Chaotic System Module for Acoustic Signal Processing

A Design Project Report

Presented to the School of Electrical and Computer Engineering of Cornell University

In Partial Fulfillment of the Requirements for the Degree of

Master of Engineering, Electrical and Computer Engineering

Submitted by

Liam Michael Sweeney

M.Eng. Field Advisor: Van Hunter Adams

Degree Date: May 2023

Abstract

Master of Engineering Program

School of Electrical and Computer Engineering

Cornell University

Design Project Report

Project Title: Chaotic System Module for Acoustic Signal Processing

Author: Liam Michael Sweeney

Abstract:

Mathematical systems that follow very different trajectories because of infinitesimally small differences in initial conditions are called "chaotic." These systems are certainly of interest to mathematicians, but they are also of interest to many physical scientists due to the chaotic nature of many natural processes. This project investigates whether they may also be of interest to musicians who configure modular synthesizers. These musicians do not "play" the synthesizer in a traditional sense, but instead configure the instrument. The musician configures probabilities for notes and note timing, and then the instruments generates music autonomously. This project developed a module for a modular synthesizer that uses a mathematically chaotic system to manipulate the instrument's configurations.

Executive Summary

During the Fall Semester of 2022, I, Liam Sweeney, with the guidance of Van Hunter Adams, set out to create a new type of modular synthesizer component. This module would incorporate pseudo-randomness in the form of mathematical chaotic systems to perform digital signal processing on waveforms to be acoustically produced. This project intertwines development and discovery as we aim to both discover the various elements of chaotic systems and how they may be used for acoustical wave creation and processing.

We used the modular synthesizer as the instrument with which we would explore musical chaos. These instruments feature an often-deep series of signal processing chains aimed at the creation of a waveform to output to speakers. While such a system is not readily available in the lab, one of the long-term aims of this project is to create a system that can fully integrate with common modular synthesizer components in a plug-and-play manner.

An RP2040 microcontroller acts as the central processing unit for the device. This microcontroller, programmed in C using its associated Software Development Kit, numerically integrates the Lorenz system and uses the chaotic trajectory to synthesize audio. Any chaotic system could be substituted for the Lorenz system, we chose it simply because it is a particularly famous example of chaos.

To facilitate the exploratory aspect of this project, the device is designed to flexibly map the chaotic trajectory to a variety of audio synthesis parameters. In particular, the user can associate the x, y, and z position of the integrated trajectory with amplitude, frequency, and audio output channel (left or right earbud) for synthesized audio. This led to some interesting discoveries and observations. However, direct mapping of these elements simply does not lead to the easiest possible interpretation for human ears. This mainly is caused by the interpretation of sound being nearly always relative. As a result, sound theory was applied throughout the project to assist in fully realizing these elements of the chaotic system under test. These modifications mostly came down to loudness and pitch values needing to be exponentiated to fully realize proper interpretation.

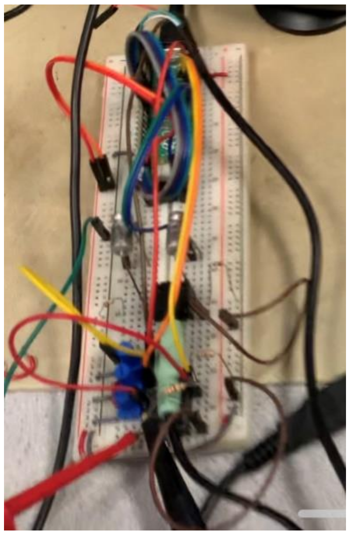
While the end-product of this project requires some modification for compatibility with commercial synthesizers, the existing prototype is capable of a compelling proof-of-concept demonstration of the theoretical module. With minimal further work, it is believed that this product can reach commercial standards for use in a modular synthesizer.

By the end of both semesters, their own respective demos were successfully built. By utilizing a MCP4802 Digital-to-Analog Converter (DAC), the system was able to play the output signal from the chaotic system's calculations and audial factoring, into a pair of speakers to play the output signal. For the final deliverable, the system utilized the Raspberry Pi Pico's inbuilt Analog-to-Digital Converter (ADC) to control the system parameters via input waveforms/signals and how exactly it utilized the chaotic system during runtime. This method of control is a core part of modular synthesis.

Overall, the system managed to produce vastly different and strange sounds as it came into full realization, and it is believed that there is a lot more left to be discovered if more work is continued to be done.

Introduction

The main interest within this project is the exploration of chaotic systems within acoustic space. The focus is placed on the implementation of the Lorenz system, one such chaotic system, while creating the infrastructure to quickly implement various other chaotic systems with ease so that the maximum range of chaotic systems may be explored. In addition to the direct acoustical properties being explored, I aimed to discover various ways to integrate the system in interesting and appealing ways. Throughout the project, I strived to create 'random' acoustic outputs by utilization of chaotic systems while avoiding simply becoming white noise. Throughout a majority of the project, the building of the system kept the possible final product in mind of creating a commercially viable product that utilizes the discoveries we made in a practical package that can be implemented as an individual module within a standard modular synthesizer system, however the project in its final form did not reach this goal.



Picture *1: Final Demo build of the system*

Background

Firstly, what is important to understanding this project is chaotic systems. Chaotic systems are a set of parameter equations which are hyper-sensitive to initial conditions. Chaotic systems lay in a middle ground between more orderly systems and random systems where chaotic systems share the element of determinism of orderly systems. This determinism inherently makes the system not random. However, the chaotic system differs from orderly systems as it is highly divergent and as such, it is impossible to be certain where it may end up without computing every step of the process. This is the primary element that makes a system chaotic; very small changes in the initial conditions will cause the system to branch off in wildly different directions eventually as the differences propagate. Overall, a chaotic system has no random element but instead, is a system so incrementally complex that it can simulate apparent randomness.

The element of chaotic systems that we are interested in exploiting is that the system remains continuous between various points. This provides favorable properties to allow for the system to directly map onto an acoustic signal, in which itself is continuous. In a purely random situation, each individual value would wildly jump all over the place as each value provided is independently sampled. And while it can somewhat synthesize if we utilize an integration of these random components as we explore later on, this is not our primary trajectory as it results in a heavily dissonant white noise while fully progressing.

Chaotic systems allow for an exploration of random-like elements while still being relatively well-behaved enough to still allow for possibly interesting acoustic sounds to still be able to be produced. This once more is the primary goal, to produce 'random' generated sound that can hold a value beyond white noise.

The second element that is conceptually important to be familiarized with is modular synthesizers. From a technical standpoint, modular synthesizers are a large array of various signal generating/signal modulating equipment strung together in series. Each individual equipment element of this array is referred to as a module, and they individually often only serve a simple selective function as they are designed to be inserted into a wider modular synthesizer and, as such, will not have to individually have too wide of a scope. Alongside a simple input/output, modules will often not only allow you to configure various parameter values through conventional means such as a turn dial, they allow you to input waveforms to act as the tuning function to that given parameter. Such setups are often referred to as an invisible hand, such as the signal invisibly turning the dial.

Due to this more-or-less loose guideline for modular synthesizers, it does open the possibility of many different standards for how exactly the system should be specified and be used. Over the course of this project, the focus was to try and push the system towards a finished product under the popular Eurorack standards, and as such, various modifications to the project were made to meet these standards.

- ***Lorenz System***

For the first semester of doing this project, the focus was to do rapid prototyping so that we might be able to establish a working project initially within a narrow scope before expanding it to include all of the features that we would want to explore. One such narrowing of the scope is the limitation of chaotic system exploration to only a singular chaotic system, the Lorenz System.

The Lorenz system, or Lorenz attractor, is a model derived from a model for fluid convection to model the apparent unpredictable patterns of weather [1]. Resulting from this, we get the following system of equations:

$$\frac{dx}{dt} = \sigma(y - x)$$

$$\frac{dy}{dt} = x(\rho - z) - y$$

$$\frac{dz}{dt} = xy - \beta z$$

The exact meaning behind each element of this equation is irrelevant in terms of this project. This system was merely chosen as our initial system due to it having somewhat regular behavior and for it being very well known as a chaotic system.

Changing around the initial conditions of the system, as discussed above, can significantly change the trajectory of the system. This also pertains to the constants being chosen within the system as well. Overall, the most widely used set of constants are:

$$\sigma = 10, \rho = 28, \beta = \frac{8}{3}$$

If we take these set of parameters and set it into phase space, tracing the position of all three parameters as they change over time, we end up with a shape like this:

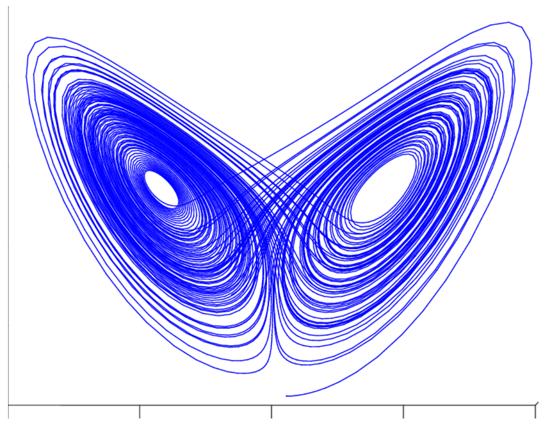


Figure 1: *The Lorenz system plotted in phase space[2]*

From this, we can see an almost periodic system that is also constantly shifting around. Plotting each axis individually across time, the resulting graph shows us a similar, almost periodic behavior.

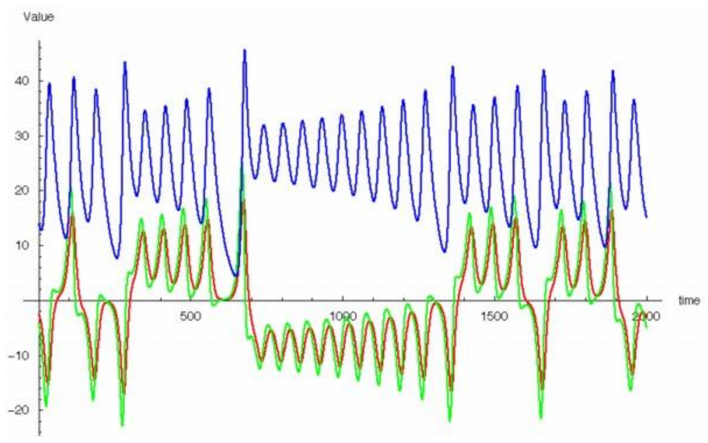


Figure 2: *The Lorenz system plotted across time for each individual variable (red x, green y, blue z) [3]*

With this system allowing for relatively bounded values in mind, I started the process of creating a system that can allow the user to both model and hear the system of equations.

- ***Sound Theory***

Sounds are generated by exciting pressure waves which propagate through a medium. In terms of speakers in open air, this is achieved by having a large diaphragm oscillating outwards and inwards. As the diaphragm shoots outwards, it creates a high-pressure wave of air shooting outwards from the speaker. Then, as the diaphragm goes back in, it creates a segment of low pressure as air rushes in to fill the now opening space, creating pressure below what could be called a steady-state air pressure before the diaphragm shoots back out and creates another wave of high-pressure air.

As the high-pressure segments attempt to disperse into occupying the low-pressure segment in front of it, the air particles pick up inertia as it rushes to fill up open space. Overall, the direction of this ends up pointing outwards from the sound source due to the initial inertia of the wave (if the diaphragm creates this pressure wave pointing a certain direction, that direction would be the overall vector of the wave). And as each high-pressure wave attempts to fill in the low-pressure segments, it ends up overcompensating, creating the previously high pressure segment now low pressure and the surrounding segments high pressure. Placing this behavior into a more continuous space as is reality, this results in the classical image of sound waves propagating through space.

To properly scale the volume of the system to best fit for a scaling parameter within this project, some further understanding of the sound wave was applied. A lot of human interpretation of sound is based around relativity and exponentials. For a pitch to go up or down an octave, you need to double or half your frequency respectively. So, if you go from a note A4, which is 440Hz, and you want to go to A3, you will go down to 220Hz, whereas if going up to A5, you go to 880Hz. That is a 220Hz decrease vs a 440Hz increase to navigate via the octaves. A similar thing also applies to the interpreted loudness of a sound. This is why the loudness of a sound is usually said to be in decibels, which logarithmically compresses the amplitude of a sound wave so that the number is somewhat interpretable by often staying within a certain range of values. A basic decibel conversion may consider the following equation:

$$dBm = 10 * \log_{10}\left(\frac{P_{meas}}{P_{base}}\right),$$

where P represents the power of the signal. However, we are not really interested in the sound power itself, but rather the pressure being carried by the sound waves (dBm is the notation representing the decibels of a sound's power). Even though the units cancel out within the interior division, it is important to discern how pressure is related to power to maintain this formal definition of sound power. Firstly, if we derive the sound intensity, from any given point, we find:

$$I = \frac{P}{A},$$

where A represents the area in which the sound has dispersed in (usually denoted as m^2 as we dictate it by the radial circumference in open air from the sound source). This is usually known as the inverse square law, as the power gets dissipated as the area of propagation increases. However, in terms of being able to properly pan a signal, what we are truly after is the physical measurement of the sound wave's propagation – pressure waves.

Pressure waves are related to sound intensity from the following formula:

$$I = \frac{(\Delta p)^2}{2\rho v_w}$$

where Δp represents the amplitude of the sound wave's pressure, ρ is the material's density where the sound is propagating and v_w is the speed of sound in that material. With ρ and v_w being independent of the pressure amplitude, we can then make the assertion:

$$I \propto (\Delta p)^2$$

This assertion is important because now we can find the decibels for sound pressure:

$$dB\ SPL = 10 * \log_{10}\left(\frac{p_{meas}^2}{p_{base}^2}\right) = 20 * \log_{10}\left(\frac{p_{meas}}{p_{base}}\right)$$

where we are simplifying the Δp notation to p, and SPL stands for Sound Pressure Level.

Now the relativity of the system comes into play with p_{meas} and p_{base}. p_{meas} simply represents the pressure of the sound wave you are trying to measure, and p_{base} represents a constant pressure point of reference. Depending on what system you are currently using this value changes, such as dBu and dBv being voltage decibels but with different reference values (0.775V and 1V respectively). For SPL, this reference point is 0.000002 pascals (N/m^2), which represents the threshold of hearing. As a result, there is such thing as negative SPL dB sounds, but they are considered outside of human hearing.

The key reason why SPL is important to be considered is for panning, we need to consider what happens when we have two speakers playing the same sound. Assuming that there is nothing weird happening where one speaker is significantly more distant than another, we can assume that the sound waves constructively interfere. In other words, the pressure waves physically add their pressure waves together at any given point, giving the additional amplitude. However, we need to be extremely cautious when we consider what this means. Putting 10 dB SPL out one speaker and 80 dB SPL out another does not result in 90 dB SPL total, in fact it hardly changes from 80 dB SPL. Even with two 80 dB SPL soundwaves, when they perfectly constructively interfere, they only double the pressure, resulting in 86 dB SPL due to a doubling of pressure resulting in only +6 dB SPL in accordance to the equation above.

What this results in is there is a non-linear relationship within dB to worry about when it comes to panning, which is an issue we map voltage into the dB space as well, so while not including the panning factor is easily enough to do this as the voltage waveform maps to sound pressure, allowing for a linear division between the panning factors.

To deduce the voltage relationship with dBm, if we consider the relationship between voltage and electrical power respectively, and with consideration that the speaker is a passive circuit element, we find that:

$$W = \frac{V^2}{R}, dBu = 10 * \log_{10}\left(\frac{V_{meas}^2}{V_{base}^2}\right) = 20 * \log_{10}\left(\frac{V_{meas}}{V_{base}}\right)$$

As a result, the voltage supplied is proportional to the sound pressure being generated. This allows the deduction that the voltage waveform being supplied to the speakers maps to pressure supplied, meaning that this system can be used to map voltage in dB elements, allowing for the proper scaling of values.

The main reason for this modification is for a more interesting range of audial sound. Without this conversion, this means that half of the values would effectively be contained in the loudest 6dBu of the signal. As a result, bearing in mind the relativity of sound interpretation as mentioned in the start of this section, having this system be able to vary on the dB axis does serve as an important element to image the chaotic system's sound.

- ***Direct Digital Synthesis***

To create the sound to play from our system, it first must be synthesized. By utilization of a Digital to Analog Converter (DAC, see **Design – Fall Semester 2022 – Digital to Analog Converter**), it can take in a value and output a corresponding voltage signal value. This can be utilized to formulate a discretized waveform that can then be interpreted as a sound once it is used to drive a speaker. The simplest way to go about this is to utilize C's variable overflow behavior, which creates a cyclic pattern, similar to that of a periodic sine wave. The key element of this overflow behavior is the number naturally cuts the number off when it overflows yet continues to set it to the remainder from the overflow.

Direct Digital Synthesis is the process of directly mapping the length of a singular sine wave period to the range of values in which the C variable can reach before overflowing. This method is useful as it can be used to create a lookup table in which certain ranges of values for the C variable can map to pre-computed values for sine, helping to keep the computational cost lowered for the sake of synthesizing in a time-critical manner. From this point, we can add a value to the accumulator variable corresponding to the frequency given by the sampling rate (effectively how frequent this addition occurs) versus how many times this value takes to trigger an overflow. As a result, we arrive at the overall equation:

$$N = \frac{F_{out}}{F_S} * 2^{32}$$

Where N is the desired increment amount, F_{out} is the desired frequency, F_S is the sampling frequency, and 2^{32} represents the overflow state of C integer variables. With the DAC in question having a maximum sampling frequency at 40kHz, finding N becomes a trivial computation.

One important item to note is that making a lookup table for all 2^{32} entries is infeasible. As a result, we can only create indices based on the more significant subdivisions of the variable. To keep the system fast, we chose to set the cutoff point to be bit-aligned. Specifically, we chose to make the system index off the 8 most significant bits. This allows for 256 entries, which is more than sufficient for the exploration in this project and can easily be changed later.

Building off this framework, it becomes apparent that the simplest implementation for this project would be mapping the chaotic system parameters to frequency fluctuations rather than dictating the exact frequency. With a parameter to ensure the system is scaled properly, the value of the chaotic system effectively dictates how much should be added or subtracted from N every single timestep by changing the F_S term each time. This changes the rate at which the variable can accumulate and as a result, becomes frequency modulated.

Problem Statement

The primary objective of this project is to do an auditory exploration of various chaotic systems and from this, extrapolate how I may utilize this to create a synthesizing module with enough intrigue to justify its existence. Through exploration, I will figure out how exactly one might best use this module and design it in a way that a music producer would have reason to want this module in their ensemble.

Design – Fall Semester 2022

For the operation of the system, we are utilizing the Raspberry Pi Pico development board which features the RP2040 microcontroller. This is a microcontroller setup that was released January 2021 by the Raspberry Pi Foundation. Alongside with the microcontroller being readily available while working with Prof. Adams as a project advisor due to his ECE 4760/5730 course and me having increased familiarity with the device from said course, the Raspberry Pi Pico offers many other advantages. The microcontroller setup is incredibly cheap, being able to be purchased for only $4, allowing for better budgeting and potentially profit margins when framing the project as a mass-produced product. Additionally, the device has plenty of I/O pins for potentially large expansion of input signals for magic hand interfacing with the rest of a given modular synthesizer setup. And finally, the Raspberry Pi Pico also features a well-documented low powered state, in which could be nicely utilized for when our specific final product module is not in use, as especially as modular synthesizer setups grow bigger and bigger, it becomes highly likely that not all modules are active while the system is in use.

Throughout the first semester of working on this (Fall 2022), the primary objective of this project was to generally figure out what a chaotic system might even sound like and start setting up the Raspberry Pi program infrastructure so that the device may be able to change whatever parameters of the system we please at any given point of time during runtime. This is due to the desire to set up a magic hand system later in the project's progress. Utilizing the Lorenz system as a starting model, the mapping of the system aims to correspond to various elements of sound, or more generally, the output waveform.

- ***Digital to Analog Converter***

To properly create this waveform, the simplest way of implementation is by utilization of a Digital to Analog Converter (DAC). This device can take an input discrete digital encoding of a wave's current value within the amplitude and map it to a voltage level representative of this value. This then allows us to easily output an analog wave for a speaker system to be powered by.

For this project, the MCP4802 integrated circuit was chosen. This, in part, was due to the ECE 4760/5730 availability, alongside overall having a chip setup that works well with the Raspberry Pi Pico. This mainly comes down to the SPI communication interface for which the Raspberry Pi Pico has an interface already setup for, alongside offering dual output signals to be controlled independently, allowing for a simple L/R panning of the system.

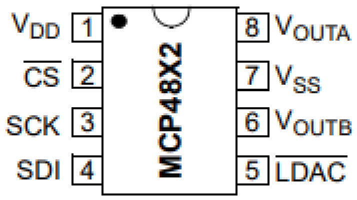


Figure 3: *MCP4802 pinout footprint*

For the initial placing together for the project, it was initially put together with the MCP4802, however, depending on how we may want to expand it, we may choose to switch to the MCP4812 or MCP4822. The difference between these various models is the bit bandwidth of the input signal, varying between 8-bit, 10-bit, and 12-bit bandwidths respectively.

For this component, V_{DD} and V_{SS} are the power supply and ground respectively of the circuit. To operate successfully, 2.7-5.5V must be supplied to power and, as a result, the 3.3V (3V3) power supply from the Raspberry Pi Pico will perfectly suffice. `CS, `LDAC, SCK, and SDI are all parts of the SPI protocol, so setting it up within the programming of the Raspberry Pi Pico is well enough to get it set up in that regard. And finally, V_{OUTA} and V_{OUTB} are the dual-output terminals, each outputting the last communicated voltage level with respect to the V_{SS} ground. As a result, this signal holds at its given voltage until the next communicated update to the signal. The important thing to remember while this is the case is that sound is dictated by fluctuations in pressure and, as a result, idling at a certain voltage does not cause any sound, nor does it cause any weird behavior with the speaker as any given voltage only dictates the speaker's diaphragm position; it is the fluctuation of this diaphragm that causes pressure fluctuations in the air as it pushes out and back in. Despite this factor, however, the speaker does become strained while being held in this non-resting-state position, so it is important specifically with regards to speaker health to ensure that this does not become the case (See further corrections in **Design – Spring Semester 2023**)

- ***Sound Theory, Applied***

The primary auditory elements chosen for possible mapping of the system were loudness, pitch, and panning of the output. Electrically, this corresponds to amplitude, frequency, and division among the output channels respectively.

As briefly described in the **Background- Sound Theory** section, the correspondence between amplitude and loudness is non-linear, as it is aimed to be able to scale this loudness by dB. This use is for when the system is the final driver unit for the given system, such as this first semester's demo. Yet, it remains important that this system remains able to output this direct amplitude variance linearly where the primary focus of this application being on signal processing (if every module in a modular synthesizer chain did this transfer, it would lead to a large deal of distortion), both in terms of the exchanging of signals and for use in magic hands of the modular synthesizer system.

Additionally, there is a similar system that goes into place for the frequency as well. Due to each octave representing a doubling or halving of frequency, we can modify the signal from a base frequency by a factor of 2^x where x is the value of the variable that is currently mapped to frequency. While we do plan on having the input waveform being the primary base waveform, for the purposes of using this module as a source, we utilize a 440Hz sine wave, an A4 pure tone.

- ***Serial Interface and Value Mapping***

With the final user experience in mind of wanting a myriad of settings available, alongside with the interest of faster exploration of the chaotic module space, as part of the first semester's demo, the system can change the values of and remap different variables of the Lorenz system to various different elements of the given wave (loudness, pitch, and panning) during runtime. This was accomplished by setting up the variables for constructing the output waveform as C pointers that pointed to the various variables of the Lorenz system, allowing for the pointers to shuffle around during runtime for dynamic remapping.

The serial interface can be communicated with a UART connection and protocol, something Raspberry Pi Pico is also built to work with. From this, a terminal such as PuTtY can be utilized to showcase this serial interface. For the purposes of this project, the baud rate was matched at 115200.

Ultimately, the serial interface was set up to work as any other serial interface would with prompt messages and the user types in the element or value they desire. By leveraging it, it allowed future modifications and constraints to quickly be realized.

The main issues that had to be addressed was the balance between keeping the system in a state in which it is stable enough to not diverge while still not ending up collapsing in and converging to (0,0,0). As a result, there is certain values that need to be considered in both the method itself and the way that it gets calculated. Overall, to assist with the successful mapping of values to relevant values while still allowing for the system to remain in a set of values for it to be well-behaved, each element of the process is multiplied by a factor, adding another set of tunable parameters during runtime.

Although this system is currently in a serial interface state, the goal is to eventually intake these various settings of variables as input signals, to allow for the system to effectively have its own set of magic hand parameters to operate with. The exact configuration of how this will work is beyond this semester's scope of work and will be addressed in the coming semester.

Results – Fall Semester 2022

Overall, we were successfully able to get a full demo up and running for the end of the semester. This demo features mapping the Lorenz system being mapped with various weights to various audial attributes, to then be played via a two-speaker system. From this point forwards, we can discern how changing various elements of the system is able to change the system on an acoustic level.

Media 1: *Demo of the acoustical properties of Lorenz system given certain initial values.*

Interestingly, one of the elements that we quickly found as causing the system to collapse onto (0,0,0) was if we attempted to take small enough step sizes as we navigated the system of derivatives. As a result, to simulate a more variety of signals, we had to vary the effective sampling rate of the system if we wanted to operate the system in slower contexts of navigation of the curve. This, however, could instead be a result of a dead zone test, where we instead did not push the step size low enough to find another window that may work with the system as expected. However, this remains to be explored for the coming semester. For the moment, the system simply performs this by effectively skipping the SPI communication with the DAC by not iterating upon the Lorenz system as a temporary issue in place for continued exploration and will be further investigated in the coming semester.

Design – Spring Semester 2023

With the demonstration setup that successfully got assembled by the end of the previous semester, a great deal of the work throughout the following semester was done to sort out the bugs introduced by the rapidly created system alongside pushing the design to be in a more hardware focused build in terms of adding controls to the system via dials and input voltages (CVs) rather than communication through software via the serial terminal.

The bugs that required fixing ranged from an off-by-one indexing error regarding the panning of the system, creating a state in which at maximum panning it caused the system to suddenly have its signal amplitude shoot down to zero, creating for unfavorable artifacts in the sound system that should not be there. In addition, there previously was an issue where the system was strangely time-sensitive in its propagation of the Lorenz Attractor and the sound's synthesis. While not the cleanest solution, the system is able to solve this problem by simply skipping over a portion of when the system enters the repeated timeout function callback, resulting in the system having some overall timing flexibility.

While researching the system to meet with more standard synthesizers, such as Eurorack, one mistake in my previous design that I ended up realizing was that I incorrectly had the speaker system setup. While this should have minimal consequences on the system depending on the application and whether it is a part of a wider speaker system that accounts for this, it is still important to note. Specifically, I made the correction on the idea about the speaker not caring about a static DC load being applied to it. While it is true that the speaker does not produce any sound while it is in this state, it does still undergo strain as current continues to flow throughout the speaker's induction system. This results in its diaphragm being constantly being pulled inwards or outwards rather than being in its neutral resting place. While this alternative resting place of the system means that the diaphragm is not moving and, as a result, not making sound, it still puts continuous strains on the speaker and results in the speaker device taking unnecessary stress overtime that can lead to long-term issues. As a result, I added an AC-pass capacitor before the output signal is seen by the speaker, paired with the drop-down resistor, allowing for only the AC elements of the system to be seen by the speaker and no matter what DC voltage the system may rest on upon stopping, it will not cause any unneeded strain upon the speaker.

Additionally while figuring out how to work the system up to Eurorack standards, I quickly had to make a decision that the system would not be fully functional within these standards. Specifically, the Eurorack standards utilize banana cable connections which offer a mono signal while having cable junction designs that can fit securely, something highly desirable in large modular synthesizer systems as it is not uncommon to have a large sum of wires connecting various ports and any chance of wires becoming loose and the system breaking in this messy state is highly undesirable for user experience. While I agree with this evaluation, I did not build the demo within the span of banana connections. And while there is little dependencies towards the output wires in place (Of which, the final product has both direct connection spots via 3.5mm TRS connections or could operated via direct electronic tapping into the system), this realization caused for a string of decisions to be made, primarily the de-emphasis of panning onto the system as that requires a dual-signal output, something that while not unachievable, minimizes the exact caliber of chaotic system elements that can be expressed in a mono output system such as banana wiring if the system was to ever turn back towards attempting to fulfill Eurorack standards more fully.

While as a result, the system is unable to meet the Eurorack standards, it does not mean that the system as a whole did not attempt to approach meeting them. Various elements were put into place solely for the purpose of fulfilling these requirements, primarily in the form of implementing a resistor voltage-divider on any CV input to the system as the standard CV operates between 0-5V while the Raspberry Pi Pico can only handle an internal voltage level of 3.3V. While this could be elaborated further by the implementation of Zener Diodes to help assist the system to clamp the input voltage at this level in case voltages greater than 5V were being placed into the system, however these precautions were not put into place as these signal inputs are simply control voltages, nothing of the sort of feeding audio into the system in which there may be a bit more of a risk of voltages in regions greater than what the user may expect.

The way in which these CV works with invisible hands is by the utilization of a specialized wire input jack with normally closed connections. This effectively makes it so the output of the jack is connected to a direct input source via a pin-in connection as a default case. Only once the jack is plugged in does this connection with the default value becomes untapped and the output of the jack jumps to whatever is being inputted via the connection instead. This allows for the system to be able to easily utilize both potentiometer dials and control voltages to control the system, successfully getting the system in a controllable state in which the user can change the system's parameters during runtime.

Ultimately, the Raspberry Pi Pico only comes equipped with three ADCs that can be connected externally (with a fourth one in-built solely for temperature reading). Due to the digitalization of the signal, the idea of hooking up an external ADC was highly undesirable as it would quickly occupy all the microcontroller's pins. As a result, after consideration, only three of the system's values could be modified during runtime in this way. Through some experimentation and qualitative pondering, I eventually settled on the parameters being more focused on the system from a musical perspective. As a result, I setup the device so that the baseline frequency the system uses to modulate the chaotic system over is now an interpretable parameter for the system to use as the system can now make notes underneath the chaotic system's modulation, alongside forming a tempo parameter that controls the rate in which the ADCs of the system are read. Presuming that the pitch is constantly changing, the tempo allows for discrete steps of various lengths of time to be taken between readings of the various input signals, effectively making it so the system has perceived notes as each time the baseline frequency is sampled and changed. Finally, one of the scaling parameters that help transfer the effects of the chaotic system onto the frequency modulation were the best candidates to be chosen and for this instance, amplitude scaling was chosen as frequency already had two dials dedicated towards it and with panning being de-emphasized due to the lingering Eurorack standards, it left amplitude scaling as the clear choice for the third CV controlled element.

Results – Spring Semester 2023

Overall, this semester's work featured a lot of tugging as I tried to figure out exactly where the best place to take the project was. And while much was accomplished, it still resulted in there being a large amount of work that can still be done. Primarily, I can easily see three primary different pathways to take the project, being fulfilling Eurorack standards for commercial use, restructuring the way in which the audio signal is formed/doing audio analysis, and the exploration of other chaotic systems.

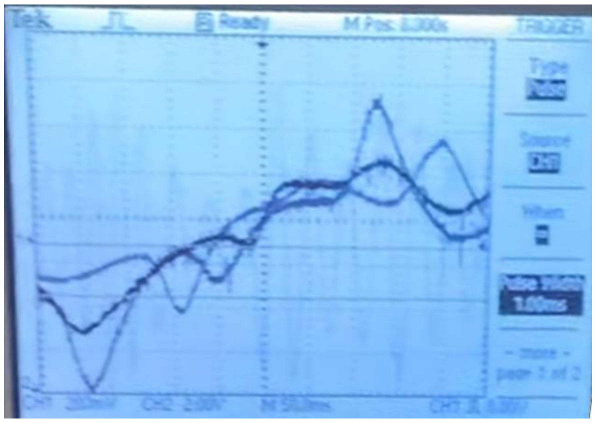
First is continuing work on getting the system to meet Eurorack standards so that it can formulate into an industry-grade modular synthesizer that can interface with other synthesizers available on the market with ease. This leap primarily has the requirement of restructuring the inputs/outputs into the form of being able to utilize banana cables alongside soldering and creating the packaging of the system into the applicable unit size once these additions are made.

Alternatively, it became increasingly apparent throughout experimentation that there may be a more direct way to create synthesized sound from a chaotic system rather than having to modulate the system over a baseline frequency waveform. This would require a reformatting of the system; however, it can also allow for the chaotic system to be directly observed in a greater way as the chaotic system would be able to take up the entire sound rather than only effecting the spectral elements of it through implicit harmonics/etc. As a result, if this route is not taken, the chaotic systems could instead be further investigated by doing audial analysis on the waveform such as FFTs. While this does not add anything to the audio waveform being observed by the user aurally, this could offer additional insights into the chaotic system or be used to diagnose any artifacts the program is inserting into the signal via these conversions if the result is not what might be expected.

Finally, one clear pathway to exploration would be trying to implement different chaotic systems into the program structure established. With much of the program structure being somewhat modular due to the previous semester's work to get each sound parameter remappable to various sound parameters, this made the program must be structured in a way that there is a level of isolation between both the chaotic system being evaluated and the synthesis of the sound being outputted. As a result, this path offers little resistance and yet little guidance as what characteristics of a chaotic system allows for it to be interesting in the scope of this program is uncertain, making the decision of what other chaotic systems to try to have as much baseless a reason as the choice of doing this project on the Lorenz Attractor had.

In terms of the results captured within this report, it took until later into the semester that I realized that I was putting too much focus on the creation of the system as a musical device in the contemporary sense that I started throttling how much I was exploring the chaotic system everything was based around. As a result, at the end of the semester, I greatly increased the primary sound parameter factors (amplitude factor, frequency factor) while minimizing the effect the baseline frequency had on the system, allowing for the output of the system to have a more chaotic characteristic to it rather than being well-behaved.

Media 1: *Demo of the Lorenz Attractor while emphasizing the elements of the chaotic system, changing baseline frequency and tempo during runtime.*



Picture 2: *Closeup of the system's output, the Lorenz Attractor causing many fluctuations in the waveform's scanning inside the oscilloscope*.

Conclusion

Through utilization of the Raspberry Pi Pico, MCP4802, audio jack, and speaker setup, a demo was successfully created to project Lorenz system acoustically. And while there was a great deal of effort put into exploring the possible musical implications of the system, the ending result shows that there is plenty of space to continue taking the project if work were ever to be picked up on it once more. Whether it is used to try and fix the system into a more musical context or dig deeper into the root of this chaotic system exploration, there is a lot more discovery to be made within this branch of projects.

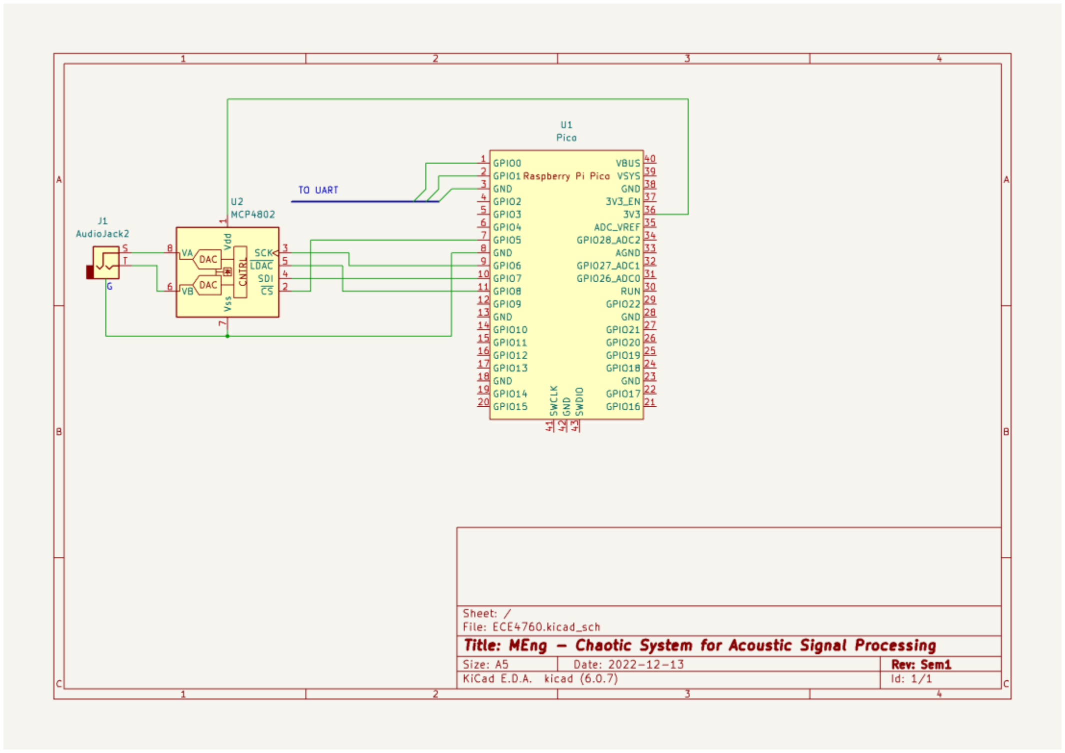
Appendix A: References

[1] Sparrow, Colin (1982). *The Lorenz Equations: Bifurcations, Chaos, and Strange Attractors*. Springer.

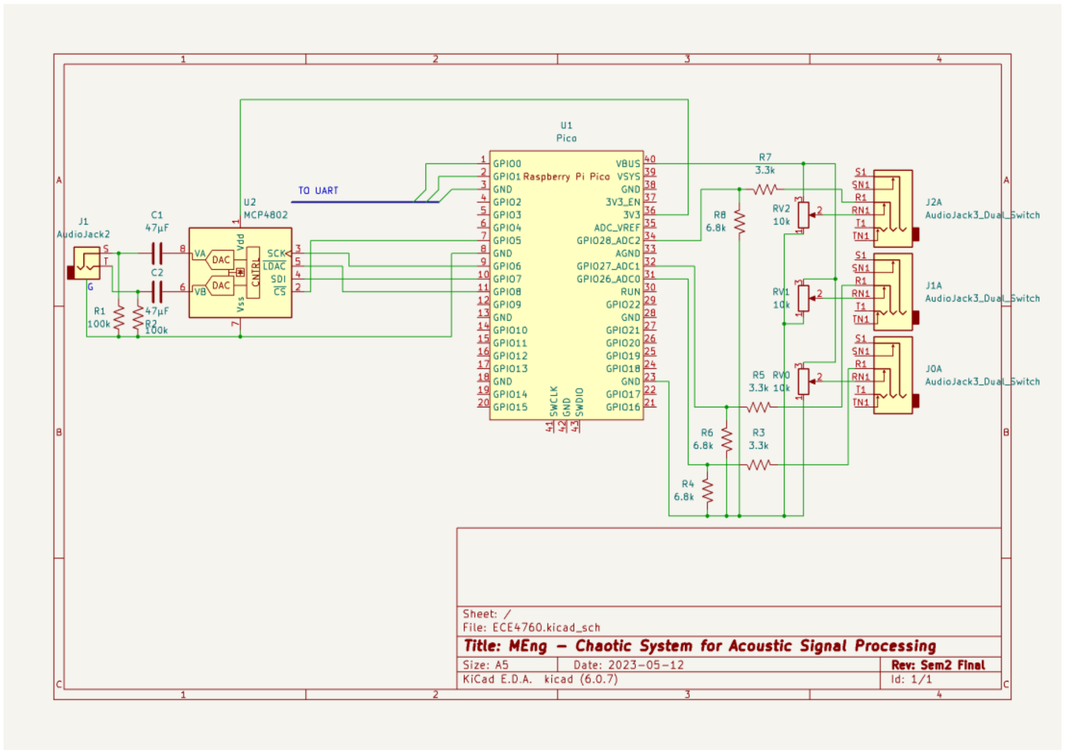
[2] A. Hart, J. Hook, and J. Dawes, “Embedding and approximation theorems for Echo State Networks,” *Neural Networks*, vol. 128, pp. 234–247, May 2020.

[3] L. Dartnell, “Runge-Kutta and the Lorenz Attractor,” *Lewis Dartnell*, 01-Jul-2003. [Online]. Available: http://lewisdartnell.com/en-gb/2003/07/runge-kutta-and-the-lorenz-attractor/. [Accessed: 13-Dec-2022].

Appendix B: Schematics



Schematic 1: Fall Semester



Schematic 2: Spring Semester; Final