THE UNIVERSITY OF TENNESSEE KNOXVILLE

Journal of the Association for Technology in Music Instruction

Volume 3 | Number 1 Article 2

2022

Teaching Signal Processing Design for Effect Pedals: Making Signal Processing Tangible

V J. Manzo
Worcester Polytechnic Institute, vjmanzo@wpi.edu

Ryan McKenna
Worcester Polytechnic Institute, rpmckenna@wpi.edu

Matthew Halper
Kean University, halper@kean.edu

Journal of the Association for Technology in Music Instruction

Vol. 3, No. 1 (2022)

Teaching Signal Processing Design for Effect Pedals: Making Signal Processing Tangible

V.J. Manzo
Associate Professor of Music Technology and Cognition
Worcester Polytechnic Institute

Ryan McKenna
Affiliate Research Associate
Worcester Polytechnic Institute

Matthew Halper
Professor of Music
Kean University

ABSTRACT

Teaching digital audio processes and their uses in musical contexts is a challenging subject area for music technology instructors. While there are many books and resources that explain audio processes using diagrams and charts that merely require the reader to have modest skills in mathematics, computer science, or electrical engineering, these extra-musical subjects still pose great challenges for music students. Given the time and scope limitations implicit in a single semester, pedagogy for effects processes often has to be reduced to qualitative descriptions and audio examples. The more ambitious task of implementing such processes, realizing an algorithm via a plugin or a physical hardware pedal, is thus not always feasible. We have addressed this difficulty by creating an effect development kit including a software-based tool to facilitate rapid effect prototyping and guides for assembling hardware around a microcontroller. With these resources, students need only focus on understanding the specific audio process they wish to implement into an effect and then use these resources to realize the effect in a standalone device. This paper describes the open-source software and guides we have assembled in this kit, and our implementation of these resources with undergraduate students enrolled in a music technology course.

Manzo, V.J., McKenna, Ryan, and Halper, Matthew. (2022) Teaching Signal Processing Design for Effect Pedals:
Making Signal Processing Tangible. *Journal of the Association for Technology in Music Instruction*, 3 (1).

Introduction

Signal processing has become an inextricable part of the landscape of music technology instruction. Signal processing plug-ins are even found in entry level software. In addition, "stompbox" effects pedals are ubiquitous in our culture, where the electric guitar has risen to prominence.

Music technology instructors know that, given the mathematical and engineering nature of this domain, it can be one of the more difficult subjects to make tangible for students. There are many excellent books that discuss digital audio processes and their uses in musical contexts, often using diagrams and charts that merely require the reader to have modest skills in mathematics, computer science, or electrical engineering.

It is quite easy to demonstrate a particular effect (for example, a digital delay or compressor) and discuss the underlying process or algorithm in a qualitative or "over-easy" mathematical way—the show/listen-and-tell approach using a plug-in or existing hardware pedal. But the leap towards implementing a well-studied effect or, even more ambitious, designing and implementing a novel effect, is typically too large in a standard one- or two-term class—or well beyond the means of many student cohorts in certain tracks of music study.

To address this difficulty and open up a wide field of opportunity for students, teachers and developers, we have created a development kit and resources through the Electric Guitar Innovation Lab (EGIL) at the Worcester Polytechnic Institute. In this paper, we will first show the FX Testing Rig software-based tool we developed which facilitates rapid effects prototyping by taking advantage of the expansive Max Gen environment, a visual programming language made by the company Cycling '74. Secondly, we will show that the hardware, the physical

stompbox implementation, is made viable by commercially available programmable microcontrollers and circuit boards, such as the Electrosmith Daisy Seed and Pedal PCB Terrarium, and facilitated by guides we have prepared.

We have implemented these resources in undergraduate music courses and will discuss the outcomes of a single case study herein. We have also made these resources publicly available and open-source through the Electric Guitar Innovation Lab (EGIL) website. Our overarching goal is to provide members of the music technology instruction community with useful tools that open the door, even if just slightly, into this rarified area of digital signal processing.

Background

Digital signal processing (DSP) is a subject area in which musicians may find themselves in unfamiliar terrain grappling with abstract concepts about signal flow, waves, samples, and so on (Boulanger & Lazzarini, 2010; Pohlmann, 2010). Even music technologists may prefer to focus on existing implementations of such concepts as they relate to musicianship, such as the use of audio effects through plugins and hardware devices, and not necessarily "get under the hood." There is, of course, merit to understanding DSP and, with that, advantages to learning how to design unique effects and processes as opposed to strictly using the tools others have developed.

The notion of a programmable pedal is not completely new; Fractal (*Fractal Audio Systems*, 2006), Line 6 (*Line 6*, 1996), Kemper (*Kemper Amps*, 2012), Neunaber (*Expanse*, 2018), and many other companies have developed standalone hardware devices with on-screen controls and front-end software editors allowing users to explore the wide sonic range of their

products. This manner of effect editing, while usually called "programing," is technically constrained to the use of predefined processes inherent to these devices. The operations are similar to dragging plugins onto a signal chain inside of a Digital Audio Workstation and adjusting the parameters and order of each plugin. These devices and high-level software editors do not facilitate programming novel effects outside of these constraints.

There are many excellent books and resources that explain audio processes; David Creasey's *Audio Processes* (Creasy, 2016), for instance, presents the fundamentals of DSP in a straightforward manner with an abundance of flow charts and other useful figures that illustrate the abstract concepts associated with signal flow. Similarly, development environments like Cycling '74's Max programming language (*Cycling '74*, 1997) allow individuals to implement audio processes in a programming environment using similar flowchart-style visual representations. Well-known books like *Electronic Music and Sound Design* (Cipriani & Giri, 2019) combine the approach to learning DSP with the fundamentals of programming through Max. In 2011, Max 6 introduced Gen, a specialized low-level operating DSP portion of the Max programming environment, that allowed users to export their signal processing programs as native C or C++ code. This enhancement opened the potential for users to leverage Max as a prototyping domain whereby exported C/C++ code could then be imported into other non-Max environments toward the further development of mobile audio applications, audio plugins, and firmware for hardware systems.

Around this same time, programmable microcontrollers became more ubiquitous and user-friendly; among them are Arduino (*Arduino*, 2005) and Bela (*Bela*, 2016), two popular "maker" environments that use community-developed tutorials and code to facilitate the creation

of new hardware devices. Currently, for those less-interested in a focus on hardware, devices like the Owl Pedal by Rebel Technology (OWL Pedal, 2017), a small footswitch-controlled programmable blackbox, allow users to develop signal processing code on their computer through a variety of languages including Max, and compile and upload the code to the device's internal microcontroller where it can run without the need for computer connectivity. Electrosmith, makers of the Daisy Seed microcontroller, (*Daisy Seed*, 2018) also sells a fully-built stompbox-style device known as the Petal (*Petal*, 2020) which is comparable to the Owl Pedal.

As streamlined as the transition from software development to hardware implementation is becoming, there are cost-prohibitive entry barriers to purchasing dedicated hardware in addition to the obvious knowledge barriers all along the way (DSP, audio programming, and hardware integration and assembly, etc.). An Arduino microcontroller costs about $30 USD before the purchase of any additional components like toggles, footswitches, and so on; Arduino's "Student Kit" costs about $75 USD. Similarly, the Bela "Mini Starter Kit" costs about $170 USD, and the fully-assembled and ready-to-be-programmed Owl Pedal costs about $270 USD. Electrosmith's Daisy Seed costs about $30, and their Daisy Petal device costs about $335 USD.

In a classroom context, the use of a fully-assembled hardware device like the Owl Pedal or Daisy Petal would allow a student to focus solely on DSP implementation (only software) without having to divide their attention into foci that might diverge towards hardware assembly (soldering, identification of switches, toggles, other components, etc.) and an understanding of device firmware and microcontroller operating systems. Given the relative cost of a required

course textbook, a student or instructor may instead find it reasonable to require the purchase of a microcontroller or a microcontroller kit ($30+ USD) as part of a course related to digital audio processing for music; a fully-built hardware device ($270+ USD) may be viewed as cost-prohibitive despite the availability of free (open) documentation, tutorials, code, and other available resources.

Methodology

Our methodology had two major areas: (1) the development of resources—specifically, tools and workflows—to facilitate immediacy in effect pedal development; and (2) a case study through which a team of undergraduates without prior effect-making experiences were asked to use the resources to assist their efforts in making an effect pedal.

Software Tools - FX Testing Rig

Surveying the available software-to-hardware workflows, we identified the Max environment as having the most accessible programming language for novice programmers given its robust support through an online forum community and documentation. Max also allows more experienced programmers to implement code written in other languages within its environment. One potential hurdle of DPS programming in Max and, largely, any programming language for students, is the starting point. Before a student can begin to implement DSP code they must develop tools, workflows, and additional code to allow audio and audio files to be routed in and out of the DSP environment. And there is a hurdle developing the requisite routines for interacting with the DSP code in a way that is analogous to using physical toggles and controls of a hardware device. To this end, we developed the FX Testing Rig (Manzo, 2020): a

software tool that considers the starting point of the user and puts the necessary functionality at their disposal from the onset, bypassing hardware and I/O concerns, thus allowing the user to focus solely on DSP design.

Figure 1 shows a typical tremolo algorithm one might encounter in an audio processing book. A figure like this may seem abstract and unfamiliar to a music student, but, when explained via typical music technology instruction, such diagrams become incrementally less esoteric.

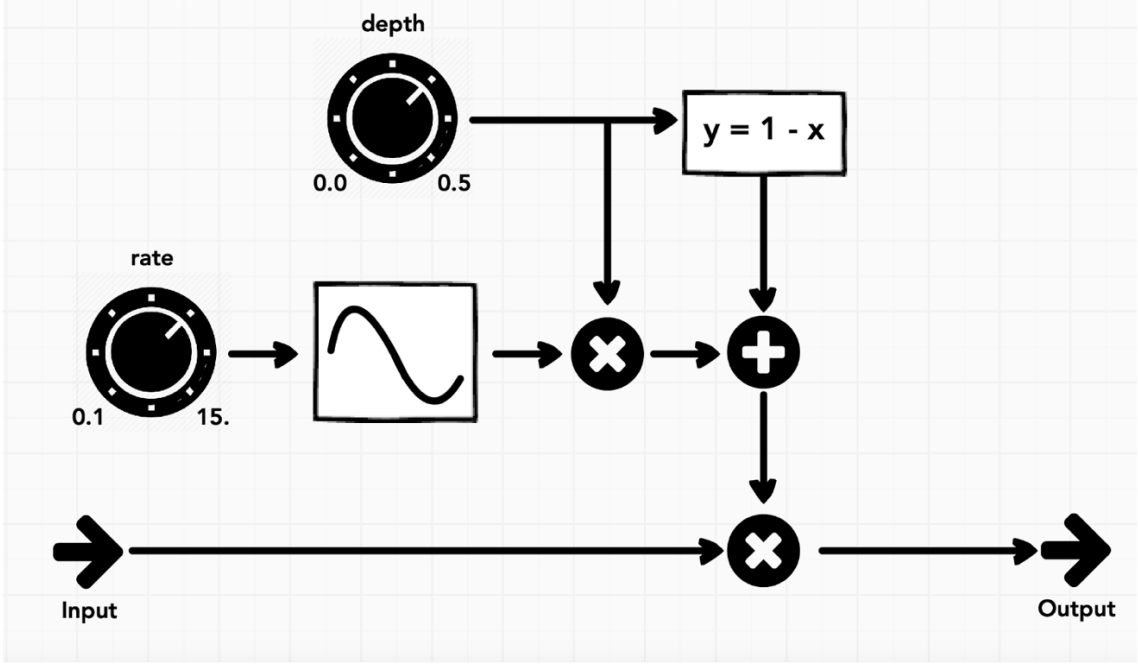

Figure 1: *A tremolo algorithm.*

At the next level, through the lens of Max programming, we can compare signal processes in such figures to programming objects, whereby a student more tangibly sees the

relationship between a sequence of theoretical audio processes and an implementation of similar programming functions, as illustrated side-by-side in Figure 2.

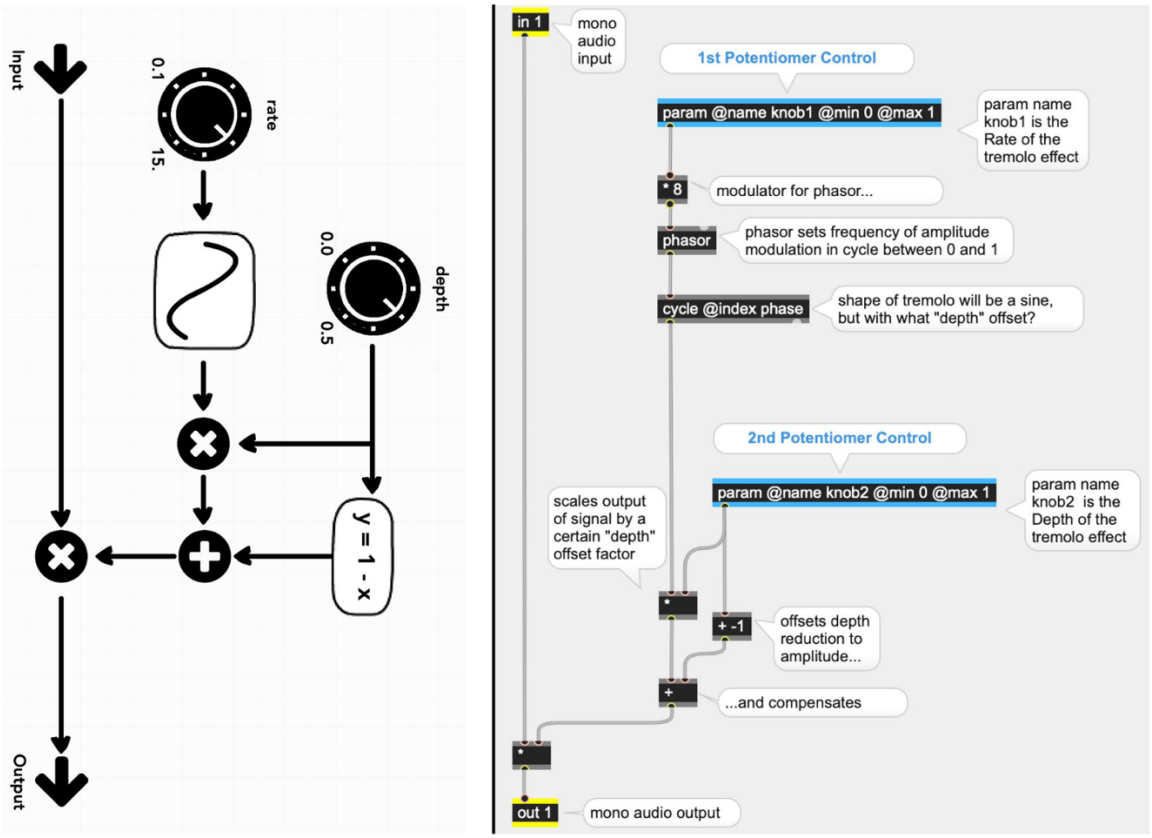

Figure 2: *A tremolo algorithm (left) alongside a programmed tremolo implementation (right)*

Our FX Testing Rig, thus, was designed so that students need only focus on this translation of abstract signal-flow diagrams and figures, ubiquitous in many DSP texts and other resources, into Max objects through their Gen programming subset of tools.

Our FX Testing Rig tool (Figure 3) is programmed to allow live audio input and audio recordings of guitar samples (direct-recorded single notes, chords, and patterns) to be passed

through to the Gen DSP subpatcher, the primary area of focus for the user. In principle, any effect that works inside of the Gen patch will be able to be pushed to a microcontroller with a few simple operations. Therefore, again, the focus in a pedagogical context can remain on the DSP and not the development of infrastructure to test and simulate the interaction with the DSP. The switches, toggles, and knobs provided in the patch are analogous to (one-to-one associated with) the hardware switches, toggles, and knobs that will ultimately be connected to the microcontroller. In the FX Testing Rig, these parameters are labeled with names that, when compiled, will map to the corresponding pins on the microcontroller, so, in short, using the virtual switches, toggles, and knobs in the FX Testing Rig as controls for a developed effect will allow the corresponding hardware controls to control those same effect parameters. As a student develops their effect through this designated portion of the FX Testing Rig, they can play guitar recordings or an actual guitar through their effect and interact with it through these controls. This audition process allows the user to debug and extensively refine their effect before committing to hardware implementation.

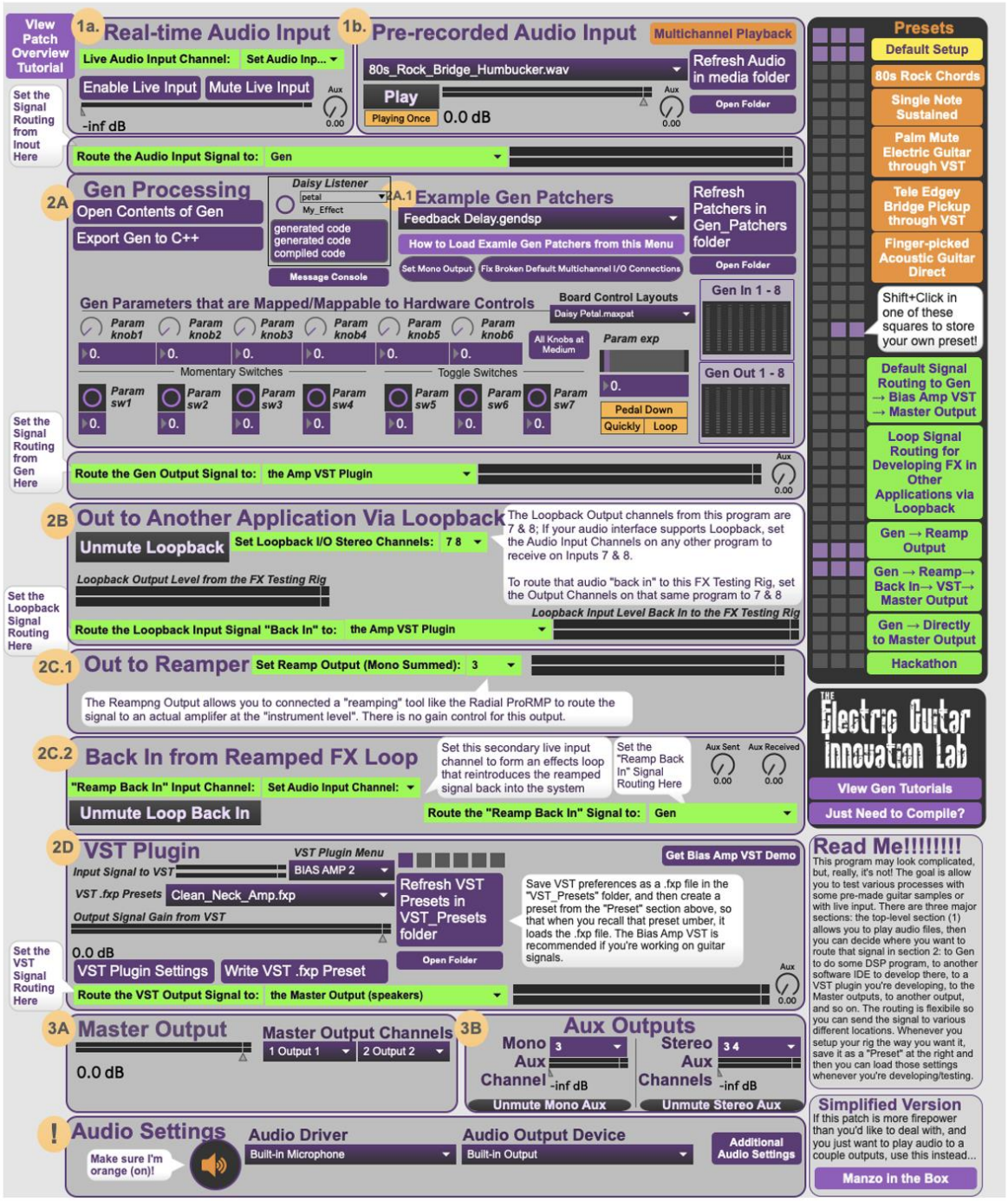

Figure 3: *FX Testing Rig, a QuickStart tool for effect prototyping*

The FX Testing Rig has flexible audio routing so that a student can run the output of their effect into a Virtual Studio Technology (VST) plugin like Bias Amp (*Positive Grid*, 2014) or Guitar Rig (*Native Instruments*, 2004) in order to simulate the way an effect might sound through an actual amplifier. A student could also reamp the output from their computer to an actual amplifier if such hardware is available. Several preset audio workflows are provided in the FX Testing Rig through a graphical preset menu, and students working with this tool can tailor the audio routing and other settings to their liking and save the configuration to the preset menu.

Our FX Testing Rig uses a modified version of Electrosmith's Oopsy package to support the export of Gen code to the Daisy Seed microcontroller. We optimized the FX Testing Rig tool to work with the Daisy Seed by default, though it could be readily adapted to work with other microcontrollers.

Hardware Guides

The Petal and the Owl are expensive devices, so we sought to develop a guide whereby a student could obtain a modest list of validated hardware components and work through a simple step-by-step tutorial to integrate those components into a device capable of loading the code exported from the FX Testing Rig. Given its cost and the flexibility of its support for Max, we chose the Daisy Seed as the ideal microcontroller for the hardware portion of our resources.

To implement connectivity to the Daisy Seed Microcontroller, with particular consideration for music students that might be new to soldering, we identified an off-the-shelf printed circuit board (PCB) developed by Pedal PCB called the Terrarium ("Terrarium," 2020) which seamlessly interfaces with the Seed. As noted, Electrosmith developed a set of programming resources for Max called the Oopsy package which facilitates the exporting of

code from Max to the device firmware. We modified this package to support the Terrarium PCB as an intermediary device.

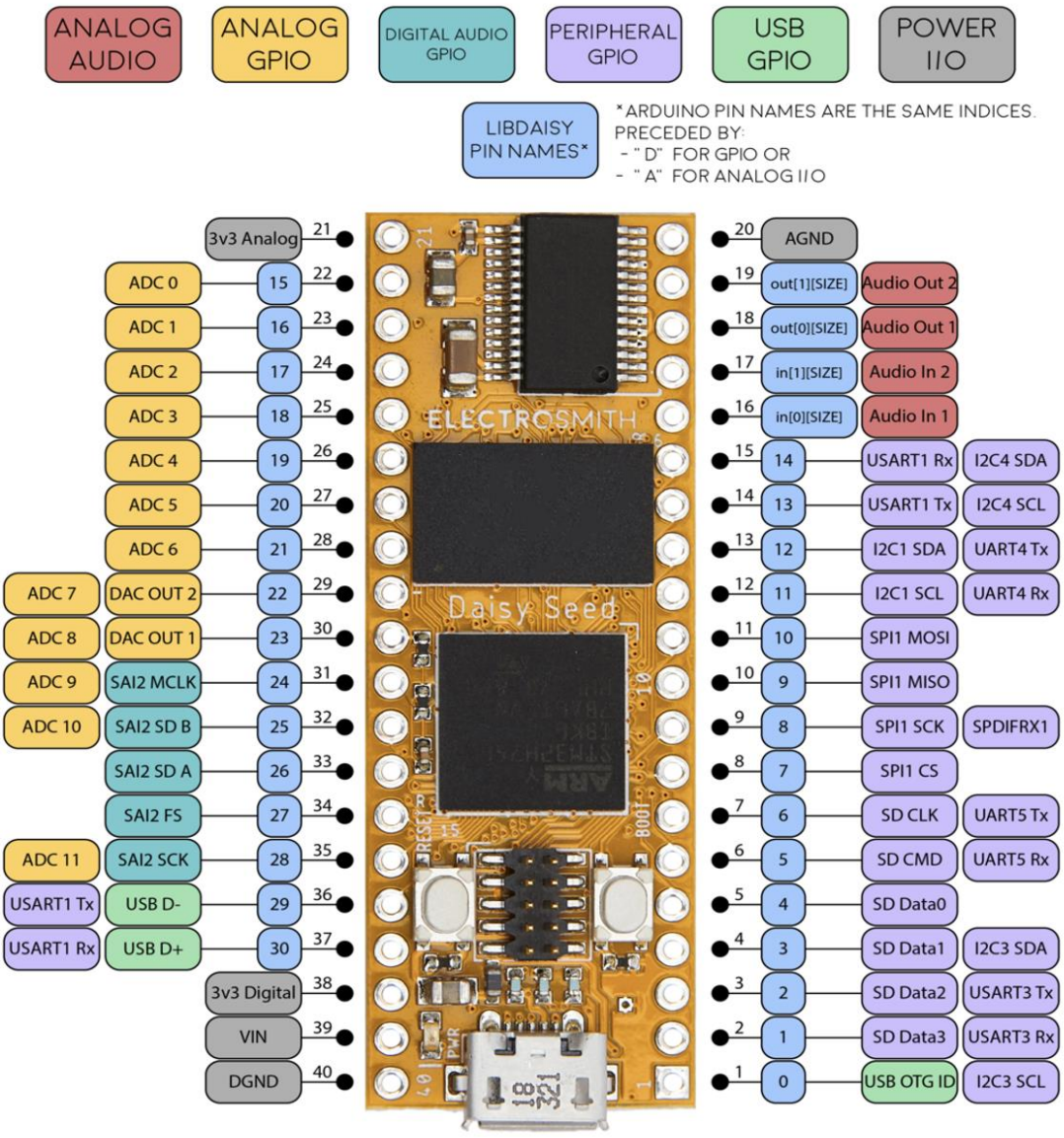

Figure 4: *Daisy Seed microcontroller with available pin connectivity*

In summary, we developed a step-by-step video guide showing the assembly of a Daisy Seed, the PCB, knobs, switches, and toggles into an enclosure (which may be machined or 3D printed). We've made all of these resources openly available at our Electric Guitar Innovation Lab (EGIL) website: http://ElectricGuitarInnovationLab.org/pedal

This portion of our website includes a link to download the FX Testing Rig software, links to further resources (books and websites) on signal processing, and step-by-step guides for setting up the software tools, building a few example effects including the aforementioned tremolo effect, and assembling the pedal. The FX Testing Rig portion of our website is regularly expanded by our team and others in our lab community to include additional effects, most of which include comments to explain, in accessible terms, how a particular effect works. Students can, accordingly, look at the way several effects function in code and use that as a point of departure for adaptation.

Case Study

Once we prepared our resources, we piloted their efficacy as a standalone resource in an undergraduate music technology course. A team of three students completing a music capstone project participated in the project and were asked to listen to Les Paul's arrangement of the song "Lover" (Adams, 2012), identify the audio processes employed in the composition, and explore the digital signal processes required to recreate that effect in the FX Testing Rig.

For those unfamiliar with Les Paul's arrangement of "Lover," he explored the concept of pitch change that resulted from recording to tape at different rates. The resulting effect in his recording are guitar sounds, recorded at the normal playback rate, sounding alongside guitars that sound one octave higher than the original. Students in this pilot were asked to identify this

phenomenon and explore approaches toward implementing these effects in a signal processing domain with the understanding that an effect that could run successfully inside of FX Testing Rig could later be easily exported to run on a device built by following the hardware resource guide. In addition to the aforementioned resources we provided, students were encouraged to explore additional digital signal processing texts and online forums (including the Cycling '74 Max forum); their work toward completing these objectives was otherwise self-directed.

The students were all undergraduate students with varying music and music technology proficiencies, though all students were largely unfamiliar with digital signal processing as well as Max and Max-style programming environments.

Results

Over the course of a single seven-week term in spring 2021, these three students were able to successfully build an effect that, when a footswitch was pressed, shifted the pitch of an incoming signal up one octave as in the "Lover" recording. The students included an additional footswitch control to shift the incoming pitch down one octave.

The PIs observed that students were able to speak articulately about their approach to both developing the effect algorithm and implementing meaningful performable controls over various parameters. A video of their final presentation further demonstrates their understanding of the underlying processes required to shift a pitch some number of semitones (Manzo, 2021).

As an added outcome of the students' exploration into the work of the arranger, the students learned about other effects Les Paul pioneered at the time, including the flanger.

Accordingly, they implemented the flanger and other "period" effects into a stand-alone multi-effect pedal with all of the associated hardware parameter controls.

Broader Impact

During the summer of 2021, the PI's worked with another group of undergraduate students to assemble 18 effect devices using the hardware guide. An adapted version of the Les Paul "Lover" multi-effect pedal from the undergraduate team (shown in Figure 5) was then loaded onto each of these completed effect pedals and used in a free summer camp for middle school students focused on music technology innovation. Students in this camp were able to use these effect pedals in recordings and performances of their own compositions.

Future Work

As noted, our team regularly expands these web resources and we welcome contributors through our open-source offerings. With each new course offering of a project similar to the "Lover" project, there is the potential to expand the FX Testing Rig with a newly developed effect each time a course is taught.

In April 2022, our team hosted a pilot of our inaugural "FX Pedal Hackathon" (2022), an all-day student event where teams of undergraduate students collaborated to make interesting effects. The resources used for this event as well as some of the students' code have already been bundled in the FX Testing Rig software. Additional materials and outcome information is published at our website: http://ElectricGuitarInnovationLab.org/hackathon

Figure 5: *Control layout on a modified multi-effect pedal made using these resources*

Conclusion

Signal processing pedagogy has several counterbalancing issues. Conjoining the study of audio signal processing both as an algorithmic and hardware realization has the added bonus of making signal processing tangible; students can get their hands on an effect they have implemented. But conversely, partitioning the study of audio signal processing into these separate activities can simplify basic understanding of the subject. In our approach, we both

integrate tangibility through physical implementation of hardware effects and we provide tools that expedite creativity in the software domain, separate from some of the burdensome engineering tasks associated with hardware.

The FX Testing Rig is an efficient means of prototyping effects processes in a virtual, readily modified and adaptable environment. The predefined mapping to the Daisy Seed microcontroller alleviates students' need to master the complexities of hardware connectivity and implementation.

Our initial case study provides a compelling proof-of-concept regarding the viability of the platform as a pedagogical tool, particularly within the time constraints of an academic semester or term. Further instructional iterations will help inform refinements and new directions for this integrated effects design environment.

Our open-source approach further advances the pedagogical impact of this system and associated resources by allowing the broader community to benefit from and contribute to the future development of this initiative.

References

Adams, Will. (2012, January 12). *Les Paul – Lover – 1948* [video]. Youtube. https://www.youtube.com/watch?v=q6VHlqH4-xs

Arduino. (2005). https://www.arduino.cc/

Bela. (2016). https://bela.io/

Boulanger, R., & Lazzarini, V. (Eds.). (2010). *The Audio Programming Book*. The MIT Press.

Cipriani, A., & Giri, M. (2019). *Electronic Music and Sound Design—Theory and Practice with Max 8—Volume 1 (Fourth Edition)*. Contemponet.

Creasey, D. (2016). *Audio Processes*. Routledge.

Cycling '74. (1997). https://cycling74.com/

Daisy Seed. (2018). Electro-Smith. https://www.electro-smith.com/daisy/daisy

Expanse. (2018). Neunaber Audio. https://neunaber.net/pages/expanse-user-guide

Fractal Audio Systems. (2006). Fractal Audio Systems. https://www.fractalaudio.com/

Guitar Rig 6 Pro | Komplete. (2004). Native Instruments. https://www.native-instruments.com/en/products/komplete/guitar/guitar-rig-6-pro/

Kemper Amps. (2012). https://www.kemper-amps.com/

Line 6. (1996). https://line6.com/

Manzo, V. J. (2020). *FX Kit—An EGIL Digital Effects Pedal Platform*. The Electric Guitar Innovation Lab. https://electricguitarinnovationlab.org/pedal

Manzo, V. J. (2021). *Les Paul—Experiencing the Innovative Process*. The Electric Guitar Innovation Lab. https://electricguitarinnovationlab.org/lespaul

OWL Pedal. (2017, August 30). *Rebel Technology*. https://www.rebeltech.org/product/owl-pedal/

Petal. (2020). Electro-Smith. https://www.electro-smith.com/daisy/petal

Pohlmann, K. (2010). *Principles of Digital Audio, Sixth Edition (Digital Video/Audio)* (6th ed.). McGraw-Hill/Tab Electronics.

Positive Grid. (2014). Guitar Amp Simulator and Modeler Software | BIAS Amp 2. https://www.positivegrid.com/bias-amp/

Terrarium. (2020). *PedalPCB.Com*. https://www.pedalpcb.com/product/pcb351/

About the Authors

V.J. Manzo is Associate Professor of Music Technology and Cognition at Worcester Polytechnic Institute (WPI). He is a composer and guitarist with research interests in theory and composition, artificial intelligence, interactive music systems, and music cognition. V.J. is author of several books published by Oxford University Press including Max/MSP/Jitter for Music, Foundations of Music Technology, and co-author of Interactive Composition and Environmental Sound Artists. He has created numerous software projects including the Modal Object Library, a collection of programming objects to control harmony in algorithmic and electro- acoustic

compositions, and EAMIR, an open-source project and non-profit charity organization that supports composition, performance, education, and research through accessible technology-based musical instruments. He is the founding director and principal investigator of the Electric Guitar Innovation Lab (EGIL) at WPI, and a co-director of the Media Arts Group Innovation Center (MAGIC) at WPI.

Ryan McKenna holds a BS in Electrical Engineering from Worcester Polytechnic Institute (WPI). The technical basis of his expertise is further refined though building a diverse set of organizations and products. Ryan currently works with Veo Robotics, developing a 3D vision-based safety sensor system for industrial robotics, and as an Affiliate Research Associate in the Electric Guitar Innovation Lab at WPI. Ryan's professional and creative development is motivated by a concentration on the communicative and productive potential of audio as both a transient and recorded medium. As an experienced musician, technician, and production designer in the touring music industry, he has delivered memorable experiences to concertgoers and listeners through performance, audio-visual displays, and the creation of emotive tools for musicians and technicians alike. Concurrent work in audio engineering and sound design for music and video yields experience in creating attention-grabbing content, tailored to cut through a cacophony of media.

Matthew Halper is Professor of Music at Kean University (Union, NJ). He has received performances in leading venues such as Lincoln Center's Alice Tully Hall, Weill Recital Hall at Carnegie Hall, and live on Chicago Radio and Public Television. He received a Whitaker Reading Prize from the American Composers Orchestra. His String Quartet was awarded the Walsum Prize and premiered by principals of the National Symphony Orchestra. Recordings include the release of his Concerto for Flute and Wind Ensemble (TROY821) which the American Record Guide lauded as "ambitious, ... lyrically dramatic, majestic and broadly American in flavor." Dr. Halper holds degrees in Electrical Engineering, Applied Mathematics and Music Theory & Composition and has lectured on contemporary music, music technology and had his works performed at conferences of the CMS, College Band Directors National Association, the Society of Composers, the National Flute Association, and at various institutions including the Juilliard School. Recent performances of his music include the National Theater and Concert Hall in Taiwan.

