Extending the soundcard for use with generic DC sensors

Demonstrated by revisiting a vintage ISA design

Smilen Dimitrov
Aalborg University Copenhagen
Lautrupvang 15
DK-2750 Ballerup, Denmark
sd@imi.aau.dk

ABSTRACT

The sound card anno 2010, is an ubiquitous part of almost any personal computing system; what was once considered a high-end, CD-quality audio fidelity, is today found in most common sound cards. The increased presence of multichannel devices, along with the high sampling frequency, makes the sound card desirable as a generic interface for acquisition of analog signals in prototyping of sensor-based music interfaces. However, due to the need for coupling capacitors at a sound card's inputs and outputs, the use as a generic signal interface of a sound card is limited to signals not carrying information in a constant DC component. Through a revisit of a card design for the (now defunct) ISA bus, this paper proposes use of analog gates for bypassing the DC filtering input sections, controllable from software - thereby allowing for arbitrary choice by the user, if a soundcard input channel is to be used as a generic analog-to-digital sensor interface. Issues regarding use of obsolete technology and educational aspects are discussed as well.

Keywords

Soundcard, Sensors, ISA, DC

1. INTRODUCTION

The "humble" beginnings of the soundcard[1] as a dedicated part of a PC system intended to produce audible sound, could be seen in the use of a timer circuit Intel 8253, to generate pulses in the audible frequency range and drive a speaker, thereby producing audible sound [27]. Since then, the PC soundcard has become a multichannel A/D interfacing device, able to work at CD quality (16 bits / 44.1 kHz) rates and above. Devices offering more than two output channels are commonly used to drive multiple speakers as part of surround sound home entertainment systems. Specialized soundcards with multiple inputs and outputs, and full duplex (playback while recording) capabilities, have found use in music recording in professional and home studios. Soundcards interface as add-ins to PCs through several busses, serial (USB, FireWire) or parallel (ISA, PCI)[2] in nature; although they are increasingly found integrated in PC motherboards.

A slightly different type of devices become increasingly popular with the academic and do-it-yourself community as A/D interfaces for utilization of sensor signals. Programmable, micro-controller based devices such as the open-source ARDUINO [2], that offer both A/D conversion and PC connectivity (through, for instance, USB), provide a relatively easy way to interface with a variety of off-the-shelf sensors, from popular software development environments such as `PD` [20], `Max/MSP` [6], or `Processing` [19]. However, these devices are also more limited in regard to sampling quality: for instance, the ARDUINO offers 6 multiplexed channels of 10 bit resolution, and the maximum serial transfer speed via USB is limited to 115200 bps - which puts a theoretical best-case upper limit of 1800 Hz[3] on the sampling rate for all channels.

Because of these limitations, a lot of prototypers and designers opt for a sound-card as a sensor A/D interface instead. This also relieves the designer of worrying about specifics of low level communication, and up-sampling the signals so they match the audio domain processing rate, when sensor signals are to be applied to audio in software. However, since a typical soundcard filters out frequencies outside of the audible 20Hz - 20kHz range, both on the input and the output, its use is limited with those sensing devices that produce output in this range. A soundcard has been in use by Leroy et al for capturing optical pickups [14], or as physical computing interface in context of artwork production [11]; but its use can go beyond musical applications - such as chemical analysis [16] or medicine [21].

The DC[4] bias filtering problem is most apparent with sensors that encode some useful information in the DC level[5]. A common way to circumvent this limitation is to use the DC signal to modulate a sinusoidal carrier in the audio range (usually using amplitude modulation); capture the modulated signal using a soundcard; and then demodulate in software [9]. In case of resistive voltage dividers, they can be driven directly by an AC[4] signal (conveniently, a sound-

[1]Ignoring earlier occurrences, such as the SID sound chip on a Commodore 64 (whose sound was provided as part of a TV output signal) and similar

[2]USB: Universal Serial Bus; ISA: Industry Standard Architecture; PCI: Peripheral Component Interconnect

[3]"*For communicating with the computer, use one of these rates: 300, 1200, ... or 115200.[1]*" Given a 64 bit frame is used to transfer analog data of six channels @ 10 bits, we have best-case period (ignoring start/stop bits) $T = 64[b]/115200[bps] \approx 555\mu s$; and frequency $f = 1/T \approx 1801.8$ Hz; a single 8-bit channel would transfer at 14.4KHz

[4]DC: Direct current; AC: Alternating current

[5]such as a force-sensitive voltage divider, which would provide pressure (or an accelerometer, which would show the influence of gravity) as change of DC level

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
NIME2010, June 15-18, 2010, Sydney, Australia.
Copyright 2010, Copyright remains with the author(s).

card can generate a sinusoidal signal for the purpose), in which case the output will conform to the soundcard input limitations. Arguably, there would be some loss of information when using this method - especially if the modulating signal has a spectrum extending above half the the carrier frequency; also, software resources are spent in demodulating a high-speed audio signal, in addition to applying the demodulated signal as a control signal in the application.

On the other hand, direct modifications to commercial sound cards - intended to bypass input sections and allow sampling of DC signals - have also been proposed [15]. In similar fashion, this paper proposes that by using software-controlled analog gates, filtering sections at soundcard inputs can be bypassed - thereby allowing for user-configurable possibility of designating chosen inputs as "sensor" inputs. This relatively simple architectural change, would allow for both high-fidelity acquisition of DC signals, and reuse of common software tools made to interface with soundcards. To test this assumption, a vintage ISA design has been implemented, along with a corresponding C program - discussed further on in this paper.

Therefore, this project aims to demonstrate a simple implementation of PC controlled bypass switches in a soundcard (as an extension to allow interfacing with generic DC sensors), by first implementing and documenting a hardware platform – that can, to some extent, be considered a soundcard.

1.1 Approach

The DC bias filtering problem outlined in the Introduction is, in essence, a problem of A/D data acquisition – and thus, an engineering problem – however, one that, arguably, influences a lot of research within electronic music instruments. While custom and suitable A/D hardware may be available on the market, it is often expensive – the cost not scaling with the budget of departments that deal with electronic music. Current affordable A/D hardware (such as ARDUINO), on the other hand, rarely provides audio-quality sampling rates. The technical specs (audio sampling rate) and ubiquity (low price) of a typical soundcard, then, would make it an ideal "middle-way" choice of A/D platform for electronic music instrument research.

Thus, even though we are talking of, in principle, a relatively simple engineering problem - it is difficult to demonstrate a simple (first-iteration) solution to it, as it is difficult to find an accessible platform *to implement* the solution *on*: this platform needs to behave sufficiently as a soundcard (can interface to a PC, and can perform A/D and D/A conversions at audio rates), and needs to allow space for hardware modifications which is not prohibitively costly.

However, a typical soundcard is an industrial product, aiming to turn a profit by satisfying a range of mass-market needs - and as such, electronic music instrument researchers are unlikely to influence industrial-level modifications to a soundcard product, useful specifically for them. Many soundcards today are single ICs integrated on a PC motherboard, making manual hardware modifications near impossible; and while standalone soundcards may still offer designs based on individual dedicated ICs (allowing more space for tinkering), they will also incur not only greater financial cost, but also a cost involved with understanding the low-level work-flow of the device (which will, most likely, have to be obtained through reverse-engineering, as soundcard manufacturers are unlikely to publish such details publicly).

In such a market environment, researchers are likely to start thinking about handicraft custom-made soundcard implementations. As the typical entry-level research electronics lab, can be likened to a lab accessible for the enthusiast electronics instrument maker, the handicraft approach also shows a promise of direct applicability of results outside of academic circles. Unfortunately, at the end of the (first decade of 20)00's, it is rather difficult to find a starting point for such handicraft development: there are no open public projects dedicated to hardware soundcard implementations, and while there are resources discussing different aspects relevant to development, their discussion level often requires more advanced engineering experience.

In documenting the development process of a soundcard-like hardware, this project could then also be used in further open discussions of handicraft implementations of soundcard and AD/DA hardware. The starting point for this hardware platform is the only easily readable resource at the time of development, [22], which discusses both hardware and software entry-level issues. Although this design, based on the ISA bus, is old and in market terms obsolete, it has the benefit of using discrete ICs for A/D and D/A converters, as well as for logic signalling. This is, arguably, closer in principle to engineering textbook material, and thus has the educational benefit of facilitating easier understanding of architectural issues surrounding soundcard-like hardware. Additionally, dealing with obsolete technology provides a historical archiving aspect to the project.

This project provides a preliminary general conclusion on the suitability of use of analog switches for interfacing DC sensors, by providing measurements of a generic input signal with a DC component from a signal generator. For educational purposes, the key issues (among them, quantifying the sampling rate of the device) in the process of obtaining these measurements will be outlined in this paper. For more details, as well as source code, consult the webpage [7] - which contains an extended version of this paper, with additional introductory material and implementation details.

2. PROBLEM OUTLINE

A simplified input channel section of a sound card is shown in Fig. 1:

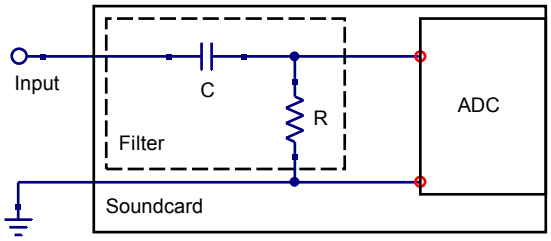

Figure 1: Simplified diagram of a sound-card input channel.

In Fig.1, a basic CR (*capacitor-resistor*) high-pass filter schematic is taken to represent a simplification of input filters usually found in soundcards, in order to emphasize the filtering of DC signals. To illustrate this influence, a simple experiment can be performed: a stereo mini TRS (*tip-ring-sleeve*) connector can be plugged in a microphone (or line-in) input of a soundcard, and the ground and a channel wire can be used to connect to a 1.5V battery. If we try to capture the resulting input data using audio recording software (such as `Audacity` [3]), we would obtain a capture like the one shown on Fig. 2.

Instead of obtaining a constant (DC) level in the time period between approx 1.3s and 4.5s on Fig. 2, what is shown is a typical signature of a high-pass filter in the time domain - a positive spike at the moment when the battery

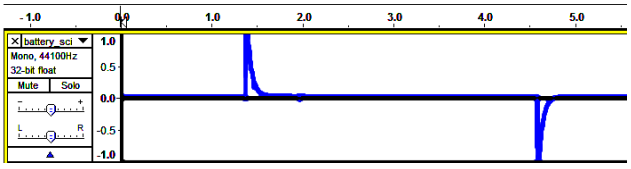

Figure 2: Capture of a battery being connected (at 1.5s) and disconnected (at 4.5s) to a sound card microphone input in Audacity software.

is connected, and a negative one when it is disconnected. This paper proposes that obtaining DC signals could be achieved by implementing a voltage controlled switch (analog a.k.a. bilateral switches [gates]), ultimately controlled by software, that bypasses the entire input filter section, as shown on Fig. 3:

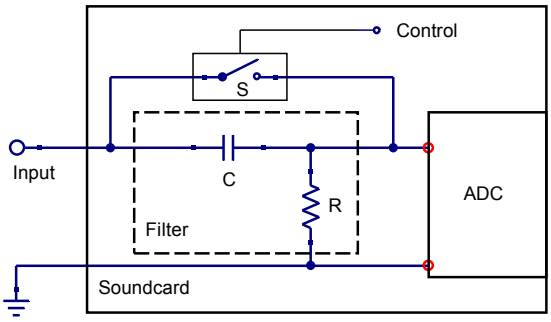

Figure 3: Simplified diagram of a sound-card input channel, with a controllable switch bypassing the input filter.

3. SOUNDCARD PLATFORM

In order to test the assumption given in the problem outline, a hardware platform needs to be chosen, that behaves essentially like a sound-card - but also allows for demonstration of activating added analog gates via software. Primarily because reverse-engineering an off-the-shelf soundcard device to behave as imagined would have been problematic, but also due to potential educational benefits - a do-it-yourself implementation of a soundcard design was sought instead.

Although the Internet does offer some information and tutorials on building basic extension cards for a PC - interfaced through either USB, ISA or PCI buses - it is difficult to find an available design, which is specifically intended to represent a sound card (or even a high-speed A/D converter). The only one found appropriate for the purpose, is the design for a ISA extension card by Joe D. Reeder [22] - a now abandoned product, that was intended for learning the essentials of software-controlled hardware. Since the schematic and the basic software code, related to this card, are still available on the website associated with this product (www.learn-c.com), it was this design that was taken as a starting point for development.

The schematic for this "Learn-C" ISA card was reimplemented as a double-sided printed circuit board (PCB), with an added **CD4066** [8] CMOS quad bilateral switch. Since the ISA bus is now obsolete and cannot be found in modern PC systems, an older PC based on an ELITEGROUP ECS P6VAP-A+ (or P6VAP-AP) motherboard, with a single ISA slot, was obtained. Code in C language from [22] was used to implement test software, and AGILENT 54621A oscilloscope was used to capture signals on board - using the open-source `agiload` [23] package for transferring oscilloscope traces to a PC as raw data. The experiment finally consisted of using a vintage WAVETEK MODEL 145 signal generator to produce an approx. 440 Hz sinusoid voltage with a DC offset level, and capturing this voltage with the "Learn-C" card and test software on disk. As the test software allowed for user-controlled activation of the bilateral switches, these were alternately turned on and off during capture - and the captured signal was observed in the open-source `Audacity` software.

3.1 ISA hardware implementation

The original schematics for the "Learn-C" ISA card found in [22] was rebuilt using open-source `KiCad` [5] software (Fig. 4); the same software was also used to produce the PCB layout. The design relies only on the ISA bus power supply pins 1, 3, 5, 7, 9, 10, 29, 31 (*GND, +5V, -5V, -12V, +12V, GND, VCC, GND*) as well as ISA IO pins 2, 13, 14, 20, 33-40, 42, 53-62 (*RESET, IOW, IOR, OSC, D0-D7, AEN, A9-A0*), for power supply and PC connectivity.

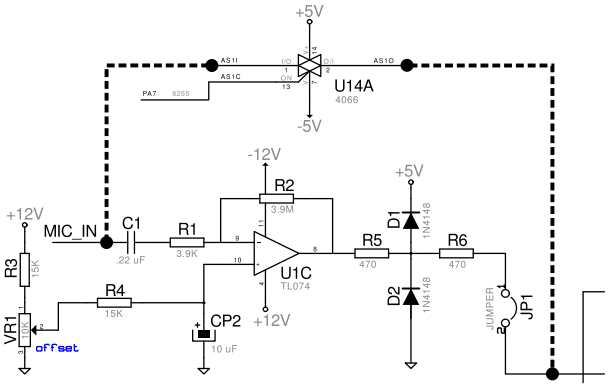

Figure 4: Part of schematic of the ISA card; emphasized dashed lines indicate connections of bipolar switch (see [7] for a complete schematic).

In simple terms, whenever a I/O command like `outportb (_port, _data)` is executed from software, a binary representation of the `_port` address is set up as respective high or low voltages on the address pins of the ISA bus; the "Learn-C" design then employs standard **74xx** TTL logic ICs to implement an address decoder that will interface with the bus, and provide appropriate trigger signals for the rest of the hardware, when the card is addressed from software.

Most of the original design of the "Learn-C" ISA card has been reproduced, although with some differences. For instance, a socket for the **CD4066** analog switch was added, and not all wired connections were implemented on the PCB (for instance, headers were left unconnected, as well as most of the I/O port pins of the **8255** [10] PPI chip). On the other hand, both **DAC0832** [17] digital-to-analog converters (DAC) were implemented[6]. The implementation of the card is shown on Fig. 5.

3.2 Software

As mentioned previously, the source code for the "Learn-C" ISA card given in [22] is C code, originally intended to run under **MS-DOS**; as part of this project, portions of that code were ported to **Linux** as well. The "Learn-C" example programs can also run in the command prompt shell of **Windows XP** - however, they cannot be directly compiled with modern **Windows** C compilers. The reason for this

[6]although only a single one was actually tested; and none are needed for the input filter switching experiment.

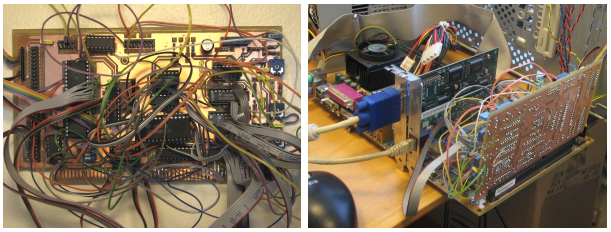

Figure 5: Left: image of the wired card; right: card in ISA slot of motherboard of test PC.

is that the code relies on C commands like `inportb` and `outportb` (or `inp` and `outp`), which represent direct I/O port access; however direct I/O port access is disabled for **Windows** architectures newer than **NT** [24] (and can be achieved only through programming a device driver). This demanded use of vintage C compilers (`DJGPP` for **DOS**) under **Windows**; on the other hand, programming direct I/O port access in C under **Linux** is relatively straightforward.

4. TESTING PROCEDURE

The testing procedure consisted of two distinct steps. The first was to use a known signal to determine the sampling rate of the analog-to-digital converter (ADC); and to check whether this rate is correct (by auditorily comparing a capture from the ISA card's ADC to the known signal). The second step (the actual experiment) consisted of sampling and capturing a known, DC-biased, sinusoidal signal; turning the analog switch on and off during capture; and looking for presence of a DC level recording when the switch was active in observations of captured data.

4.1 Determining the ISA card sampling rate

The ISA card sampling rate was determined to be around 12725 Hz (the sampling resolution is limited at 8 bit by choice of **ADC0809** [18] chip). However, this could be reached only after dealing with an apparent timing problem.

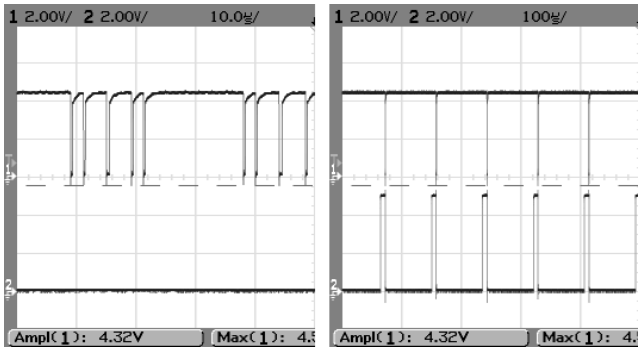

Figure 6: Oscilloscope screen captures of card signals `EOC` (top) vs `IOW` (bottom). Left: 50 μs of non-periodic behavior; right: 0.5 ms of periodic behaviour.

Figure 6 (left) shows that `EOC` (signal produced by the ADC chip on the card at end of each sample conversion) is not periodic. Eventually, it turns out that this can be resolved by introducing a short delay between the `outport` and the first `inport` command in the reading loop, as shown in Listing 1:

```
unsigned base; unsigned adcport; base = adcport =
    0x200; //1000000000
if ((fp = fopen(FILENAME,"w+b")) != NULL) {
  while(!kbhit())
  {
    outp(adcport, 0); // start channel 0 conversion , by
          writing whatever value to address adcport
    iz = del; while (iz > 0) iz--; // fake delay, ' del '
          increments
    while(!(inp(adcport+0x18) & 0x80)); // wait for EOC
          ready : 0x18 = 000011000, 0x80 = 010000000
    x = inp(adcport); // read ADC value into variable x
    fputc((char)x,fp); // since value is 8- bit anyways ,
          just cast to char and save to disk
  }
}
```

Listing 1: ADC reading loop code

As Listing 1 indicates, the recorded file is simply a stream of 8-bit characters. This file can be imported in `Audacity` as raw data, and the sampling frequency is set upon import. The multi-track capabilities of `Audacity` also allow both the original 440 Hz source signal, and its ADC capture from the ISA card, to be played simultaneously in spite of differing sampling rates (44.1 KHz vs. 12.7 KHz); their respective pitches can be heard as audibly close – which is a confirmation that the measurement of the sampling rate is correct.

4.2 Test of analog switch functionality

As mentioned previously, a **CD4066** was used to implement an analog switch, which bypasses the input preamp/filter section of the ISA card. This chip offers four analog switches - only a single one was used, defined by pins 1 and 2 as switch connectors (connected as on Fig. 4), and pin 13 (**CD4066**/p13) as voltage control. To control **CD4066**/p13, the **8255** PPI on the ISA card was used, as it offers three ports (A, B and C) of 8 pins each, which can be configured to act as either digital inputs, or digital outputs. Just a single output pin is needed from a single (configured as output) port, in order to control the analog switch; pin 37 (or pin 7 of port A) of **8255** (**8255**/p37) was picked for the purpose, and was connected to **CD4066**/p13. The **8255** offers three different modes of configuration of its three ports; here any mode that configures port A as output will do, and the function `set_up_ppi` from [22] was used to quickly configure the ports.

5. RESULTS

The procedure described in section 4.1 was used to capture a DC offset sinusoidal signal. This signal was generated by a vintage WAVETEK MODEL 145 signal generator, which doesn't provide for fine-tuning control of the AC amplitude and the DC offset separately. Eventually, a signal spanning between 0.5V and 1.66V, set at approximately 450 Hz, was used as the input signal for ADC capture. The signal arriving at the input pin **ADC0809**/p26 (IN0), without and with the influence of the switch, is shown on Fig. 7.

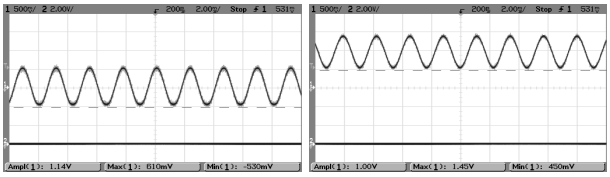

Figure 7: Oscilloscope screen captures of the input signal brought to ADC0809 IN0 (left) without the influence of analog switch; (right) while analog switch turned on

This input signal was captured on a disk file using the procedure given in section 4.1, as the analog switch was turned

on and off during capture using keyboard key presses. The data captured by the ISA card can be verified through an import in `Audacity`, depicted in Fig. 8.

As it is obvious from Fig. 8, activating the switch allows the DC level of the signal to be captured by the card's ADC; and the waveforms obtained in `Audacity` remain relatively faithful to the original input signal shown in Fig. 7 (ignoring the clipping of the negative semiperiod of the original signal).

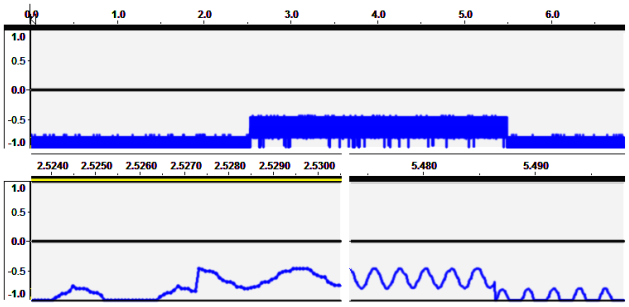

Figure 8: Top: Six second capture of an input signal with DC, with the switch activated in mid-capture; Bottom: temporal zoom at moments of activating (left) and deactivating (right) the switch.

6. DISCUSSION

As we have shown that, with this platform, we can arbitrarily choose to capture the DC level of a signal from a signal generator – we can conclude that the same behavior can be expected for signals derived from sensor circuits (for instance, a FSR-based voltage divider) as well. Thus, this paper indicates that analog switches bypassing the input preamplifier section of soundcard inputs, would be a relatively simple change to perform on existing soundcard designs, thereby expanding their purpose to high speed A/D interfaces for generic sensors. Similar change could be implemented for output sections, thereby allowing soundcards to be used as generic signal generators. The DC blocking capacitors are present in a sound-card, of course, for a purpose - primarily to protect speakers from constant DC biasing, and thereby prolong their lifetime [29]. The design change proposed here, would ideally leave that regime unchanged for audio purposes - and simply introduce a different one, more suitable for sensor data acquisition, when the user requires it.

6.1 The soundcard platform

This project started by looking for a hardware platform, that behaves essentially like a soundcard. Arguably, the ISA platform used here cannot be even considered a sound card, until it can be used by typical audio software (like players or editors) from the application level in an operating system. Although this ISA card could be extended to play back audio (encoded for the measured DAC frequency) - at the state presented here, it behaves more like a generic signal generator and sampler, than a modern soundcard. Finding a modern open hardware platform, that allows for the type of research as in this project, is still problematic; although FPGA[7] based designs, such as the ones described in [13], are very promising as a base for multichannel, high-speed, hybrid audio/sensor interfaces.

Whereas it is utopistic to expect that this paper could significantly influence industry in such a manner, that similar modifications become a standard for future sound-cards - it certainly aims to inspire designers working in the electronic music instrument field, to focus at the intricacies of digital interfacing with analog signals; and to consider using older, historic designs for appropriate purposes - while being aware of potential obstacles. Mostly, one has to deal with hardware availability, although software can be an issue as well. However, in the case of ISA, this project shows that currently there is still a palette of tools that can target such machines and corresponding functionality, many of them free and open-source; yet, one has to be prepared for a time investment for straightening out potential glitches.

Hardware. Arguably, the ISA design used as a test platform here, doesn't even come close to issues in contemporary sound-card production – to begin with, negative voltage values of the input signal are clipped, as the ADC chip by default can sample only positive voltages. However, it is a good educational tool to introduce general issues related to design of soundcards and corresponding software. Namely, it is often difficult for beginner engineers to come to a practical example, which is both relatively simple to understand as introductory material (and thus easy to relate to theory) - and can be practically implemented to serve a purpose, already known from a user perspective. So in spite of the obsolescence of ISA, this design can still be useful educationally. One positive point of using discrete components in this card implementation, is the possibility to measure their signals individually with an oscilloscope and thus observe the interdependence of different signals on the physical, electric level - something that becomes arguably difficult, if the components are integrated as part of a single chip.

Software. Finally, if there is a single thing this project points out, it is that one cannot perceive the soundcard as a system separate from its host PC; that is, a soundcard-like hardware is genuinely bound to the intricacies of the host operating system. In other words, even if the hardware is theoretically capable of achieving greater sampling speeds, the effective achieved sampling rate will be limited by the method of accessing the device from software. In this project, a steady 12.7 KHz sampling rate was achieved through a program in C, only after implementing delays in the reading loop. The cause of this need for delay is, likely, that `outportb` in Listing 1 returns, before the corresponding `IOW` signal (which triggers each start of conversion) has been set electrically. Hence we need to wait, so we're sure conversion has started – before reading the `EOC` signal to see if a value is ready. However, due to operating system overhead, the actual sampling period is probably more indicative of the time it takes for the operating system to complete a single iteration of the main `while` loop in Listing 1, than it is of the limits of the hardware itself. Thus, entering development of soundcard hardware, necessarily implies involvement with issues in the inherent unpredictability of instruction execution times, OS scheduler granularity and latency [4] and development of device drivers [28, 12] – with a particular focus on the flow of time [25] – before coming close to a platform able to interface with an operating system, in a manner expected of a soundcard. In other words, here the hardware acquisition of *each* sample is initiated and transferred by C software on the PC – with a proper driver approach, the hardware acquisition process could run independently on the card hardware at high rates; and batches of sampled data could be transferred as arrays to the PC at longer intervals (i.e. "*buffered reads*"; see also Direct Mem-

[7]FPGA: Field-programmable gate array

ory Access/DMA [26]). Note, however, that in spite of the shortcomings of this projects' naïve approach, it still managed to obtain 12.7 KHz effective end-to-end sampling rate, which is close to the 14.4 KHz USB-limited maximum for a single channel, 8-bit ARDUINO transfer (but is still below the CD-quality 44.1 KHz expected of a typical soundcard).

7. CONCLUSION

In conclusion, the project managed to demonstrate the possibility to use analog switches, for software controlled bypass of input filters of a sound-card device; thereby, in principle, allowing it to interface with generic sensor circuits that produce DC-offset voltages. However, claims cannot be made on the feasibility of implementing such a change in an existing commercial sound-card design. The project also illustrated the specific problems encountered with usage of a card design for the now obsolete ISA bus; while demonstrating how it can be used to emulate modern hardware (at least to a degree, sufficient to expose the problem at hand). Additionally, the simplified analysis of particular issues, aims to serve as an educational introductory example for designers starting with digital hardware design; in line with this aspect, the source files for schematics and code, as well as the full list of online references (too numerous to include here) relevant to the topic discussed in this paper, are provided on the project webpage [7]. The educational and the historical perspective of this project, both aim to contribute to furthering the discussion of A/D and D/A hardware in the context of electronic music instruments development.

8. REFERENCES

[1] Arduino. Arduino - begin. http://www.arduino.cc/en/Serial/Begin, 2010.

[2] arduino.cc. Arduino homepage. http://www.arduino.cc/.

[3] audacity.sourceforge.net. Audacity homepage. http://audacity.sourceforge.net/.

[4] T. P. Baker, A. i Andy Wang, and M. J. Stanovich. Fitting linux device drivers into an analyzable scheduling framework. In *Proceedings of the 3rd Workshop on Operating Systems Platforms for Embedded Real-Time Applications*, 2007.

[5] J.-P. Charras. Kicad homepage. http://www.lis.inpg.fr/realise_au_lis/kicad/, 2010.

[6] Cycling 74. Max/msp homepage. http://cycling74.com/products/maxmspjitter/.

[7] S. Dimitrov. Extending isa soundcard webpage. World Wide Web electronic publication, http://imi.aau.dk-/∼sd/phd/index.php?title=ExtendingISASoundcard, Last Accessed: 20 March, 2009.

[8] Fairchild Semiconductor. *CD4066 datasheet*, http://www.fairchildsemi.com/ds/CD%2F-CD4066BC.pdf edition, 2005.

[9] M. H. Puts, J. Pokorny, J. Quinlan, and L. Glennie. Audiophile hardware in vision science; the soundcard as a digital to analog converter. *Journal of Neuroscience Methods*, 142(1):77–81, 2005.

[10] Intel Corporation. *8255 Programmable Peripheral Interface (PPI)*, http://jap.hu/electronic/8255.pdf edition, 1995.

[11] K. Jo. Audio Interface as a Device for Physical Computing. *Proceedings of Audio Mostly 2008 - a Conference on Interaction with Sound*, pages 123–127, 2008.

[12] M. K. Johnson. Device driver basics. http://users.evtek.fi/∼tk/rt_html/ASICS.HTM, 2010.

[13] S. Kartadinata. The Gluion advantages of an FPGA-based sensor interface. In *Proceedings of the 2006 conference on New interfaces for musical expression*, pages 93–96. IRCAM-Centre Pompidou Paris, France, France, 2006.

[14] N. Leroy, E. Fléty, and F. Bevilacqua. Reflective optical pickup for violin. In *Proceedings of the 2006 conference on New interfaces for musical expression*, pages 204–207. IRCAM-Centre Pompidou Paris, France, France, 2006.

[15] S. Molloy. How to Modify a PC Sound Card to Allow D.C. Voltage Measurements. World Wide Web electronic publication, http://web.archive.org/web/20080108175023/-http://www.mandanet.net/adc/adc.shtml, Last Accessed: 7 April, 2009.

[16] D. Nacapricha, N. Amornthammarong, K. Sereenonchai, P. Anujarawat, and P. Wilairat. Low cost telemetry with PC sound card for chemical analysis applications. *Talanta*, 71(2):605–609, 2007.

[17] National Semiconductor. *DAC0830/DAC0832 datasheet*, http://www.national.com/ds/DA/DAC0830.pdf edition, 2002.

[18] National Semiconductor. *ADC0808/ADC0809 datasheet*, http://www.national.com/ds/DC/ADC0808.pdf edition, 2009.

[19] processing.org. Processing homepage. http://processing.org/.

[20] puredata.info. Puredata homepage. http://puredata.info/.

[21] K. Reddy, J. Bai, B. George, N. Mohan, and V. Kumar. Virtual Instrument for the Measurement of Haemo-dynamic Parameters Using Photoplethysmograph. *Proc 23rd Int ConfIEEE, IMTC-2006*, pages 1167–1171, 2006.

[22] J. D. Reeder. Tutorial - controlling the real world with computers. World Wide Web electronic publication, http://learn-c.com/, Last Accessed: 12 March, 2009. Last Modified: 03/21/2005 22:53:00.

[23] J. Rinas. agiload - fetch data and screenshots from agilent 5462x oscilloscopes. http://www.ant.uni-bremen.de/whomes/rinas/agiload/.

[24] D. Roberts. Dr. Dobb's - Direct Port I/O and Windows NT. World Wide Web electronic publication, http://www.ddj.com/184409876?pgno=3, Last Accessed: 12 April, 2009.

[25] A. Rubini and J. Corbet. Flow of time. In *Linux device drivers*, chapter 6. O'Reilly Media, Inc., second edition, 2001.

[26] A. Rubini and J. Corbet. mmap and dma. In *Linux device drivers*, chapter 13. O'Reilly Media, Inc., second edition, 2001.

[27] T. Savell. 23.3 Digital Audio Processors for Personal Computer Systems. *Linear Algebra and Ordinary Differential Equations*, 1993.

[28] C. Williams. Linux scheduler latency. *Red Hat Inc*, 3, 2002.

[29] www.maxim ic.com. APPLICATION NOTE 3979 - Overview of DirectDrive® Technology. World Wide Web electronic publication, http://www.maxim-ic.com/appnotes.cfm/an_pk/3979/, Last Accessed: 12 April, 2009.