Andrew Schweig
Group: Nick Harmon, Neil Foxman, and Jonathan Wolf
MUS66 – Lehrman

# The Boxitron 5 Million

**Original Concept:**

The idea of this project was to create an instrument that could be played by multiple performers and allow an amount of chaos from one performer to be controlled and "shaped" by another person. That way, all music produced is created and manipulated on the fly, with no need to learn a one-to-one note creation system. To create the notes, rubber balls were decided on, as they are influenced by their bounce angle and gravity, leaving very little for the performer to control. The idea of a box lined with switches was streamlined to a pyramid-shaped funnel to allow the balls to strike multiple walls before coming to rest. Then, the chaotic note generation would need to be collected and stored, in order, on the computer. The idea of using a foot panel for this came from the concept of guitar loop pedals, allowing the performer to use his hands to throw the balls. The second performer (the one controlling the sound), needs a control panel with useful buttons and sliders to piece together the loops and make sure they sound as good or as discordant together as desired.

**Functionality:**

The Boxitron is played by one person throwing any number of rubber balls into the enclosed pyramid funnel to generate notes. Each of the eight switches in the funnel corresponds to a scale degree, and a note is only stored when one of the four "stomp" switches is pressed to start recording a loop. When the loop is recorded, the sequence of notes and the time interval between them is stored on the computer.

A second person can access these four loops on the control panel by turning the corresponding switch on, causing the loop to be played back at its original speed. Each loop can be individually modified on the control panel using several parameters. A slider for each loop adjusts the loop's playback speed, from 10 times speed to $1/10^{th}$ times speed (using an exponential scale). Each loop also has individual rotary volume control to allow fading in and out of loops and change the presence of each loop in the sound. Since each loop is sent through a different MIDI channel to Reason, each loop has a separate instrument sound. A second rotary potentiometer controls a channel-specific MIDI parameter. The parameters used were release amount, chorus delay, and filter frequency. Changing these allows dynamic alteration of the sound of each loop as they play.

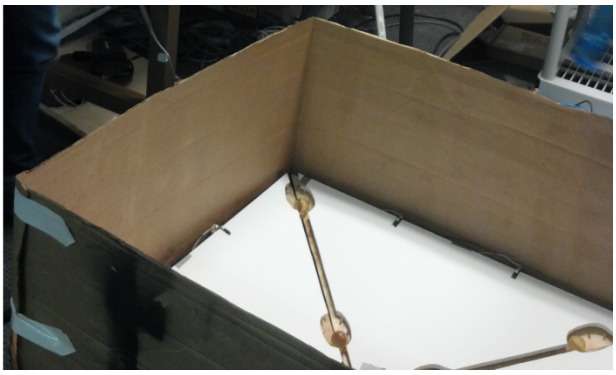
Finally, a single toggle switch modifies whether all loops play back in a major or minor scale. Toggling it lowers the major third and major seventh in the scale for all loops. A set of eight buttons allow the performer to change the current root note of the scale, ranging from middle C to high C, with each of the intermediate major scale degrees included in between. This affects the root note of all four loops simultaneously.

**Hardware:**

The Boxitron has three major components: the box/funnel, the foot panel, and the control panel. The funnel is comprised of four trapezoidal pieces of plywood, held together by eight door hinges. The hinges are screwed onto the rear of the plywood, allowing a slight amount of bend between the segments, but maintaining the pyramid shape. Attached to each segment of plywood is a matching piece of 1/4" foamcore. The foamcore is raised above the plywood by a small amount of standard egg crate foam at the bottom and top. Mounted to the top of each piece of foamcore are two lever switches, spaced roughly 1/4 the way in from each top corner. These are also padded with foam so that

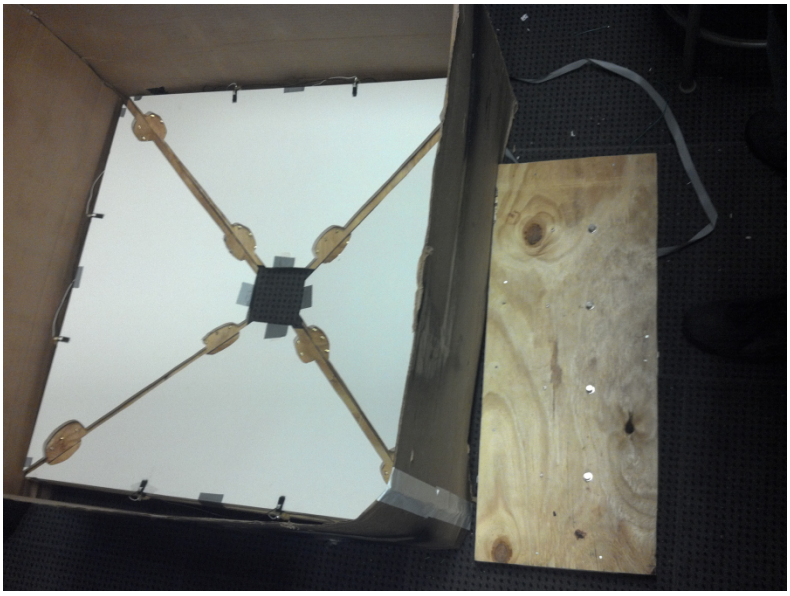
the weight of the foamcore does not permanently depress them. This separation makes a very sensitive trigger, such that a ball dropped or thrown at the panel triggers one or both lever switch easily. The eight switches are wired to a common +5 V source, with a separate output wire attached to each one. All nine wires are collected at one point using and lead out to the foot panel via a ribbon cable. Around the funnel are four walls of a cardboard box, making sure that the balls cannot bounce out of the funnel during a performance. A picture of the funnel and box can be seen below.

*The funnel, showing plywood and foamcore.*



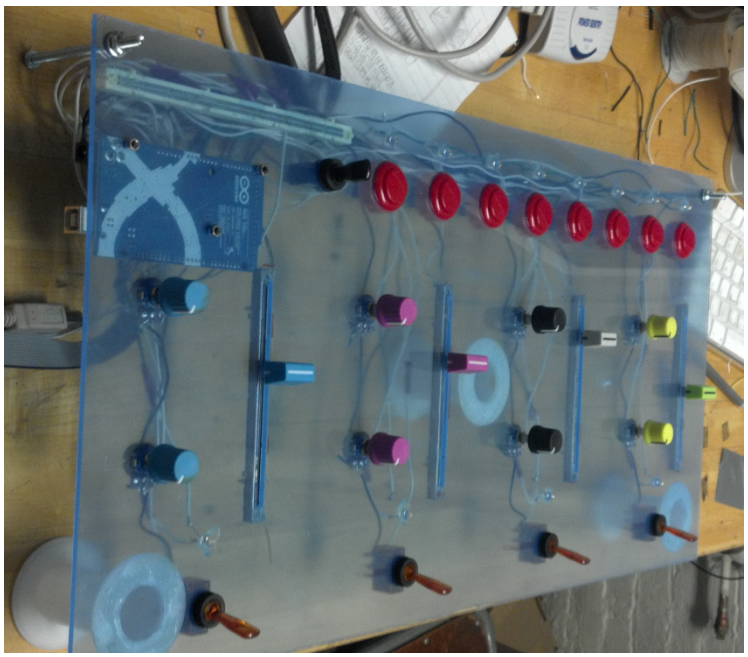
*The cardboard box around the funnel.*

The foot panel is constructed with a sheet of plywood and 2x4's for supports. Four 3PDT stomp switches are mounted in the plywood with a red LED above each. These correspond to the record on/off for each of the four loops. Underneath the plywood, a ribbon cable leading to the control panel is wired to a common ground and +5 V. Additionally, four wires are attached to the stomp switches to send on/off signals to digital inputs on the Arduino. The red LEDs are wired so that an on signal from any stomp switch also allows current through to light up the LED and give feedback to the performer. The ribbon cable from the funnel is wired underneath as well to eight pull-down resistors and then connects to eight wires going out to digital inputs on the Arduino. A picture of the foot panel connected to the box and funnel can be seen below.



*The foot panel, connected to the box and funnel via a ribbon cable.*

The controller panel is the most complex component of the Boxitron, consisting of eight buttons, five switches, four slide potentiometers, and eight rotary potentiometers. The panel itself is made of one sheet of 1/8” acrylic supported by two large bolts and several empty wire spools. The Arduino is mounted on the underside in the upper-left corner with M3 screws through the acrylic. Glued next to the Arduino is a strip of breadboard used to make a row for ground and a row for +5 V. The ribbon cable carrying the inputs for the record on/off switches and the note switches in the foot panel and funnel is attached there. The control panel is divided horizontally into four identical sections, one for each loop. Each section has an SPST toggle switch that feeds into an Arduino digital input to control playback for that loop. Originally, an LED was wired in series with this switch to provide feedback to the performer. However, the Arduino does not output enough current through the digital inputs to light an LED, so the LED above each switch is wired to a digital output on the Arduino and to ground. The code on the Arduino was modified to output a digital high on each of these four outputs whenever the switch’s digital input read as on. Each switch required a pull-down resistor to eliminate noise. In Each section, there are two rotary pots that are connected to analog inputs on the Arduino, utilizing a resistor as a voltage divider. This is due to the fact that the Arduino has a maximum current limit for each port, which was discovered during initial testing. Without a resistor wired in series to ground, when the potentiometer reaches below a certain resistance, too much current flows through, causing the Arduino to disconnect itself from the computer. Similarly, each section has a slide pot that is also wired to an analog input on the Arduino via a voltage divider. Above the sections for the loops is a row of eight arcade-style momentary-on pushbuttons. These are all wired to a common +5 V source, then to digital inputs on the Arduino with a pull-down resistor. Like the playback on/off switches, there is an LED above each button. However, since the buttons are momentary-on, wiring the LEDs in series with the buttons would not work. Instead, each LED is connected to ground and to a digital output on the Arduino. The Arduino code was modified further to store the value of the last button pressed and output a digital high to only the corresponding LED above it until a new button is pressed. Finally, to the left of the buttons is an SPDT toggle switch. This switch is wired to +5 V and to a digital input on the Arduino with a pull-down resistor. All told, the Boxitron uses 12 analog pins and 29 digital pins on the Arduino. A picture of the control panel disconnected from the foot panel can be seen below.



*The control panel, with all components mounted and wired.*

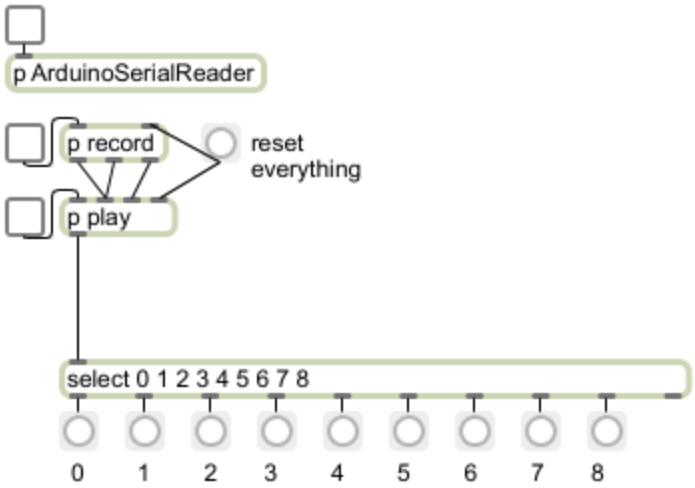
**Max Patch:**

The Max patch consists primarily of four identical series of subpatches, one per loop. In each set of subpatches, the control flow starts with a recording system. When the Arduino sends Max a command to start recording a loop, a timer starts. As each note is triggered, the number of the switch is added to a list, as well as the time since the recording started at which the note was triggered. After recording is topped, the patch stores the list of notes and time intervals in two corresponding lists. These are modified and stored in a coll object. There is an option within the record patch to randomize each note trigger, using semi-random probability based on music theory tendencies. This option was not used in the final performance.

When the Arduino sends Max a command to start playback of a loop, the stored note and timing lists are accessed one element at a time. First, the interval is used to determine when to send out a new note, using counter and timer objects to match up the interval. When the correct time is reached, the counter increments by one and a note is sent out. When the list of notes is exhausted, the counter resets and the loop is played back from the beginning. The slider on the control panel controls the playback speed of the loop by multiplying or dividing the rate at which the timer object counts up to reach the time interval amount, thereby slowing down or speeding up the rate of playback.

Finally, the note numbers sent out by the playback patch are added to the current root note's MIDI value before being sent out to Reason. The pushbuttons on the control panel change the root note's value for all four loops, so a simple addition object takes care of the note value to be sent out. The toggle switch for major/minor scales changes the values added to the root note by changing the interval for the third and seventh notes in the scale.

Additionally, the other eight analog Arduino signals were simply scaled and sent directly to Reason as MIDI CC numbers on the appropriate MIDI channel. A simplified sample of the Max patch can be seen below.




*Simplified series of subpatches in Max showing the control flow of a loop's note generation.*

**Reason Rack:**

The Reason rack consists of four different synthesis modules: three SubTractor modules and one NN19 module. The first loop's notes are sent to a SubTractor synth using a tone called "Faerie's Harp." This sound was modified to produce a bouncy, short tone reminiscent of raindrops or the rubber balls striking something. The parameter modified by the second rotary pot for this sample is the release

length so that the loop can sound more ambient instead of percussive. The second loop is fed into the NN19 module with an acoustic guitar sample. This module was not modified much except to give a longer sustain to give a strummed guitar feel. The rotary pot for this loop also controls the release amount, letting the sound either reverberate or stop quickly, similar to a palm mute. The third loop goes to a SubTractor module with an electric bass guitar tone. The sound of this synth was heavily modified to produce a warmer tone and not have too much reverberation (which caused a lot of clipping). This is then fed through a chorus module. The rotary pot for this loop controls the chorus delay, which modifies how much the sound lingers, as well as how bright the tone is. Finally, the fourth loop is fed into a SubTractor synth using the "Cloud Pad" tone. This is a very ambient, swelling sound that remains playing for a long time after triggered and creates nice layers with new notes played during the decay of the previous one. The rotary pot controls the filter frequency of the synth. These four tones were carefully selected to make sure that each loop could be played back individually and sound good, but also layer with any other loop to create a pleasing texture.

**Problems and Setbacks:**

The biggest problem encountered was creating a funnel that was sensitive enough to register the balls striking, but still be able to utilize lever switches. Many different designs for the funnel were considered, including a two-tiered circular funnel, an open-top box, and a round funnel with a cone coming up from its center. Ultimately, the pyramid shape was selected to give a good amount of bounce and make it easy to mount the foamcore panels. The foamcore itself had to be carefully selected, since it needed to provide enough bounce without being too heavy for the lever switches to hold up. Ultimately, this was overcome by padding with foam to take most of the weight off the switches, but provide enough give to allow a ball striking with an average amount of force to trigger them.

Other than that, the only other problem was between the wiring and the Arduino. While testing the wiring of the control panel, the Arduino would often disconnect itself from the computer saying that it was drawing too much power. This led to the discovery of the Arduino's maximum current output. Once that was known, the components causing the issue (the potentiometers) were used as voltage dividers instead of series resistors. Another problem with the Arduino came with LEDs either only lighting when a button was pressed or not lighting at all. This was solved by modifying the Arduino's onboard code to output full current on different digital outputs. This raised the number of digital pins needed by 12, which also required the code in the Arduino to be modified to read in all digital input pins. A minor inconvenience was that the digital pins had too much crosstalk and every single one needed a pull-down resistor, amounting to at least 32 in total.

**Conclusion:**

Overall, the Boxitron 5 Million accomplished the goals we set forward to create an instrument that implements randomness or chaos and allows another player to modify that and create music from it. The funnel end of the Boxitron is very responsive, but does require re-"tuning" of the switches in between performances to make sure that all the switches are working properly. Otherwise, the instrument is sturdy enough to be transported, and can be taken apart into its three components. The controller end is intuitive to use, and doing almost anything on it produces a pleasing sound. Once a performer has a more in-depth idea of the function of everything on the board, more interesting music can be made. There is even more capability with this instrument than we showed, including the randomization feature that we already built, and an instrument selection routine that could be easily added with a rotary encoder or a couple of buttons. Additionally, the input end of the device can be changed to virtually anything involving random input since the funnel is connected by a ribbon cable that collects eight separate digital signals. Most important, however, is that the Boxitron is an easy to play, highly collaborative instrument that is both fun to perform with and enjoyable to watch.