Audio Engineering Society

Convention Paper

Presented at the 14th Regional Convention
2009 July 23–25 Tokyo, Japan

External latency-optimized soundcard synchronization for applications in wide-area networks

Alexander Carôt[1] and Christian Werner[1]

[1] *University of Lübeck, Ratzeburger Allee 160, 23538 Lübeck, Germany*

Correspondence should be addressed to Alexander Carôt (carot@itm.uni-luebeck.de)

ABSTRACT

Soundcards 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 soundcards, 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 it is due to numerous reasons not possible to transmit the clock in wide area networks (WAN), exact synchronization requires a direct cable connection. Hence, this paper investigates a new solution, which provides precise remote soundcard synchronization via a novel frequency comparison and adjustment method.

1. INTRODUCTION

Telecommunication can be considered as a well-established field in research and development as well as in industrial applications [7]. The original analogue telephone network transmitted a voice signal in its time and value continuous form. However, due to numerous drawbacks, the principle of analogue signal transmission became replaced by its digital successor. In the digital domain a signal is first converted from a time and value continuous representation to a time and value discrete representation. This so-called A/D conversion results in a number of signal samples in fixed time intervals, each holding the current signal amplitude. Upon reception each sample undergoes the D/A conversion process, in which firstly each sample's value is assigned with a certain voltage and secondly a convolution with a sinc signal is applied. Both steps finally reconstruct the original audible time and value continuous shape of the signal. In terms of reliability and robustness, telephone providers explicitly used synchronous net-

works until the mid 1990s, which transmit a voice signal sample by sample in their respective guaranteed time slots. Such time slots are directly related to the conventional sample rate of 8 kHz, which corresponds to a sample generation each 125 μs [5]. Nevertheless, although synchronous networks represent the ideal digital signal transmission principle, they appear to be slowly replaced by a competing asynchronous technology – the Internet.

The reasons for transmitting realtime data on the packet based asynchronous Internet are its comparably high bandwidth capacities, its strong worldwide distribution and the possibility to combine conventional voice traffic with video streams and new sociocultural Internet services. Furthermore, due to not existing technical conventions, it is possible to transmit data of any desired sample rate, bit resolution and compression scheme. As a first step, the VoIP principle approached to achieve telephone service qualities comparable with the conventional synchronous network. However, more recently, new applications – often related to music – require better standards in terms of quality and delay. In that context the most demanding field is the domain of distributed music, where musicians expect a maximal audio transmission quality with a minimal latency [4]. As a consequence, a number of aspects related to conventional VoIP cannot be applied in distributed music. In that context, a major general problem exists, which will be described in the following section.

2. PROBLEM AND RELATED WORK

Even if two interconnected soundcards 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 verse, the slower card sends less data then the faster cards expects [8]. 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 [6]. 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 soundcards in order to provide the desired soundcard 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 workdclock signal on this network: Due to the asynchronous transmission principle, the sent data packets underly the effect of network jitter, which describes a packet time delivery variation depending on the amount of additional foreign traffic. 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 soundcard synchronization and applies a different approach: Firstly, 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 does artificially increases the amount of sample values in the faster machine's network buffer and reduces them in the slower machines's network buffer respectively by a resampling process [8].

A special application in the Internet realtime traffic domain is the field of distributed music, which requires to deliver audio streams with a minimum of delay at a maximum quality. In that context professional PC based soundcards as well process data in a blockwise manner instead of transferring each sample directly to the userspace of an operating system. As a result, the area of distributed music exhibits similarities with classical VoIP. However, in order to achieve the desired quality, it applies a sample rate of 48 kHz rather than 8 kHz, and in order to achieve lower delays, it applies lower audio

block sizes down to 64 samples/block rather than 1024 samples/block [2]. With respect to the latter, a distributed music system furthermore requires a network buffer 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 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 soundcard process. Vice verse, 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, this paper presents a concept and a solution, which provides a remote soundcard synchronization, while keeping up with a strict block based audio processing.

3. CONCEPT

Rather than applying the common sample buffer modification, our concept works with an input signal timing analysis and compares it with the timing of the local soundcard. With every soundcard 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, we slightly adjust the local soundcards 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 1. It is performed continuously in order to address the problem of clockdrift caused by changing quartz temperatures etc.

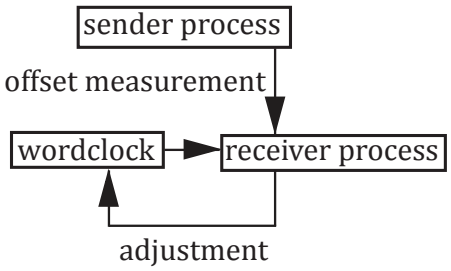

Fig. 1: Basic principle of the clockdrift adjustment

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, our measuring algorithm as well takes this issue into account and eliminates it as described in the following section.

4. REALIZATION

The described sample rate adjustment assumes a correctly applied comparison measurement between the local soundcard process and the remote soundcard process. Due to the fact that a distributed system cannot hold one and the same clock, this comparison measurement must be processed on one single host. In turn we declare 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 soundcard will be adjusted accordingly. The measurement reference is the slave's audio callback function, which is triggered in constant intervals each time the soundcard 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 frame 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 as well attached to a function, which is triggered each time the socket receives a network

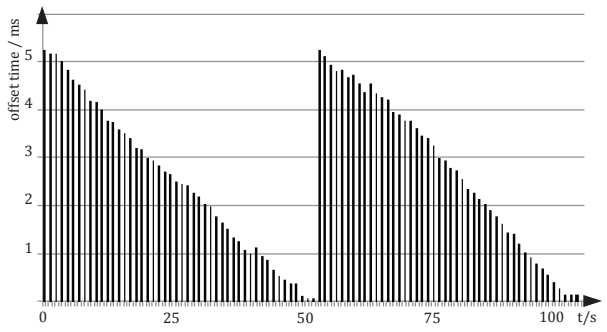

Fig. 2: Measured clockdrift sawtooth characteristics in a LAN without frequency adjustment

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 soundcards run in precise synchrony at precisely the same frequency. However, if this time gap increases over time, the local soundcard runs faster than the remote card – if it decreases over time, the local soundcard runs slower than the remote card. In that context the offset can take a value between zero and the maximal soundcard 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 a sawtooth characteristics of a frequency, which rises or falls depending on the amount of drift between the two cards. Figure 2 shows this characteristics of a measurement performed on a LAN between our reference setup – a MacBook Pro with an RME Fireface 400 soundcard [1] – and a PC with an onboard soundcard. The audio stream was sent with a sample rate of 48 kHz and a block size of 256 samples.

According to the measured frequency drift we apply the local soundcard's sample rate adjustment in such a way that the local wordclock is firstly 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, we observe the measured offset in fixed intervals of 3 seconds, which we consider as sufficiently long enough in order to indicate the amount of drift. If the offset between two subsequent values decreases we stepwise increase the local sample rate 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 as well lead to a modified wordclock frequencies over time.

Nevertheless, this principle does only work in local area networks (LAN), which 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 some milliseconds [9]. Since our 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 we apply 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 we apply an acceptance range of altogether 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, we take 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, we apply 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, we addi-

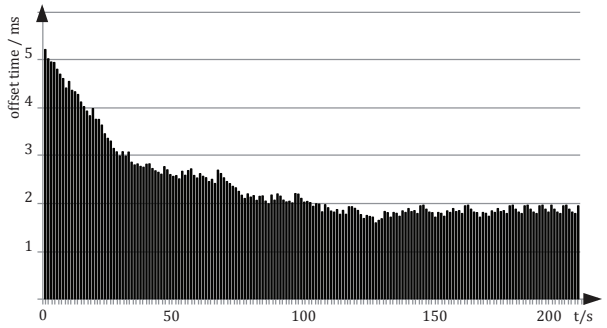

Fig. 3: Adjusted clockdrift characteristics in a LAN

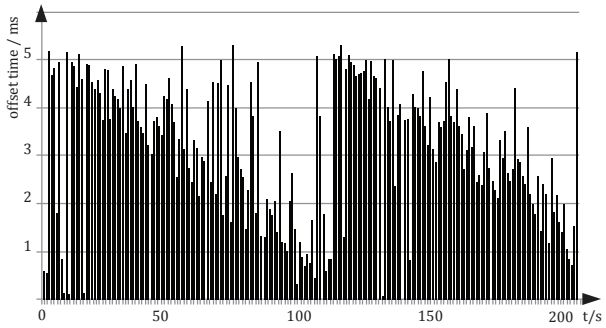

Fig. 4: WAN measurement without jitter compensation

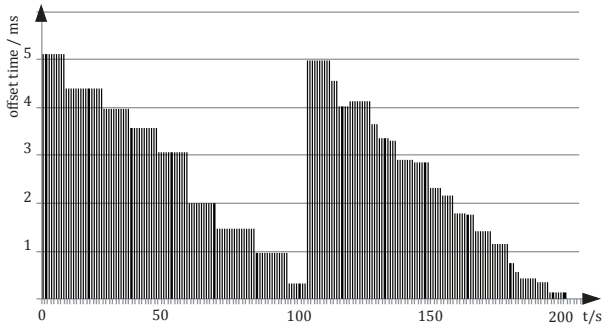

Fig. 5: Measured characteristics in the WAN setup

tionally calculate 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 respectively lower with a resolution of 0.25 Hz until the offset remains constant.

5. EVALUATION

The measurement of the previous section showed the effect of clockdrift between two 48 kHz soundcards configured with a block size of 256 samples on a LAN. The result is the expected sawtooth characteristics. Figure 3 illustrates the wordclock drift after applying our 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 us conclude that the clockdrift ranged at 3.9 Hz.

In order to evaluate our 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 our audio streaming application [3]. The route between them consisted of 15 hops and a total length of about 1,200 km. Since we decided the DSL-peer soundcard to be the reference or "master" card, we applied the proposed solution, on the university "slave" peer. As a first step we applied a measurement without the described jitter compensation technique. The result is illustrated in figure 4, 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.

In the next step we measured the actual clock drift with the described jitter compensation: The measurement lead to the graph illustrated in figure 5. Subsequently, we turned the wordclock adjuster on and plotted the values respectively. The results are displayed in figure 6, which – apart from a rougher 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 indicates a drift of 2.2 Hz between the remote machine and our reference setup.

6. CONCLUSION AND FUTURE WORK

Our external soundcard synchronization approach represents a novel solution, which is able to adjust

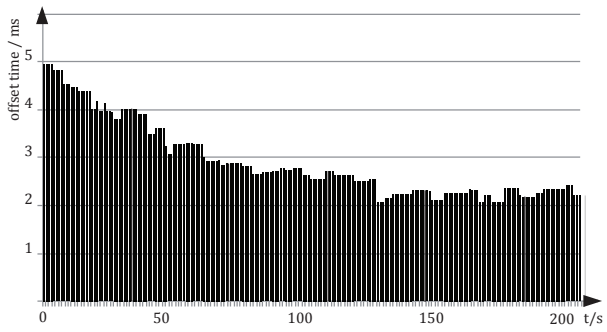

Fig. 6: Adjusted clockdrift characteristics in the WAN setup

two soundcard's workclocks with the same precise frequency. In contradiction to conventional clockdrift compensation techniques, our 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, would allow 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, our 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. At the current state the solution works with a frequency generator as the adjustable wordclock device. In the future we will substitute this with a stand-alone PLL circuit.

7. REFERENCES

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

[2] 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.

[3] Alexander Carôt and Christian Werner. Distributed network music workshop with soundjack. In *Proceedings of the 25th Tonmeistertagung*, Leipzig, Germany, 2008.

[4] C. Chafe, S. Wilson, R. Leistikow, D. Chisholm, and G. 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.

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

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

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

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

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