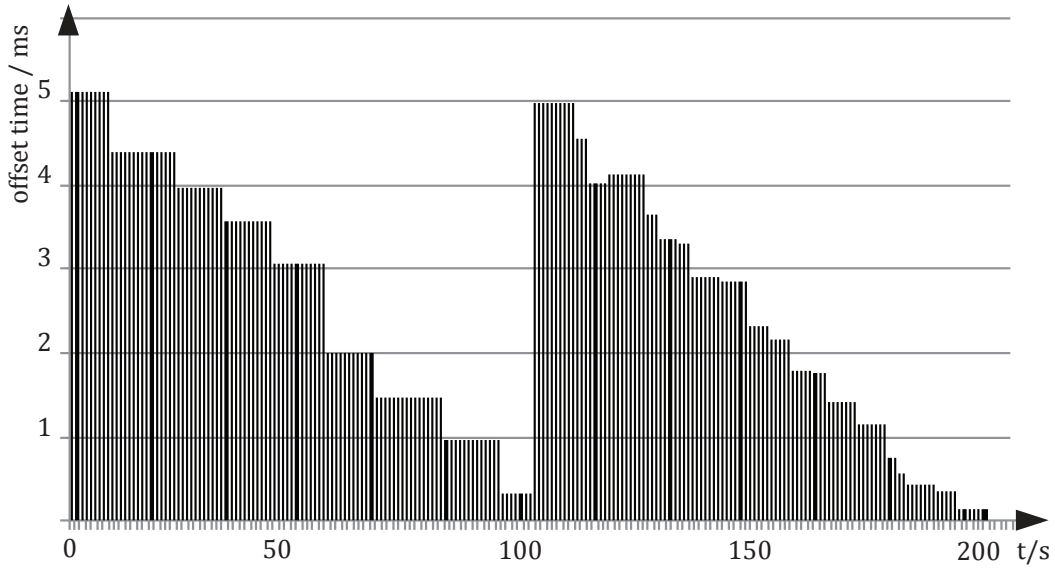


Figure 7.17: Measured characteristics in the WAN setup

timing value resolution – exhibits the same progression as the LAN measurement did. In this case the slave card was adjusted from 48,000 Hz to 48,002.2 Hz and indicated a drift of 2.2 Hz between the remote machine and the reference setup [51].

7.4.5 Conclusions

The external sound card synchronization approach represents a novel solution, which is able to adjust two sound cards' workclocks with the same precise frequency. In contradiction to conventional clockdrift compensation techniques, the author's solution provides the maximal signal quality by keeping up with a consequent block based audio processing, which does not require any artificial sample buffer modification. This on one hand prevents the under- and overruns and, on the other hand, allows for the adjustment of a minimal time gap between a network buffer arrival and the respective buffer pull process. In turn a minimal playback latency can be achieved. Hence, this new solution provides the maximal signal quality at the lowest achievable latency, which fulfills the time and quality critical demands in the domain of distributed music [51]. At the current state the solution works with a frequency generator as the adjustable wordclock device. In future the author will substitute this with a stand-alone PLL circuit.

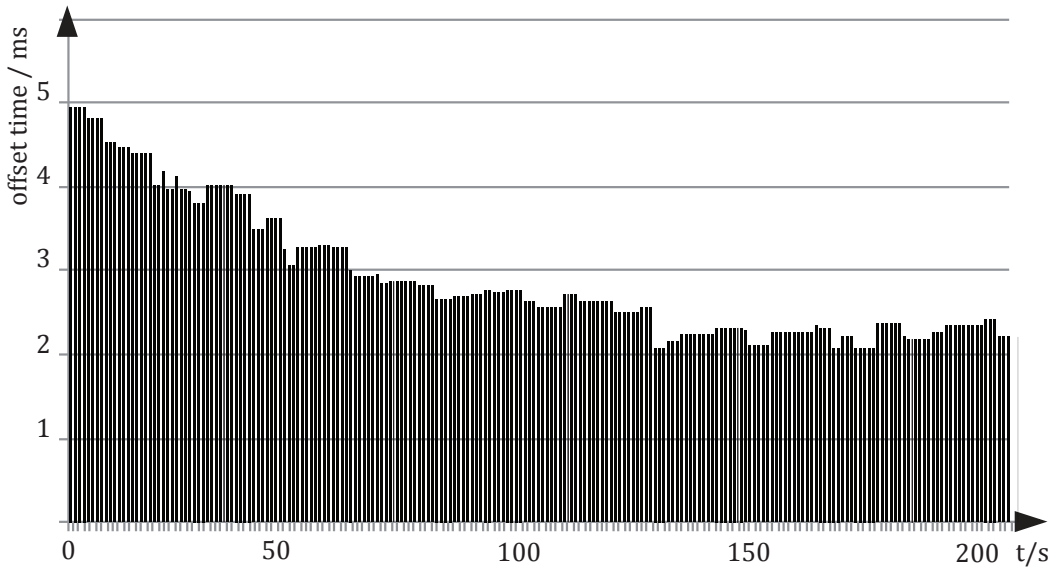


Figure 7.18: Adjusted clockdrift characteristics in the WAN setup

Chapter 8

Discussion and future work

The area of distributed music represents an extremely complex field of interdisciplinary research and development, which embraces aspects of music cognition, sound engineering and computer science. Special aspects of audio engineering, related to the strong cognitive restrictions in terms of latency and quality, make strong demands on a successful network music performance in order to provide musicians with a convenient and playable situation as if in the same room. Although the Internet constantly evolves regarding the delivery of high qualitative realtime traffic, the main issue is still the transmission latency between two music peers.

A convenient latency condition, which matches a situation as in the same room, is difficult to obtain due to a variety of factors. The conductor's propagation speed, the total cable length and transmission capacities of routers and switches determine the total transfer delay. The sound processing and network jitter buffers generate significant additional delays and finally restrict the radius of a playable situation to a maximum of approximately 1,000 km. In that context one must no longer consider the Internet as a location-independent medium as is the case for conventional web services such as web browsing, email and even VoIP. Although the latter can currently be considered as the most representative realtime traffic application, its feasibility is – in contradiction to the domain of distributed music – not determined by the physical distance between two peers. The precise maximal delay and in turn the precise maximal distance musicians can perform with, however, cannot be stated by a precise number: As the result of numerous experiments with professional musicians, this value primarily depends on the speed of a rhythm in bpm and also on the musicians individual delay acceptance limit. This so-called "ensemble delay acceptance limit" (EDAL) corresponds to an individual rhythmical attitude and has the same significance in conventional music scenarios played in the same room as it has on a network. Practically observed values range between 5 ms for fast rhythms up to 65 ms for slow rhythms.

In order to give a musician the possibility to precisely figure the best possible latency to a remote host, the author developed a distributed music system, which is able to stream low delayed audio data on the Internet and allows for the manual adjustment of any relevant stream parameter. Depending on the current network condition

users can individually set the audio frame size and the jitter buffer sizes in order to approach the best compromise between latency and audio dropouts. Hence, unlike in VoIP, dropouts are consciously taken into account as the main goal lies in the reduction of latency. Generally it can be stated that the approach of consciously accepting a certain amount of individually tolerable audio dropouts also conflicts with conventional disciplines in the field of audio engineering, where the achievement of maximal audio quality is the dominating motivation. Nevertheless, disrupted audio signals can be concealed with conventional audio restoration techniques such as audio extrapolation. However, investigations in context with low delayed audio streams lead to the conclusion that on current PC hardware and respective operating systems the existing algorithms are unable to generate such extrapolated audio blocks with the desired quality at the the required speed. Hence, with respect to future work, a major goal lies in achieving a high-quality-extrapolation speed, which remains below the critical minimal audio block duration of 1 ms.

In case a network does not offer enough bandwidth for the delivery of uncompressed audio streams, it is also possible to apply either decimation or lossless compression techniques with corresponding delays below 500 μs but with a low compression factor of 2. Alternatively the developed distributed music system hosts two lossy low delay audio codecs with compression rates of factor 8: The Fraunhofer ULD codec achieves an additional latency of approximately 2.7 ms, while the open-source CELT codec provides approximately 0.65 ms at a sampling rate of 48 kHz. The CELT codec additionally implies a conventional packet loss concealment (PLC), but also in this context numerous practical tests again confirmed, that current PLC techniques are insufficient in terms of low delayed audio streams on the Internet.

Furthermore, the author's latency optimized audio streaming concept hosts measurement techniques for constantly monitoring the actual latency, remote side jitter and total jitter of a reflected test mirror stream. As a special feature, audio blocks can also be sent via the ICMP echo service, which gives the opportunity to use every route's hop as a signal reflector in order to figure problematic network segments. Since this principle does not assume a manually configured UDP reflector on the remote host, it provides a simple and effective test setup. This is of particular interest if a network consists of UDP firewalls and hence prevents the transportation of test streaming data.

However, even if either insufficient sound card latencies, long network routes, or large jitter buffers, result in latencies beyond the ensemble-delay-acceptance-limit (EDAL). The author defined a number of musical interaction categories, which allow a compromized interplay, which either work with the effect of an artificially laid back, avantgardistic music styles without a rythmic reference or with artificial self delays. In any case – even if alternative interaction categories are being applied – latency improvement is appreciated in order to either approach a realistic musical interaction or reduce the effects of compromised interplay such as the amount of self delay or the artificial laid back effect. Consequently the author's streaming solution takes both

aspects into account: On one hand it offers minimal latency by directly accessing the sound card driver and processing data as described above, on the other hand it supports both the single-delayed-feedback approach (SDF) and the dual-delayed-feedback approach (DDF) by applying an adjustable self delay, an automatic high quality feedback loop and a customizable metronome without the need for additional sound engineering setups or devices. In terms of the master/slave approach (MSA), laid-back-approach (LBA) and the latency-appecting approach (LAA), no additional functionality is required as such approaches address the players ability to adapt their musical interaction styles. Hence, the current distributed music system supports each of the proposed categories.

With respect to more than 5 years of experience with the proposed concept of low delay audio transmission on the Internet, the author comes to the conclusion, that – apart from audio engineering, network and music skills – the awareness of delay dimensions and their musical consequences is the main basic requirement for a successful network music performance. Depending on the actual network connection and the respective delays, the user has to consciously apply the suitable category of musical interplay, which finally allows him to perform under any given network situation. In turn this allows him awareness of actual possibilities and limitations in his current situation. However, due to the high amount of interdisciplinary knowledge, distributed music has thus far been used in a limited capacity by a small community of experts in IT as well as in music, so that it cannot be considered as an established technology for musical interaction. Despite the existence of first commercial products, musicians and sound engineers remain passive in terms of accepting and applying this new approach. As the technical facts clearly prove the feasibility of network music performances, with this elaboration, the author hopes to motivate musicians and engineers to take advantage of the musical possibilities distributed music can offer. In the future the author will further investigate in the realistic interaction approach for the Internet in order to increase the radius, in which realistic interaction approach (RIA) can be applied. Since this work has mainly focused on the feasibility of remotely performing rhythmical music such as jazz, pop or rock, in the future the author will focus on conducted classical music. Due to the role of a conductor here a different form of musical interaction is present: Rather than the interplay of a rhythm section, the visual cues form the rhythmical reference of an orchestra. This principle has yet to be examined in depth in terms of a displaced scenario and hence requires additional scientific attention.

All in all the author is convinced that the era of distributed music on the Internet has just begun, and with this elaboration he hopes to overcome the general confusion in this domain by providing a solid scientific basis for future investigations.

Bibliography

[1] Ninjam website, October 2006. http://www.ninjam.com.

[2] Cycling74 website, November 2008. http://www.cycling74.com/.

[3] *Der grosse Brockhaus in drei Bänden.* Brockhaus Verlag, fourth edition, 2008.

[4] Digitalmusician.net website, November 2008. http://www.digitalmusician.net.

[5] Geant2 website, November 2008. http://www.geant2.net/.

[6] German telephone switching center, November 2008. http://www.aufbau-ffm.de/doku/Sonder/fm/Bilder/fern32.jpg.

[7] Internet2 website, March 2008. http://www.internet2.edu.

[8] Quintet.net website, November 2008. http://www.quintet.net.

[9] Secondlife website, November 2008. http://www.secondlife.com.

[10] Skype website, August 2008. http://www.skype.com.

[11] Soundwire research group, November 2008. http://ccrma.stanford.edu/groups/soundwire/.

[12] Steinberg website, November 2008. http://www.steinberg.net.

[13] Ultravideo research group, November 2008. http://ultravideo.mcgill.ca/.

[14] Win labor erlangen website, November 2008. http://www-win.rrze.uni-erlangen.de/ippm.

[15] Alsa website, February 2009. http://www.alsa-project.org.

[16] Encyclopedia britannica, January 2009. http://www.britannica.com/EBchecked/topic/431888/organ-of-Corti.

[17] European space agency website, March 2009. http://www.esa.int.

[18] Musicplant soundstudio, January 2009. http://www.musicplant.de.

[19] Musikhochschule Lübeck website, March 2009. http://www.mh-luebeck.de.

[20] RME Website, February 2009. http:/http://www.rme-audio.de.

[21] Sounddivecenter, January 2009. http://www.sounddivecenter.com/assets/images/Middle-Ear-Pressure.gif.

[22] Steinberg-software website, February 2009. http://www.steinberg.net.

[23] Dirk Van Aken and Sascha Peckelbeen. Whitepaper: Encapsulation overhead(s) in adsl access networks. Technical report, Thomson, June 2003.

[24] Tom Alby. *Web 2.0. Konzepte, Anwendungen, Technologien.* Hanser Verlag, 2007.

[25] Alesis. *ADAT-XT reference manual*, 1995.

[26] Leland I. Anderson. *Priority in the Invention of Radio – Tesla vs. Marconi.* 21st Century Books, 1980.

[27] Gisa Aschersleben. Temporal control of movements in sensorimotor synchronization. *Brain and Cognition*, 48:66–79, 2002.

[28] Anders Askenfelt and Eric Jansson. From touch to string vibrations: Timing in the grand piano action. *Journal of the Acoustical Society of America*, 88:52–63, 1990.

[29] MIDI Manufacturers Association. *Complete MIDI 1.0 Detailed Specification*, November 2001.

[30] RME Audio. *Fireface 400 Datasheet*, 2007.

[31] RME Audio. *Multiface Datasheet*, 2007.

[32] William A.Yost and Donald W. Nielsen. *Fundamentals of Hearing.* second edition, 1985.

[33] Alvaro Barbosa. *Displaced Soundscapes.* PhD thesis, Music Technology Group (MTG) – Universitat Pompeu Fabra, Spain, 2006.

[34] Alvaro Barbosa. *Proceedings of the 4th International Conference on Digital Arts.* 2008.

[35] Ross Bencina. Portaudio and media synchronisation – it's all in the timing. In *Proceedings of the ACMC Conference*, Perth, Australia, 2003.

[36] Günther Bengel. *Grundkurs Verteilte Systeme.* Vieweg Verlag, third edition, 2004.

[37] Klaus Beuth and Olaf Beuth. *Digitaltechnik.* Vogel Verlag, twelfth edition, 2003.

[38] Uyless Black. *Voice Over IP.* Prentice Hall, first edition, 2000.

[39] Rudolf Bock and Werner Krischer. *The Data Analysis Brief Book.* Springer

Berlin, first edition, 1998.

[40] Leon Brillouin. *Wave propagation and group velocity*. Academic Press Inc., 1960.

[41] Donald Broadbent and Peter Ladefoged. Auditory perception of temporal order. *Journal of the Acoustic Society of America*, 31:39–40, 1959.

[42] Wolf Burbat. *Die Harmonik des Jazz*. dtv Verlag, first edition, 1988.

[43] Juan Pablo Caceres, Robert Hamilton, Deepak Iyer, Chris Chafe, and Ge Wang. To the edge with china: Explorations in network performance. In *Proceedings of the 4th International Conference on Digital Arts*, Porto, Portugal, November 2008.

[44] Alexander Carôt. *Livemusic on the Internet*. Diploma thesis, Fachhochschule Lübeck, Germany, 2004.

[45] Alexander Carôt, Torben Hohn, and Christian Werner. Netjack –remote music collaboration with electronic sequencers on the internet. In *Proceedings of the Linux audio conference*, Parma, Italy, June 2009.

[46] Alexander Carôt, Ulrich Krämer, and Gerald Schuller. Network music performance in narrow band networks. In *Proceedings of the 120th AES convention*, Paris, France, May 2006.

[47] Alexander Carôt, Alain Renaud, and Pedro Rebello. Networked music performance: State of the art. In *Proceedings of the AES 30th International Conference on Intelligent Audio Systems*, Saariselkä, Finland, March 2007.

[48] Alexander Carôt, Alain Renaud, and Bruno Verbrugghe. Network music performance with soundjack. In *Proceedings of the 6th NIME Conference*, Paris, France, June 2006.

[49] Alexander Carôt and Christian Werner. Network music performance – problems, approaches and perspectives. In *Proceedings of the Music in the Global Village – Conference*, Budapest, Hungary, September 2007.

[50] Alexander Carôt and Christian Werner. Prinzipien musikalischer Telepräsenz. In *Tagungsband der 25. Deutschen Tonmeistertagung*, Leipzig, 2008.

[51] Alexander Carôt and Christian Werner. External latency-optimized soundcard synchronization for applications in wide-area networks. In *Proceedings of the 14th regional AES Convention*, Tokyo, Japan, July 2009.

[52] Alexander Carôt and Christian Werner. Fundamentals and principles of musical telepresence. *Citar – Revista de ciência e technologias das artes*, 1:26–37, May 2009.

[53] Alexander Carôt and Christian Werner. Verteiltes Musizieren mit Soundjack.

FKT – Fachzeitschrift für Fernsehen, Film und elektronische Medien, 63, 2009.

[54] Alexander Carôt, Christian Werner, and Timo Fischinger. Towards a comprehensive cognitive analysis of delay-influenced rhythmical interaction. In *Proceedings of the International Computer Music Conference (ICMC) 2009*, Montreal, Canada, August 2009.

[55] Chris Chafe, Michael Gurevich, Grace Leslie, and Sean Tyan. Effect of time delay on ensemble accuracy. In *Proceedings of the International Symposium on Musical Acoustics*, Nara, Japan, March 2004.

[56] Chris Chafe, Scott Wilson, Randal Leistikow, David Chisholm, and Gary Scavone. A simplified approach to high quality music and sound over ip. In *Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFX-00)*, Verona, Italy, December 2000.

[57] Elaine Chew and Alexaner Sawchuk. Distributed immersive performance. In *Proceedings of the North American Summer Meeting*, Brown University, Providence, Rhode Island, USA, June 2004.

[58] Martin P. Clark. *ATM Networks – Principles and Use*. Wiley-Teubner, 1996.

[59] George C. Clark, Jr. and J. Bibb Cain. *Error-Correction coding for digital communications*. Plenum Press, first edition, 1981.

[60] Douglas E. Comer. *Computernetzwerke und Internets mit Internet-Anwendungen*. Pearson Studium, third edition, 2002.

[61] Alan B. Coppens, Austin R. Frey, Lawrence E. Kinsler, and James V. Sanders. *Fundamentals of acoustics*. John Wiley and Sons, Inc., third edition, 1982.

[62] Compaq Computer Corporation, Hewlett-Packard Company, Intel Corporation, Lucent Technologies Inc, Microsoft Corporation, NEC Corporation, and Koninklijke Philips Electronics N.V. *Universal Serial Bus Specification*, 2000.

[63] Michael Dickreiter. *Handbuch der Tonstudiotechnik*. Saur Verlag, sixth edition, 1997.

[64] Dieter Eberlein. *Lichtwellenleiter-Technik: Grundlagen, Verbindungs- und Messtechnik, Systeme, Trends*. Expert Verlag, seventh edition, 2007.

[65] Kjeld Borch Egevang and Paul Francis. RFC 1631: The IP network address translator (NAT), May 1994.

[66] Roland Enders. *Das Homerecording Handbuch*. GC Carstensen, third edition, 2003.

[67] F. Alton Everest. *The Master Handbook of Acoustics*. New York: McGraw-Hill, second edition, 2001.

[68] Martin Fischer. *Faszination Schellack*. Battenberg, November 2006.

[69] Bryan Ford, Pyda Srisuresh, and Dan Kegel. Peer-to-peer communication across network address translators. In *Proceedings of the USENIX Annual Technical Conference*, Anaheim, CA, USA, 2005.

[70] Behrouz A. Forouzan. *TCP/IP Protocol Suite*. McGraw-Hill, second edition, 2003.

[71] Anders Friberg and Johan Sundberg. Time discrimination in a monotonic, isochronous sequence. *Journal of the Acoustical Society of America*, 98:2524–2531, 1995.

[72] Xiaoyuan G, Matthias Dick, Ulf Noyer, and Lars Wolf. Nmp – a new networked music performance system. In *Proceedings of the 4th NIME Conference*, June 2004.

[73] Philip Golden, Hervé Dedieu, and Krista S. Jacobsen. *Implementation and Applications of DSL Technology*. Auerbach Publications, first edition, 2008.

[74] Lillian Goleniewski and Kitty Wilson Jarrett. *Telecommunications Essentials*. Pearson Education, second edition, 2007.

[75] Ivo Emanuel Goncalves, Silvia Pfeiffer, and Christopher Montgomery. RFC 5334: Ogg media types, September 2008.

[76] Richard T. Griffiths. History of the internet, internet for historians, August 2008. `http://www.let.leidenuniv.nl`.

[77] W. Lawrence Gulick. *Hearing: Physiology and Psychophysics*. Oxford University Press, 1971.

[78] Georg Hajdu. Quintet.net – a quintet on the internet. In *Proceedings of the International Computer Music Conference*, Singapore, 2003.

[79] Hubert Henle. *Das Tonstudio-Handbuch*. GC Carstensen, fifth edition, 2001.

[80] Florian Hernschier. Digital ist nicht gleich digital – Analyse von Jitterbehafteten Wordclock-Signalen und deren Klangeinfluss auf ein digitales Audiosystem, 2006. Diploma thesis.

[81] Florian Hernschier. Jitter influences onto wordclock synchronisation. In *Proceedings of the 24th Tonmeistertagung*, Leipzig, Germany, November 2006.

[82] Imogen Holst. *Das ABC der Musik*. Reclam Verlag, first edition, 1992.

[83] Bernhard Hommel. The cognitive representation of action: Automatic integration of perceived action effects. *Psychological Research*, 3:176–186, 2004.

[84] Lars Hörmander. *The Analysis of Linear Partial Differential Operators I*. Springer Berlin, second edition, 2003.

[85] Apple Computer Inc. *Audio and MIDI on Mac OS X*, 2001.

[86] International Telecommunication Union (ITU). *Recommendation G.722: General aspects of digital transmission systems – 7 kHz Audio-coding within 64 kbit/s*, 1988.

[87] International Telecommunication Union (ITU). *Recommendation H.323: Audiovisual and multimedia systems – Infrastructure of audiovisual services - Systems and terminal equipment for audiovisual services – Packet-based multimedia communications systems*, 1998.

[88] International Telecommunication Union (ITU). *Recommendation G.114: Transmission systems and media, digital systems and networks – General Recommendations on the transmission quality for an entire international telephone connection*, 2003.

[89] International Telecommunication Union (ITU). *Document: FoV/04: Future of Voice - Status of VoIP*, 2007.

[90] David James. Multiplexed buses: the endian wars continue. *IEEE Micro*, 10:9–21, 1990.

[91] Axel Jungbluth. *Praxis Jazzharmonisation*. Schott Verlag, first edition, 1989.

[92] Gary C. Kessler and Peter Southwick. *ISDN – Concepts, Facilities, and Services*. McGraw-Hill, third edition, 1990.

[93] Dimitri Konstantas, Yann Orlarey, Olivier Carbonel, and Simon Gibbs. The distributed musical rehearsal environment. *IEEE MultiMedia*, 6:54–64, 1999.

[94] Anton Kos, Borut Klepec, and Saso Tomazic. Techniques for performance improvement of voip applications. In *Proceedings of the 11th Electrotechnical Conference MELECON*, Cairo, Egypt, 2002.

[95] Ulrich Krämer, Jens Hirschfeld, Gerald Schuller, Stefan Wabnik, Alexander Carôt, and Christian Werner. Network music performance with ultra-low-delay audio coding under unreliable network conditions. In *Proceedings of the 123rd AES-Convention*, New York, USA, October 2007.

[96] Clemens Kühn. *Musiklehre*. Laaber Verlag, first edition, 1981.

[97] James Kurose and Keith Ross. *Computer Networking: A Top-Down Approach Featuring the Internet*. Addison Wesley, third edition, 2004.

[98] Nelson Posse Lago and Fabio Kon. The quest for low latency. In *Proceedings of the International Computer Music Conference 2004*, Miami, FL, USA, 2004.

[99] Dirk Larisch. *TCP/IP*. Moderne Industrie Buch AG, second edition, 2004.

[100] Bhagawandas Pannalal Lathi. *Signal Processing and Linear Systems*. Oxford University Press, first edition, 1998.

[101] Paul D. Lehrman. Digicon 83. *Creative Computing*, 10:164, 1984.

[102] Burkhard Lenze. *Einführung in die Fourier-Analysis*. Logos Verlag Berlin, second edition, 2002.

[103] Michael W. Levine and Jeremy M. Shefner. *Fundamentals of Sensation and Perception*. Oxford University Press, third edition, 2001.

[104] Markus Lonardoni. *Popularmusiklehre*. Reclam Verlag, first edition, 1996.

[105] Marshal Mc Luhan, Quentin Fiore, and Jerome Agel. *The Medium is the Massage: An Inventory of Effects*. Gingko Press, 1996.

[106] Bill Mallon and Jean-Pierre Caravan. Olympic news notes. *Journal of Olympic History*, 6:1–8, 1994.

[107] Henrique Malvar and David Staelin. The lot transform coding without blocking effects. *IEEE Transactions on Speech and Audio Processing*, 37:553–559, 1989.

[108] Frederick N. Martin. *Introduction to Audiology*. third edition, 1986.

[109] Jiri Mates and Gisa Aschersleben. Sensorimotor synchronization: The impact of temporally displaced auditory feedback. *Acta Psychologica*, 104:29–44, 2000.

[110] Stephen McAdams and Emmanuel Bigand. *Thinking In Sound*. Oxford Science Publications, first edition, 1993.

[111] Beate Meffert and Olaf Hochmuth. *Werkzeuge der Signalverarbeitung*. Pearson Studium, first edition, 2000.

[112] Daniel Molkentin. *Qt 4: Einführung in die Applikationsentwicklung*. Open source press, first edition, 2006.

[113] Brian C. Moore. *Psychology of Hearing*. Elsevier Academic Press, fifth edition, 2007.

[114] Sivannarayana Nagireddi. *VoIP Voice and Fax Signal Processing*. Wiley, first edition, 2008.

[115] Institute of Electrical and Inc. Electronics Engineers. *1394-2008 IEEE Standard for a High-Performance Serial Bus*, 2008.

[116] Jens R. Ohm and Hans D. Lüke. *Signalübertragung – Grundlagen der digitalen und analogen Nachrichtenübertragungssysteme*. Springer Verlag, Berlin, 2002.

[117] Alan V. Oppenheim and Ronald W. Schaefer. *Discrete-time signal processing*. Englewood Cliffs, NJ: Prentice Hall, Inc., 1989.

[118] Franz Pichler. Physikgeschichte. *Plus Lucis*, 26:29–33, January 2001.

[119] Ken C. Pohlmann. *Principles of Digital Audio*. The Mcgraw-Hill Companies, fifth edition, 2005.

[120] Jonathan Postel. RFC 768: User datagram protocol, August 1980.

[121] Jonathan Postel. RFC 791: Internet Protocol, September 1981.

[122] Jonathan Postel. RFC 792: Internet Control Message Protocol, September 1981.

[123] James Pritchett. *The Music of John Cage*. Cambridge University Press, Cambridge, UK, 1993.

[124] Rudolf Rasch. The perception of simultaneous notes such as in polyphonic music. *Acustica*, 40:21–33, 1978.

[125] Jörg Rech. *Wireless LANs: 802.11-WLAN-Technologie und praktische Umsetzung im Detail*. Heise Verlag, second edition, 2008.

[126] Michael Reisch. *Elektonische Bauelemente*. Springer Verlag, second edition, 2007.

[127] Stephan Richter and Thomas Wickl. *Netzwerktechnik*. Herdt Verlag, second edition, 2006.

[128] Wolfgang Riggert. *Rechnernetze*. Fachbuchverlag Leipzig, third edition, 2005.

[129] Thomas D. Rossing, Richard F. Moore, and Paul A. Wheeler. *Science of Sound*. Addison Wesley, third edition, 2002.

[130] Dean Rubine and Paul McAvinney. Programmable fingertracking instrument controllers. *Computer Music Journal*, 14:26–40, 1990.

[131] Eve M. Schooler. Distributed music: A foray into networked performance. In *Proceedings of the International Network Music Festival*, Santa Monica, CA, 1993.

[132] Nathan Schuett. *The Effect of Latency on Ensemble Performance*. Bachelor thesis, CCRMA Department of Music, Stanford University, 2002.

[133] Hans Schulze. The detectability of local and global displacements in regular rhythmic patterns. *Psychological Research*, 40:173–181, 1978.

[134] Seymour Shlien. The modulated lapped transform, its time-varying forms, and its applications to audio coding standards. *IEEE Transactions on Speech and Audio Processing*, 5:359–366, 1997.

[135] Paul Ferdinand Siegert. *Die Geschichte der E-Mail. Erfolg und Krise eines Masssenmediums*. Transcript Verlag, 2008.

[136] Ellen Siever, Aaron Weber, Stephen Figgins, Robert Love, Arnold Robbins, and Lars Schulten. *Linux in a Nutshell*. O'Reilly, fourth edition, 2005.

[137] Rishi Sinha, Christos Papadopoulos, and Chris Kyriakakis. Loss concealment for multi-channel streaming audio. In *Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, Monterey, California, USA, June

2003.

[138] Steven W. Smith. *Digital Signal Processing.* California Technical Publishing, San Diego, second edition, 1999.

[139] Paul Söhner. *Allgemeine Musiklehre.* Kösel Verlag, seventh edition, 1979.

[140] Andreas Spanias, Ted Painter, and Venkatraman Atti. *Audio Signal Processing and Coding.* Wiley-Interscience, first edition, 2007.

[141] William Stallings. *ISDN and Broadband ISDN with Frame Relay and ATM.* Prentice-Hall, third edition, 1995.

[142] AES Standard. *AES Recommended Practice for Digital Audio Engineering – Serial Multichannel Audio Digital Interface (MADI)*, 2008.

[143] Stanley Smith Stevens. On the psychophysical law. *Psychological Review*, 64:153–181, 1957.

[144] Stanley Smith Stevens. Perceived level of noise by mark vii and decibels (e). *Journal of the Acoustic Society of America*, 51:575–601, 1957.

[145] Stanley Smith Stevens. *Hearing – Its Psychology and Physiology.* John Wiley and Sons, second edition, 1983.

[146] Li Tan. *Digital Signal Processing: Fundamentals and Applications.* Academic Press, first edition, 2007.

[147] Andrew S. Tanenbaum. *Computer Networks.* Pearson Studium, fourth edition, 2003.

[148] Michael Thaut, Biao Tian, and Mahmood Azimi-Sadjadi. Rhythmic finger tapping to cosine-wave modulated metronome sequences: Evidence of subliminal entrainment. *Human Movement Science*, 17:839–863, 1998.

[149] Hans-Jörg Thiele and Marcus Nebeling. *Coarse Wavelength Division Multiplexing (Technologies and Application).* CRC Press, 2007.

[150] Jean Marc Valin, Timothy Terriberry, Christopher Montgomery, and Gregory Maxwell. A high-quality speech and audio codec with less than 10 ms delay. *IEEE Transactions on Audio, Speech and Language Processing*, 2009.

[151] Daniel Christian von Grüningen. *Digitale Signalverarbeitung.* Carl Hanser Verlag, third edition, 2007.

[152] Michael Warstat and Thomas Görne. *Studiotechnik – Hintergrund und Praxiswissen.* Elektor Verlag, Aachen, 1994.

[153] Alexander Weber. *Freeware VST/VSTI Audio-Plugin.* GC Carstensen, first edition, 2006.

[154] Stefan Weinzierl. *Handbuch der Audiotechnik.* Springer, first edition, 2008.

[155] Alfred. D. Weiss. *Human Aging: A Biological and Behavioral Study*, chapter Auditory perception in relation to age. PHS Publications, 1963.

[156] Steve Winder. *Telecommunications*. Newnes publications, third edition, 2001.

Lebenslauf des Autors

Persönliche Daten

Alexander Christian Carôt
geboren am 30. August 1974 in Lübeck

Werdegang

08/85 – 04/94	Oberschule zum Dom, Lübeck
06/94	Schulabluss Abitur
08/94 – 08/97	Berufsausbildung zum Radio- und Fernsehtechniker in Lübeck und Frankfurt am Main
09/97 – 09/99	Freier Musiker, Tontechniker und Softwareentwickler
10/99 – 04/04	Studium "Informationstechnologie und Gestaltung" an der Fachhochschule Lübeck
04/01 – 04/02	Studienaufenthalt an der Danmarks Designskole in Kopenhagen, Dänemark
05/02 – 11/02	Studienaufenthalt am CCRMA (Center for Computer Research in Music and Acoustics) in Stanford, USA
04/03 – 10/03	Studienaufenthalt an der MTG (Music Technology Group) in Barcelona, Spanien
05/04	Abschluss Diplomingenieur
06/04 – 02/05	Wissenschaftlicher Mitarbeiter bei Prof. Dr. Alvaro Barbosa, Portuguese Catholic University, in Porto, Portugal
seit 03/05	Nebenberuflicher Toningenieur und Musiker im Popularmusikbereich
03/05 – 12/07	Wissenschaftlicher Mitarbeiter an der International School of New Media in Lübeck
08/07	Anerkennung des Promotionsrechtes der Technisch-Naturwissenschaftlichen Fakultät der Universität zu Lübeck
seit 03/08	Wissenschaftlicher Mitarbeiter am Institut für Telematik an der Universität zu Lübeck
07/09	Promotion zum Dr.-Ing.

Wissenschaftliche Tätigkeit

- 16 Veröffentlichungen in Monographien und Beiträgen in begutachteten Zeitschriften oder Konferenzbänden
- Entwickler der “Soundjack”-Software [48]
- Ausrichter von Workshops zum Thema “verteiltes Musizieren” auf der NIME Konferenz 2006 in Paris (New Interfaces for Musical Expression), AES-Konferenz 2007 in New York (Audio Engineering Society), AES-Konferenz 2008 in Amsterdam, ICMC-Konferenz 2008 in Belfast (International Computer Music Conference) und der Tonmeistertagung 2008 in Leipzig
- Diverse “invited talks” und beratende Tätigkeiten zum Thema “verteiltes Musizieren”