CTRL: A Flexible, Precision Interface for Analog Synthesis

John Harding
Ulster University, Magee
Derry/Londonderry, N. Ireland
j.harding@ulster.ac.uk

Richard Graham
Delta Sound Labs
Memphis, TN, USA
ricky@deltasoundlabs.com

Edwin Park
Delta Sound Labs
Charlotte, NC, USA
edwin@deltasoundlabs.com

ABSTRACT

This paper provides a new interface for the production and distribution of high resolution analog control signals, particularly aimed toward the control of analog modular synthesisers. Control Voltage/Gate interfaces generate Control Voltage (CV) and Gate Voltage (Gate) as a means of controlling note pitch and length respectively, and have been with us since 1986 [3]. The authors provide a unique custom CV/Gate interface and dedicated communication protocol which leverages standard USB Serial functionality and enables connectivity over a plethora of computing systems, including embedded devices such as the Raspberry Pi and ARM based devices including widely available Android TV Boxes.

We provide a general overview of the unique hardware and communication protocol developments followed by use case examples toward tuning and embedded platforms, leveraging softwares ranging from Pure Data (Pd), Max, and Max for Live (M4L).

Author Keywords

CTRL, CV, Control, Voltage, Analog, Synthesis, Communication, Protocol, Interface, Max, Pd, Embedded, Max for Live

CCS Concepts

•**Applied computing** → **Sound and music computing;**
•**Hardware** → *Sound-based input / output;*

1. INTRODUCTION

Recent years have seen a resurgence in popularity of analog synthesis technologies, with NAMM 2017 figures describing the analog synthesis market as vibrant [1] showing the biggest increase in the music market currently at nine percent growth, while manufacturers such as ARP, Korg and Roland re-enter the boutique analog synthesis market following their departure in the late 1980s to early 1990s.

Digital to Control Voltage (CV) interfaces have existed since the mid-eighties and, in many cases, MIDI continues to be employed as the driving communication protocol for those interfaces. It is our contention that with a pitch range of 128 values the standard MIDI protocol fails to provide sufficient resolution to explore the full potential of analog synthesis devices.

Recent years have seen some notable exceptions, in terms of higher resolution CV control interfaces, from manufacturers such as Expert sleepers[1] and Snyderphonics[2] amongst others. Many of these examples, however, require additional external hardware in order to function; for example an ADAT/Light-pipe[3] or SPDIF[4] enabled audio interface. This factor clearly increases complexity and cost for the user whilst practically eliminating the possibility of using an embedded system for control due to lack of available hardware drivers for the required supporting devices. The more recent 14 bit MIDI Standard has the potential to solve the combined resolution and connectivity issues to some degree. However, this particular variation of the MIDI protocol has yet to be widely adopted.

Recent community interest toward embedded designs is clearly evident [13, 6, 8, 5]. It is also our contention that a driver based approach to connectivity results in limitations in application across platforms (Win, PC, Linux, etc.) and makes combining analog synthesizers with embedded systems more difficult. Clearly, a driver based approach provides a certain level of reliability.

We present an alternate approach to communication and control voltage generation and mapping. As the range of available control values increases, the complexity of managing these values in a practical application becomes increasingly more challenging for the user. In response, we present various examples (appendices) of the increased capabilities afforded by CTRL within existing frameworks such as Max, Pd and Max for Live, in order to simplify execution.

The following section presents a general overview of CTRL, its hardware architecture and custom communication protocol and is followed by brief working usage example cases.

2. ARCHITECTURE

The design goals for CTRL are to provide a high resolution control voltage interface[5] that fits within the popular eurorack standard [2] and which features approaches to user accessible calibration in order to both increase logarithmic conformity and to provide extensive possibilities for user configurable intonation/scaling. The driver-less approach outlined in section 2.3 is a key consideration for us in ap-

NIME'18, June 3-6, 2018, Blacksburg, Virginia, USA.

---

[1] `http://www.expert-sleepers.co.uk`
[2] `https://snyderphonics.com`
[3] `http://www.expert-sleepers.co.uk/es3.html`
[4] `http://www.expert-sleepers.co.uk/es4.html`
[5] 16 bit within a 0VDC to 5VDC range allows 1092 values per semitone @ 1V/Oct. Calculated by (2ˆ16/5)/12.

proaching to the long term goal of cross-platform (Linux, OSX, Win), cross-application (Pd, Max, M4L) cross-hardware (Pi, ARM) access to control voltage interfaces. We contend that this approach will enable developments towards embedded analog/digital instrument design, as demonstrated in section 3.3, while offering the potential to leverage a plethora of embedded devices in future.

2.1 Design and Specification

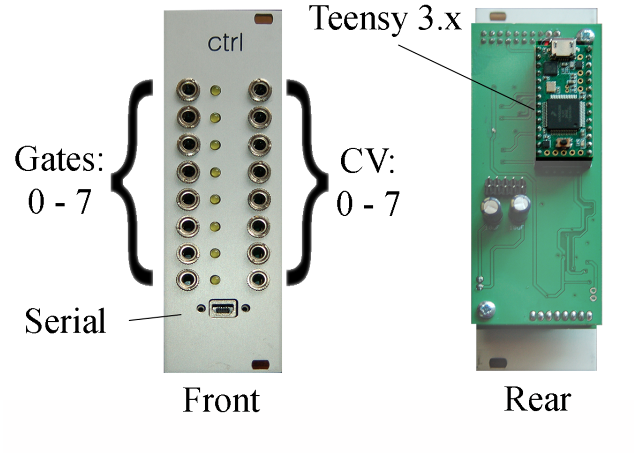

Figure 1: The CTRL unit in standard eurorack format

The CTRL hardware, depicted in Figure 1 and authored by John Harding in 2018, is specified as follows:

- 8 x Control Voltage (CV) outputs: 0VDC through to 5VDC
- 8 x Gate outputs: 0VDC or 5VDC with individual LEDs providing status indication
- Powered by Teensy 3.1/3.2 Arm Cortex M4 based Microcontroller.
- CV is produced by a Octal Precision DC stable 16 bit Digital to Analog Converter (DAC)

The DAC chosen, model DAC8568 from Texas Instruments, is DC stable and highly accurate, featuring a 5ppm/°C max drift. Individual step voltage of 0.0000763 VDC.

2.2 Communication Protocol

CTRL features a custom communication protocol which leverages standard USB Serial communication functionality and its structure is outlined as follows.

{Status}{DAC0}{DAC1}{DAC2}{DAC3}
{DAC4}{DAC5}{DAC6}{DAC7}{Gates}

Figure 2: A functional overview of the custom communication protocol which drives CTRL

The protocol authored by John Harding and Edwin Park in 2018, and as outlined diagrammatically in Figure 2, begins with a Status byte for the purpose of message ordering analysis which is immediately followed by a stream of seventeen bytes which collectively represent the states of all eight DAC and Gate outputs. Since output values of the individual DAC channels are sixteen bits wide, and serial communication is limited to transmitting single byte size packets, we separate each sixteen bit value into High and Low byte pairs. These byte values can then be transmitted serially and reassembled computationally by the micro-controller upon receipt with an inverse process. The values which define the state of Gate outputs zero through seven are represented by a single zero or one, eight of which are manipulated to fit within a single byte by leveraging bit shift operations on the performance system, and further decoded by the micro-controller upon receipt with an inverse process.

2.3 Latency Performance

Recent studies have discussed the importance of low and consistent latency in musical systems and the accepted thresholds [11]. In a real world system we obviously aim to achieve the lowest latency figure possible, with Mc Pherson et. al. quoting Wessel and Wright's 10ms standard as the considered upper threshold [11]. Since CTRL produces and transmits no audio directly its latency figures can be kept exceptionally low.

In order to estimate the latency of CTRL, a modified version of the micro-controller firmware is employed and described as follows. The micro-controller decoding firmware is modified to toggle a previously unused digital pin; high at the beginning of the serial read, and low again at the end of the complete DAC/Gate write cycle. In this manner the instruction cycle latency can be estimated in real world conditions by means of time based oscilloscope measurements of the periodicity between successive toggle events. The instruction time period, measured using a digital storage oscillocope[6] has tested to be consistently within 50uS to 320uS range, extending toward 1mS max. in some cases.

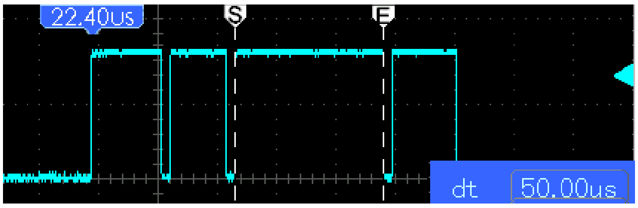

Figure 3: Modified oscilloscope capture of the firmware loop cycle latency

These figures align well with those predicted, with comparison to the benchmarking tests on Teensy, performed by the manufacturer PJRC [15].

3. EXAMPLES OF USAGE

3.1 Tuning and Intonation Calibration

Analog oscillators, of the type commonly used in modular synthesis, most frequently feature a hardware exponential converter which functions to process the frequency related control voltage input signal(s) in order to achieve exponential conformity over frequency.

This conversion stage often comprises of a pair of matched bipolar junction transistors (BJT) in a mirrored arrangement as shown diagrammatically in Figure 3 and can be seen in schematics from Oberheim and Moog [4, 12].

---

[6] Model Hantek DSO5102P 100MHz, dual channel DSO.

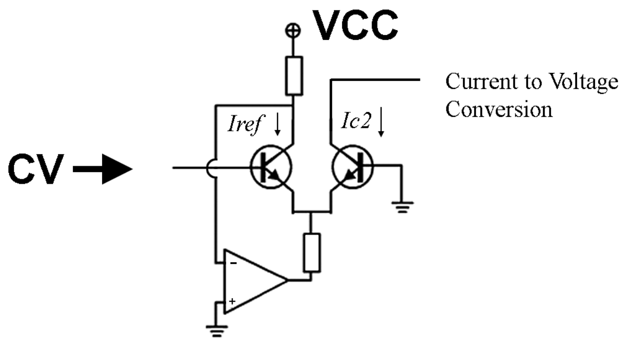

Figure 4: A linear Control Voltage to exponential current converter schematic. Adapted from R. Schmitz [14].

The purpose of the primary exponential input stage is to provision conversion between linear CV input, for example the common One Volt per Octave (1/VOct) standard, and exponential frequency in order to drive the analog oscillator core. It is common for analog oscillators to drift in exponential conformity due to various manufacturing variations between the paired BJT devices themselves, and/or in relation to how successfully the BJT devices are thermally coupled and compensated within the VCO design. [10, 14]. Any resulting variation in exponential conformity inevitably leads to a drift from the target frequency.

Hardware MIDI to CV converters largely remain fixed in terms of calibration toward a particular target tuning system. Most common of these today is the aforementioned One Volt per Octave (1V/Oct) standard. Our contention is that the fixed nature of the calibration over voltage range limits the user toward targeting specific tuning standards, with physical hardware calibration often being a necessity. The use of MIDI as a driving mechanism places an inevitable limit on potential for fine frequency related adjustments to be made by the user.

In response, we present approaches toward calibrating the output of CTRL in attempt to circumvent tuning issues; both a manual procedure and a more experimental automatic tuning procedure, which are briefly discussed below.

3.1.1 *Manual Calibration*

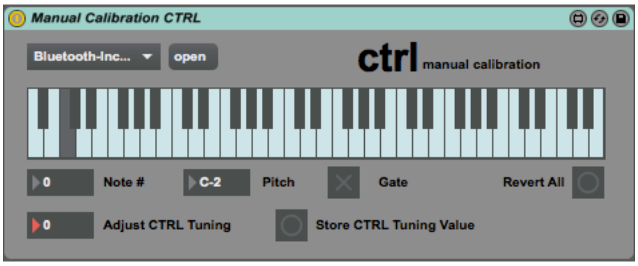

Figure 5: Max for Live manual calibration example.

By considering MIDI input data as simple trigger information, rather than relating to any particular pitch, we become free to assign any output voltage within the DACs 2ˆ16 range. This concept allows us to manage the expanded potential of pitches within existing frameworks.

Sixteen bit control values are stored within simple text files with indices that are accessed in response to MIDI note input data. In this manner, the user can freely choose and adjust, the tuning parameter for each input trigger and save each trigger parameter within a text file. With eight separate CV outputs, the user is free to explore separate tunings on each CV output. While the manual calibration process is somewhat labor-intensive, the code is prepared to initialize with twelve tone tuning (12-tET), which minimizes user input. This collective process allows the ability to calibrate toward exponential conformity or indeed, to explore alternate tunings. Calibration can be saved for recall at a later date. This procedure is demonstrated in the following video reference: `http://bit.ly/2mByKtc`

3.1.2 *Automatic Calibration*

As an additional approach, the authors present an experimental automatic calibration procedure which is designed to automatically calibrate an analog oscillator toward exponential conformity. Through iteration of the available 2ˆ16 range of output values and computational analysis of the frequency via pitch analysis[7], our goal is to provide an automatic calibration system to correct for any deviation from exponential conformity of the oscillator in question. This approach is currently at beta status and is demonstrated in the following video reference: `http://bit.ly/2BOWifZ`

3.2 Micro-tonal Sequencing M4L

As highlighted previously, MIDI to CV converters ordinarily provide fixed intonation that remains focused toward twelve tone equal temperament (12-tET). In this example, we explore the alignment of MIDI trigger data toward an alternate tuning system. Specifically, we target the twenty-two tone equal temperament, or 22-tET [7].

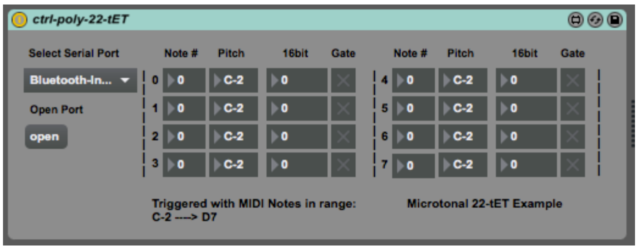

Figure 6: Max for Live 22-tET example.

In this scale, each octave is divided equally into twenty two equal steps. With a five octave range, and 2ˆ16 steps, we have access to 595(.78) values per division[8]. By approaching a similar triggering concept to that outlined in section 3.1.1, we assign 595 values per step and result in 22-tET microtonal control via standard MIDI capable software. In this example MIDI notes numbers 0 to 110 (C-2 to D7 respectively) are employed for triggering purposes.

Using similar approaches CTRL offers potential to implement alternate microtonal structures and systems by leveraging the functionality of both Pd and Max, for example the hayward tuning vine [9].

3.3 Embedded Examples

The cross platform nature of the USB serial communication protocol allows potential for connectivity via a vast array of

---

[7] Helmholtz∼ Max/MSP port is authored by Richard Graham and Edwin Park with the approval of the original author Katja Vetter is available from: `http://deltasoundlabs.com`

[8] Calculated as follows: (2ˆ16/5)/22.

serial capable devices including embedded computing. To date, we have successfully employed CTRL with two distinct embedded technologies; Raspberry Pi and an ARM based Android TV Box. Details toward video references of both embedded examples can be found in the appendices.

It is our position that the application towards these ARM based TV Boxes is of particular interest, given their generally attractive cost-to-specification ratio, wide availability and robustly enclosed nature. In this example, we leverage a low-cost and widely available example, model X96mini, which is re-purposed to run a Linux based operating system named Armbian and a distribution of Pd to drive CTRL, which is further driving an analog Eurorack synthesizer via CV and Gate outlets.

Developments in embedded technologies are currently expanding rapidly, we propose that significant scope exists for a variety of embedded, digitally controlled, analog musical systems to be developed in future, with CTRL featuring as the intermediary.

4. DISCUSSION AND FUTURE WORK

In this work, we have demonstrated a small sample of the possibilities afforded by CTRL and will strive to continually provide additional software and demonstration based content to the relevant repositories listed in the appendices, in the hope of easing boundaries to access. We also intend to make the hardware element available in the very near future.

The current revision of CTRL produces unipolar voltages within the fixed range of 0VDC to 5VDC. Future developments may include increased voltage output range in order to enable access to extended frequency range. Additional bipolar voltage modes are considered, which would allow the user to explore bipolar modulation capabilities such as digitally generated low frequency oscillators for example. The dedicated communication protocol is slimline and reliant on existing serial communication functionality however it has proven, through extensive testing, to be significantly robust for use in performance environments. Future work may include the generation of custom externals for both Max and Pd, which we hope may allow for direct transmission of audio-rate signal data to CTRL through employment of the libusb library for C. This approach may enable higher speed communication whilst retaining the important driverless design paradigm. Similarly, we are working toward development of custom VST/AU plug-ins within JUCE 5.3 (`https://juce.com`) in order to replicate software examples for use within VST/AU compatible Digital Audio Workstations. Finally, the development of a standalone version of the CTRL unit hardware is planned in order to simplify connectivity toward aforementioned embedded designs and beyond.

5. ACKNOWLEDGMENTS

We would like to extend a grateful acknowledgement to Katja Vetter for allowing the port of Helmholtz∼ to the Max/MSP environment. We would also like to extend gratitude to the Arts and Humanities Research Institute at Ulster University Magee for their support.

6. REFERENCES

[1] The 2017 NAMM Global Report. `https://www.namm.org/membership/global-report` page 32.

[2] Doepfer A100- construction details. `http://www.doepfer.de/a100_man/a100m_e.htm`.

[3] Doepfer MC1 specification. `http://www.doepfer.de/alte_anl.txt/MCV1_V5E.txt`.

[4] Oberheim OBX service manual, OB-X voice card schematic. `http://www.synthfool.com/docs/Oberheim/Oberheim_OBX/Oberheim_OBX_Service_Manual.pdf` page 20.

[5] M. Blessing and E. Berdahl. The joystyx: A quartet of embedded acoustic instruments. In *Proceedings of the international conference on new interfaces for musical expression*, pages 271–274, Copenhagen, Denmark, 2017.

[6] I. Bukvic. Pd-l2ork raspberry pi toolkit as a comprehensive arduino alternative in k-12 and production scenarios. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 166–163, London, United Kingdom, 2014.

[7] P. Erlich. Tuning, tonality, and twenty two tone temperament. *In XenharmonikÃt'n 17, An Informal Journal of Experimental Music.* 1998, revised 4/2/02.

[8] I. Franco and M. Wanderley. Practical evaluation of synthesis performance on the beaglebone black. In *Proceedings of the international conference on new interfaces for musical expression*, pages 223–226, Baton Rouge, Louisiana, USA, 2015.

[9] R. Hayward. The hayward tuning vine: an interface for just intonation. In *Proceedings of the international conference on new interfaces for musical expression*, pages 209–214, Baton Rouge, LA, USA, 2015.

[10] B. Hutchins. Log and exponential (antilog) circuits. *Electronotes, issue S-019*, 1981. `http://electronotes.netfirms.com/s019.pdf`.

[11] A. McPherson, R. Jack, and G. Moro. Action-sound latency: Are our tools fast enough? In *Proceedings of the international conference on new interfaces for musical expression*, pages 21–25, Brisbane, Australia, 2016.

[12] R. Moog. Service manual for moog minimoog model 204D. `http://www.synthfool.com/docs/Moog/minimoog/Minimoog_Service_Manual.pdf` page 4.

[13] G. Moro, A. Bin, R. H. Jack, C. Heinrichs, A. P. McPherson, et al. Making high-performance embedded instruments with bela and pure data. University of Sussex, UK, 2016.

[14] R. Schmitz. A tutorial on exponential convertors and temperature compensation. `http://www.schmitzbits.de/expo_tutorial/`.

[15] P. Stoffregen. USB virtual serial receive speed benchmarks. `https://www.pjrc.com/teensy/benchmark_usb_serial_receive.html`.

APPENDIX

The video repository houses reference toward embedded technologies and CTRL, and can be found at: `https://vimeo.com/user78996547`

The bitbucket code repository for CTRL firmware and coding examples for Max, Pd and Max for Live can be found at: `https://bitbucket.org/Nonzerojohn/ctrl-examples/src`