Tennis For Two

By Angel De La Cruz and John Ryan

Abstract

Tennis for Two was one of first video games ever created. It was built in 1958 by William Higinbotham in Brookhaven National Laboratory using an oscilloscope, vacuum tubes and transistors. Based on military computers designed to calculate the ballistic path of missiles, it used integrating circuitry to calculate the flight path of a ball as it was hit back and forth between two sides of a court by a pair of simple controllers. For our 6.101 project recreated this game using modern analog components. The two major design problems in this project were working within component tolerances while designing circuits that needed to be stable over the course of several seconds and in the precise synchronization needed for an oscilloscope display.

Table of Contents **Page**

1-Introduction

Before the advent of digital technology, computers were massive devices made from large arrays of vacuum tube amplifiers. By arranging discrete components around these amplifiers then stringing them together, they were able to quickly perform a wide variety of complex mathematical operations. These analog computers were used to solve complicated differential equations; they were commonly used by the military to calculate ballistic trajectories, taking into account variables like distance, air resistance, and even the coriolis effect. However, there were also some less serious uses for these computers. In 1958 at Brookhaven National Laboratory, William Higinbotham used an analog computer designed to calculate ballistic trajectories to build one of the world's first video games. It was called Tennis for Two, and the goal was to hit a ball back and forth on an oscilloscope screen that had a net drawn in the middle. For our final project was a recreation of this game using modern analog components.

2-System Overview

This project has two major parts. The first is the analog computer, and the second is the audio and visual output for the game. The analog computer calculates the x and y trajectories of the ball as it moves across the screen, handling the interactions with the boundaries of the stage and the net in the middle. It takes input from a pushbutton which adds some velocity to the ball in the y direction and optionally flip the x velocity of the ball when either of them is pressed. The analog computer outputs a voltage level between ±10V for the x coordinate and 0-14V for the y coordinate of the ball. A 1.6kHz 300mVp-p matched cosine and sine are then added to the x and y voltages respectively to create a circle on the oscilloscope screen. A separate circuit generates the x and y waveforms for the static portions of the stage, i.e. the net and boundaries. This is combined with the waveform of the ball using a high frequency relay and sent to the scope, allowing players to see the game. Finally, there are two sounds that are integrated to add an audio component to the game, a noise when the ball bounces, and a noise when it is hit by the user.

The entire system runs off the ±15V power supply in the 6.101 labkit and the only component used that isn't normally found in the lab is the spdt relay, LCC110[1]. This switch has several useful properties for this project. It requires very little power to turn on, and it switches faster than the human eye can see. Additionally, although pins 6 and 7 are meant to be shorted together for use as an spdt switch, by leaving them unconnected it is able to be used as a pair of spst switches. This saved us from having to buy a separate component when we needed an spst relay, and was immensely helpful in a single case that required a pair of spst switches that behaved in an opposing fashion to the same signal.

[1] Appendix 1

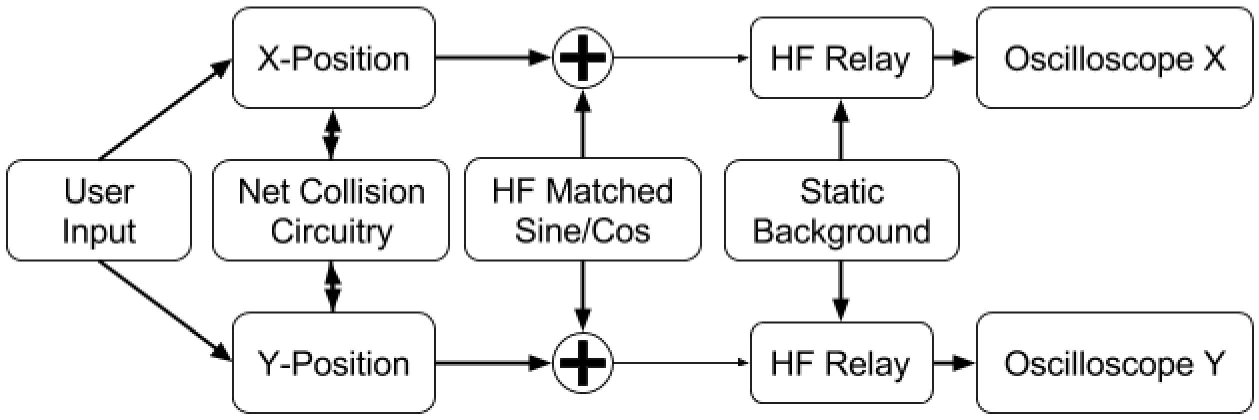

Figure 1: High level block diagram showing the components in our implementation of the video game.

3-Analog Computer (John Ryan)

3.1-Calculating Y Position

To simulate the projectile motion of the ball as it is hit across the screen, a pair of op-amp integrators are chained together. This section closely mimics the original analog computer implementation of the game. Inputting a constant acceleration into the first integrator creates a linearly ramping velocity at the input to the second integrator which causes a quadratic waveform at its output. Feedback between the input and output of the first integrator adds a factor of air resistance, causing the ball to lose energy over time if it is not hit by the user. Figure 2 shows how the integration chain is set up. Additional circuitry is then used to detect when the ball hits the X-axis and invert the velocity.

In the design of this circuit, the main consideration is making the RC value of the integrators as large as possible so that the motion of the ball happens over the course of seconds, allowing users to see the movement of the ball and to react and hit it. The first limiting factor on components is that no guarantee can be made about the polarity of the voltages across the capacitors, so it is necessary to use bipolar capacitors in the integration. Therefore, the largest bipolar capacitors available, 1μF, were used to create a large time constant of integration. Using 1MΩ resistors for R1, R6 and R5 gives an integration time of ~1s.

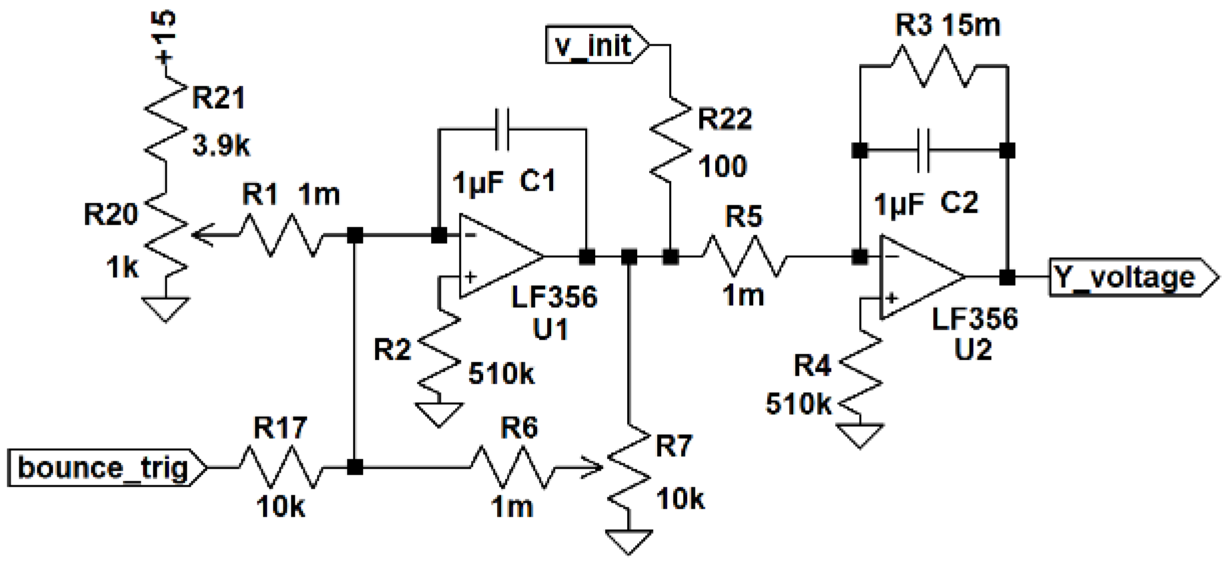

Figure 2: Integration chain for calculating quadratic motion

When considering the signals in this circuit, it is helpful to consider each stage separately. At the input to the first integrator (U1) is a constant acceleration through R1 and a time varying acceleration coming from the feedback through R6. The input of the first integrator is therefore $\ddot{y} = g + k\dot{y}$. Small changes in the magnitude of *g* or *k* can have a large effect on behavior of the circuit, so the magnitude of *g* is adjusted by the potentiometer R20 and the magnitude of *k* is adjusted by R7. If the value of *g* is too small, the y-velocity will not change quickly enough, so the y-position will quickly saturate. If *g* is too large the velocity will ramp so quickly that the ball will not visibly leave the ground. When *g* is at the correct level, the velocity should ramp from the positive rail of the op amp output to the negative, without ever saturating. Once an adequate value is determined for *g*, changing the value of *k* allows for finer adjustment of the waveform. When R7 is turned to ground, the velocity will not significantly affect the acceleration and the ball should keep bouncing almost indefinitely once started. As R7 is dialed up the ball begins to lose energy over time, allowing it to get trapped when it can no longer clear the net in the middle of the court. At the input of the second integrator (U2) is the velocity ramp from the output of U1 through R5. R3 is added to prevent C2 from saturating over time due to DC bias, but is made large enough that it doesn't significantly contribute to feedback. As the velocity ramps down from positive to negative, U2 outputs a parabola, inverted from the final display so it can be summed with a sine wave through an inverting adder. To completely simulate the y motion of a ball, two additional connections are necessary: v_init, which pulls the velocity to some initial value when a button is pressed; and bounce_trig, which inverts the velocity when the ball hits the floor.

To start the ball bouncing, and to hit it while in the air, players use a physical pushbutton switch. When the button is pressed R39 and C8 behave like a passive integrator, creating a quick positive pulse when the connection is completed. D3 prevents a corresponding negative pulse when the button is released by conducting to equalize the charge in C8. The positive pulse briefly causes the MOSFET switch M1 to conduct, activating the switch S1. S1 is an LCC110 switch, with the normally closed pole left floating, so it behaves like an spst switch. While the switch is activated, the velocity is pulled up through R22 to the voltage determined by R50 and R54, popping the ball up in the y direction.

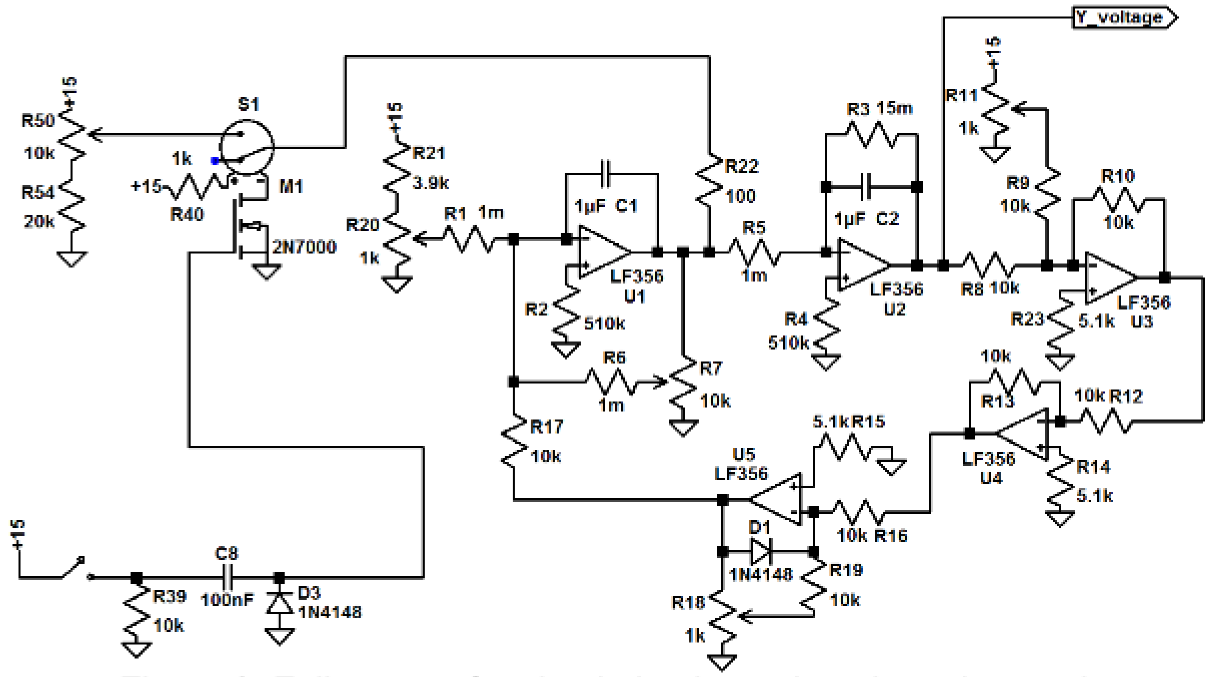

Figure 3: Full system for simulating bouncing along the y-axis

Several things need to happen to create the bouncing effect. First a constant voltage floor is added to the y position through the op amp adder U3 in order to adjust the voltage at which the ball bounces. The result is then inverted by U4 to make the parabola negative again and sent to U5. While voltage at the output of U4 is less than 0, the output of U5 hovers around ground as all of the current coming out of U5 goes through D1. When the voltage coming from U4 crosses to positive, the output of U5 jumps down to -15V. This sharp spike causes current to be conducted through R17, quickly pulling charge off of C1. The value of R17 is chosen to give a time constant on the order of 10ms, which is slow enough that the change can propagate through the op amps in the circuit but fast enough that it appears practically instantaneous to a human observer. The time it takes the ball to come to rest after it first crosses the voltage threshold is equal to the time it takes for the ball to accelerate from rest back across the voltage threshold. Because the the positive voltage ramp in velocity caused by this spike is linear, this therefore means the velocity when the bounce is complete has the same magnitude and opposite sign as when the ball first hit the floor. R18 can be used to adjust the sharpness of the negative spike, and R11 can be used to adjust the voltage level for the floor.

3.2-Calculating X Position

To control the linear motion in the x direction as the ball moves across the screen, a voltage controlled current source/sink was used to push current on and off of a capacitor at a constant rate. An inverting Schmitt trigger is used to set positive and negative boundaries on the x position, and the output of this Schmitt trigger gives a signal of ±14V, the sign of which corresponds to the direction the ball is traveling.

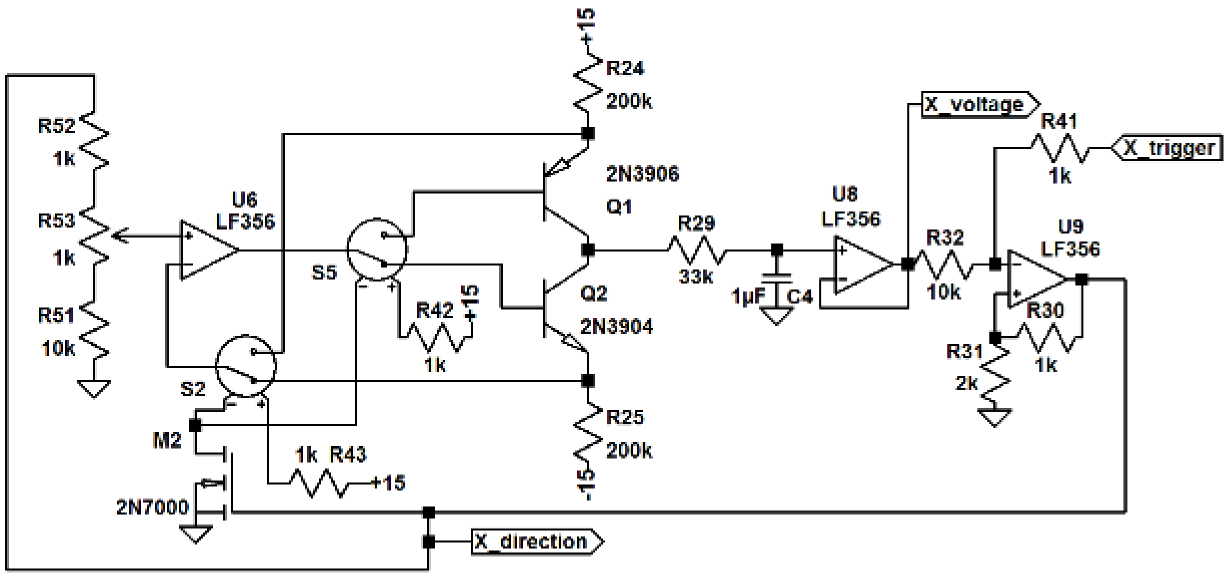

Figure 4: System for calculating movement in X

In order to create a linear voltage ramp with an adjustable slope, the best way is to pump a constant current into a capacitor. Over a constant voltage range, the time it takes to cross between the boundaries depends on C/I, the capacitance divided by the current. Therefore, similarly to the y simulation, a large unpolarized capacitor and very small current are needed to maximize the amount of time it takes for the ball to travel across the screen. With this in mind C4 was chosen to be 1μF, the same as the capacitors in the y circuit. The two resistors used to set the current, R24 and R25, were chosen to give currents that were around 15 μA, giving the ramp a slope of of ~15 V/s. This ensured the time to cross the screen was on the same scale as the integration time of the y stage. By using the Schmitt trigger output to set if the current was sourced or sunk, the ball could be made to switch direction when it hit any boundaries in the stage.

BJTs were chosen for the transistors in the current source, so that they required current from the op amp to be turned on. This was to prevent them from both being turned on at the same time, as a FET can hold charge and remain on when the base is floating.

The speed of the ball is controlled by the voltage at the input to the non-inverting terminal of U6, and is determined by the values of R51, R52 and R53. Over the range of R53 the magnitude of the controlling voltage V_c ranges from about 11.6-12.4V and the sign is determined by the direction the ball is traveling. The switches S2 and S5 are controlled by the output of the Schmitt trigger. When the Schmitt trigger is negative, the output of U6 is connected to the base of Q2 and there is feedback from the emitter of Q2 to the inverting terminal of U6. The op amp then attempts to drive the voltage at the emitter of Q2 to V_c, causing a voltage drop of ~2.6-3.4V across R25, sinking a current of -13-17 μA from the capacitor C4. When the Schmitt trigger goes positive, U6 is connected instead to Q1, and the same process happens, but with the sign of the voltages and currents inverted. Because the behavior can be unpredictable during the switching time, R29 is added to prevent the capacitor from discharging significantly during the switch. R29 should be as large as possible without affecting the behavior of the circuit. To prevent clipping, the voltage drop I_{max}*R29 plus V_{CE} needs to be less than the difference between the minimum value V_C and the triggering threshold. With the chosen boundary threshold of ±10V, and with the rough approximation that V_{CE}~V_{BE}, the voltage drop across R29 needs to be <1V for a current of 17 μA. 33kΩ was chosen for R29, to give a voltage drop of about .6V.

The voltage level from the capacitor is passed through a voltage follower (U8) to add a buffer and prevent C4 from being drained by later stages. This voltage is passed to other parts of the circuit as the x position, and is sent to the Schmitt trigger (U9) to set the boundaries of the stage. Using feedback with R30 and R31, the thresholds for switching are set to ±10V; whenever the x voltage crosses that threshold it flips direction. X_trigger is normally left floating, but is connected and pulls the voltage at the inverting input of U9 across the ±10V threshold when the ball hits the net in the middle of the court.

3.3-Creating a Ball

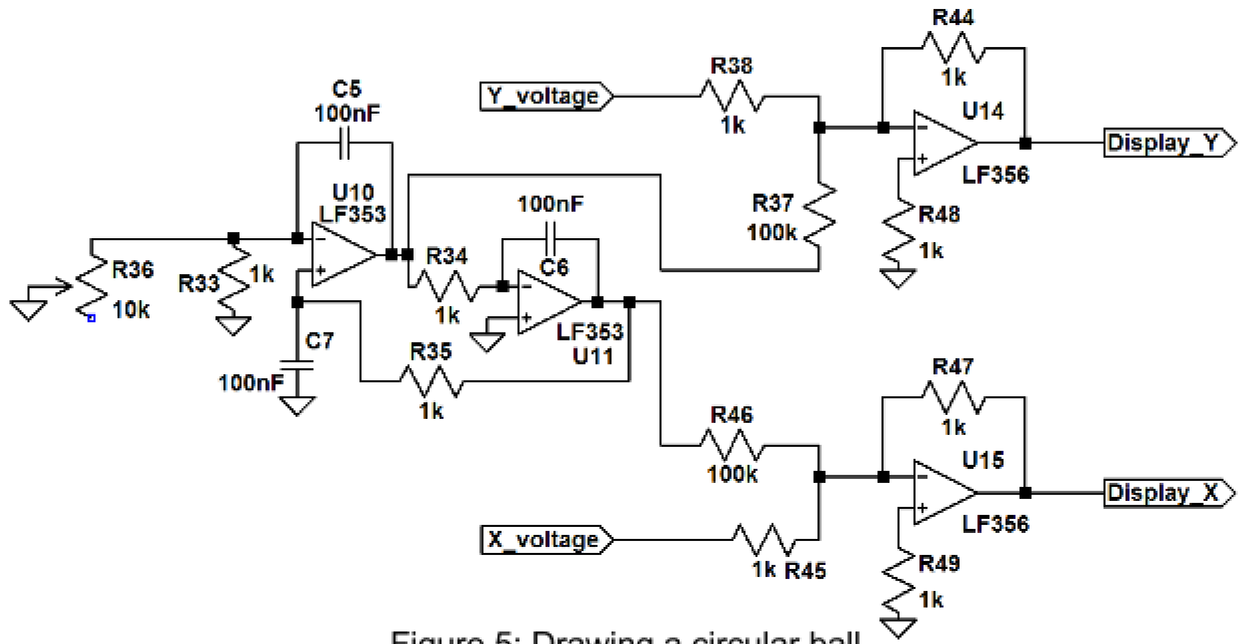

Figure 5: Drawing a circular ball

In order to draw a circle on the xy oscilloscope screen, we need a sine and cosine pair, matched to the same frequency with a 90º offset. This is done using a quadrature oscillator, a circuit designed to create both a sine and cosine. The quadrature oscillator uses a pair of integrators with identical component values placing one in the feedback path of the other. The op amp U11 adds a 90º phase shift to the output of U10, then sends it back to the U10's non-inverting terminal. U10 attempts to drive its inverting terminal to be equal to its non-inverting terminal, and so the system resonates and creates a sine wave. The output of U11 is therefore the integral of that sine wave, a cosine of the same frequency. The frequency of the sine wave is determined by the RC values in the system, where $f=(2\pi RC)^{-1}\approx 1.6$kHz for the chosen component values. The gain in the system can be adjusted through R36 to prevent clipping, another solution is to place a pair of zener diodes in parallel with C5. The two sinusoids are then added to the x and y voltage using a pair of op amp adders to give a voltage offset to the ball. The sine and cosine have an amplitude of ~15V coming out of the oscillator, so they are reduced by a factor of 100 by the adders, giving ~300mVp-p output for the size of the ball.

3.4-Hitting the Net

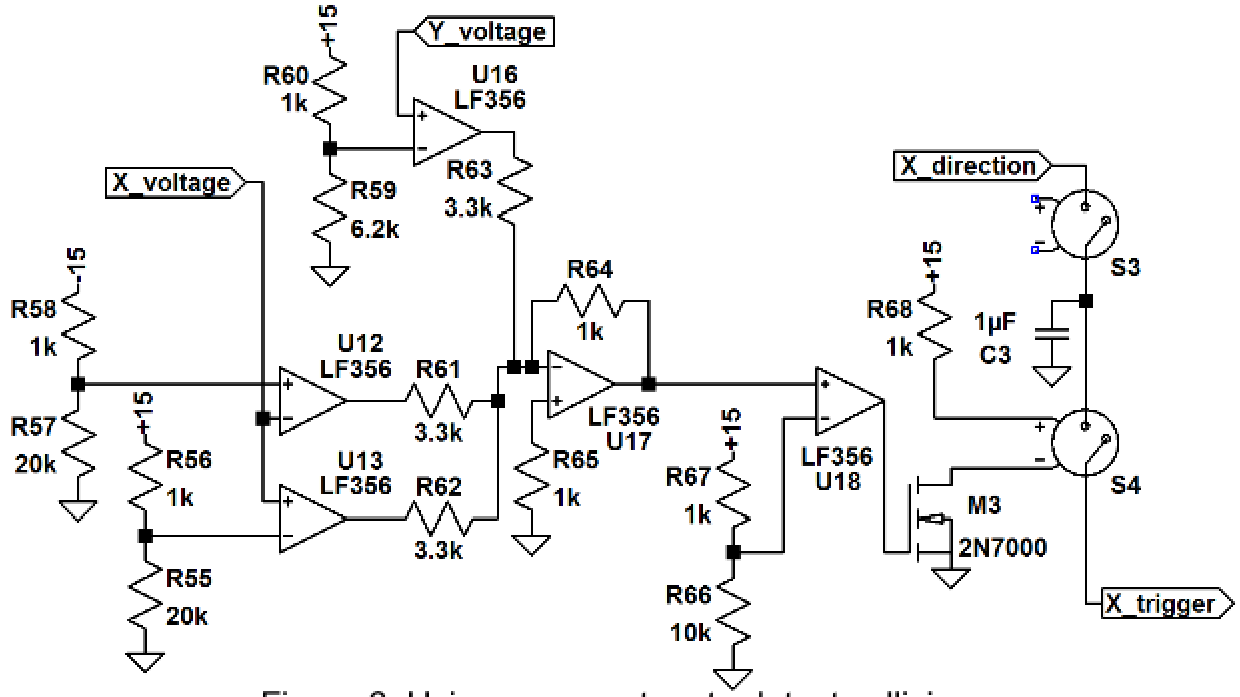

Figure 6: Using comparators to detect collisions

To detect the collision with the net in the middle, a series of comparators are used to flip the x velocity when the y voltage is less than the net height and the x voltage is between the positive and negative boundaries of the net. When the x position is to the right of the left boundary, determined by R57 and R58, the output of U12 goes to -15. Likewise, when the x position is to the left of the right boundary, the output of U13 drops to -15 and when the y position is below the barrier height U12 outputs -15. By summing these three signals using an inverting adder (U17) using equal weight, we get a series of voltage steps of magnitude 5, 10, or 15 volts. The output of U17 is only 15V when all three comparators are at -15V i.e. the ball is in contact with the net. By comparing the output of U17 with a constant voltage of ~13.5 V, a high pulse is generated when the ball hits the net. This high pulse is then used to turn on the MOSFET M3 and send a trigger pulse to flip the x velocity of the ball. S3 and S4 are a single LCC110 switch, but with pins 6 and 7 kept separate to create a pair of spst switches where one is closed when the other is open, and vice versa. S3 is the normally closed switch, and it charges C3 to ±15V depending on the direction the ball is traveling. When the S3 is opened and S4 is closed, C3 discharges into the comparator U9 in the x circuit, creating a voltage pulse that switches the direction of the ball.

3.5-Bringing It All Together

Combining the four major circuit blocks gives the diagram in Appendix 2, the final circuit for simulating the movement of a ball around a tennis court. In addition to the x and y voltages representing the ball, two additional signals are passed to the output stage for the purpose of triggering the audio. The fast negative pulse at the output of U5 is sent to the audio system to trigger the bouncing noise of the ball. The positive pulse from the user's pushbutton is likewise sent to trigger the hit noise when the ball is boosted up in the y direction by the user.

This project highlighted many of the difficulties in building analog circuits designed to work with very small currents over time on the order of seconds. They require the use of parts with large resistances and capacitances, and variations within the component tolerances start to play a large factor, especially when using only a ±15V power supply. It also makes it very important to consider driving of subsequent stages, and how impedance affects the behavior of circuits.

This design project wound up being a very enjoyable experience, and had a great mix of different challenges. It was interesting in the design of the y computer adjusting older schematics to get them to work and figuring out where changes and improvements could be made. It was also very fulfilling to design several new circuits and see them work.

4-Audiovisuals(Angel De La Cruz)

In an attempt to make our implementation of Tennis for Two a complete videogame experience we set out to have a complete display and sound effects that correspond to the gameplay.

4.1-The Tennis Court

Tennis for Two only requires the display of two different components: the tennis court and the ball. The tennis court in our game depicts the side view of an actual tennis court. This results in a pretty simple geometry consisting of a long horizontal line spanning the length of our 'floor' and a short vertical line tangent to the center of the topside of our 'court' to represent the net as seen in Figure 7.

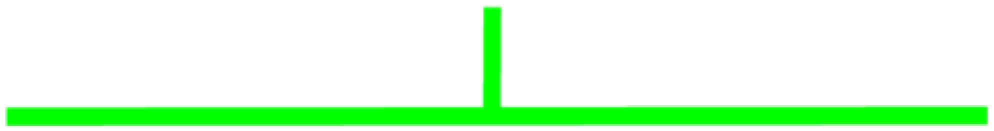
Figure 7: Tennis for Two Court. Idealized representation of the static tennis court for our game

While geometrically simple, implementing the tennis court for display on an oscilloscope turned out to be quite a tedious task. Our approach to implementing the tennis court was very heavily inspired by the work of Daniel Beierl on his Analog Text Display project for 6.101 in the Spring 2015 semester. We drew a lot from his circuit implementation to draw the letter T as our tennis court is basically an upside down 'T' with altered dimensions. We learned from his work that being able to display static shapes within a certain subset of geometries was possible through synchronizing different waveforms. Particularly relevant was the fact that for shapes with even or odd symmetry, using a triangle wave as a reference (which would ultimately be the X-coordinate) and synchronizing it with an appropriate waveform would etch out the desired shape. We started our designing process for the static display by figuring out what kind of waveform would help us accomplish the desired shape. We quickly settled at the waveform combination shown in Figure 8.

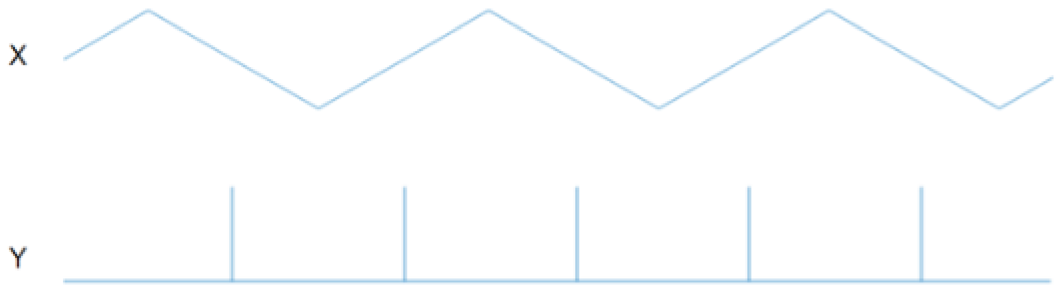

Figure 8: Tennis Court Waveforms. Waveforms required to produce the static tennis court for our game

After determining what kind of waveforms would be required to create our tennis court we began designing the circuitry that would give us these waveforms. Getting the triangular waveforms was achieved through a Schmitt trigger and an OpAmp integrator circuit in feedback with each other as shown in Figure 9.

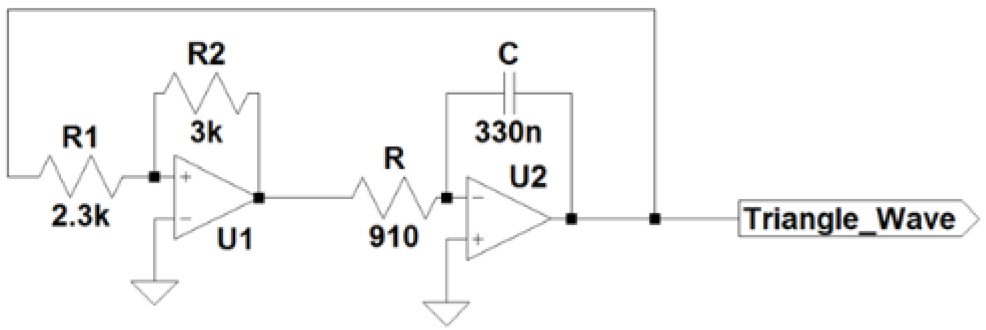

Figure 9: Triangle Wave Generator. Schematic for the circuitry that produces that triangle wave that is used as reference for the display of the tennis court

Determining the component values for the resistors and the capacitor was a matter of determining what characteristics we wanted the triangle wave to have. Our first concern was having the dimensions of the tennis court correct. We wanted the floor to extend from -10V to 10V in the X direction. Accordingly, the triangle wave needed to span from -10V to 10V (have a 20V peak-to-peak amplitude with no DC offset). The maximum and minimum voltages correspond directly to the threshold voltages of the Schmitt trigger which we manipulated by setting the values of R_1 and R_2 . Since we wanted a symmetrical waveform, the corresponding equation is $\pm V_{Threshold} = \pm V_{CC} \frac{R_1}{R_2}$. Ideally, $R_1 = 10k\Omega, R_2 = 15k\Omega$ or any other combination or resistors with an equivalent ratio should have given us the desired amplitude. This, however was not the case, as this combination of resistors led to a considerably lower amplitude than expected. After experimenting a bit, we were able to get the approximate amplitude we desired with $R_1 = 2.3k\Omega, R_2 = 3k\Omega$.

After extensive experimentation we discovered that drawing all of our features at frequencies in the order of 1kHz led to a much more fluid looking display. Thus we chose to ‘draw’ the tennis court 1000 times per second. We calculated that the frequency of the triangle wave is given by: $f = \frac{R_2}{4R_1} * \frac{1}{RC} \Rightarrow 1000Hz = \frac{3k\Omega}{4*2.3k\Omega}\frac{1}{RC}$. We arbitrarily chose C=330nF which gave us a value for $R = \frac{3k\Omega}{4*2.3k\Omega}\frac{1}{1000Hz*330nF}$ $\Rightarrow R \approx 988.14\Omega$. For the sake of minimizing the number of components we ultimately ended up using a 910Ω resistor giving us a frequency of 1086 Hz. This completed our design for the X component of the tennis court.

Once our reference waveform was established, we were then able to shift our focus towards getting the desired waveform for our Y: a periodic delta function. As luck would have it, getting such a waveform from a square waveform is actually a pretty straightforward task and in our generation of the triangle wave for the X component of our tennis court we generated a square wave with matching frequency. Our implementation is shown in Figure 10.

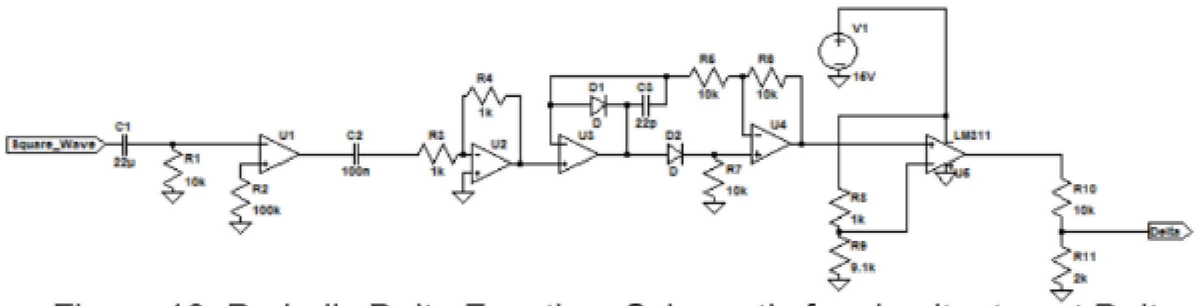

Figure 10: Periodic Delta Function. Schematic for circuitry to get Delta Function from Triangle Wave

We first passed the square waveform through a high pass filter with a cutoff frequency of .7234 Hz in order to eliminate any DC offset that may have found its way into our system. The subsystem formed by U_1 and R_2 serves as a phase shift. This will be elaborated on shortly. The subsystem composed of U_2, C_2, R_3, and R_4 is a differentiator amplifier. This is where the Delta Function ultimately comes from as the large change in slope as the square wave switches between its high and low values come through as spikes in the differentiator while maintaining a value of 0V in between. Expanding on this, when the square wave changes from its low value to its high value we have a very large positive slope while the opposite is true when it changes from its high value to its low value. This, of course, leads to our Delta spikes being both positive and negative. Because we wanted only positive Delta functions, a full wave rectifier was in order. This is shown in Figure 10 in the part of the schematic ranging from U_3 to U_4. The output at U_4 was the periodic all-positive Delta waveform we wanted; however, due to the high frequency nature of our system, when we tried synchronizing this output with the reference (X) waveform, the spike showed a very considerable horizontal component to it. This prompted us to try to improve our Delta functions in order to make the spike as instantaneous as possible. Our solution to this was actually pretty simple: a comparator. We fed the comparator our Delta waveform as its and set the reference voltage based on the highest voltage seen in the Delta spikes. The highest value observed in the Delta spikes fed to the comparator as input was about 13.25V. With our positive voltage source measuring in at about 14.87V, the reference voltage was set with the voltage diver given by R_8 and R_9 to be about 13.236V. The output of the comparator was a very clean periodic Delta waveform which served our needs very well. At that point, it was simply a matter of adjusting the dimensions of our Delta waveform to match the desired dimensions of our net. We opted to set the net height at 2V, which we accomplished by running the Delta waveform through the voltage divider composed by R_{10} and R_{11}. This completed our implementation of the tennis court.

4.2-Integrating the Display

The ball is mainly handled by the analog computer as it calculates its position and uses a matched sine and cosine to show a ball in its corresponding location. This leaves only the matter of displaying the ball concurrently with the tennis court. This was accomplished by using a pair of synchronized LCC110 relays—each one being dedicated to outputting either the X-coordinates or Y-coordinates—to change the outputs to the oscilloscope between the tennis court and the ball at high frequency. The triggering of the relay was done by an OpAmp oscillator as shown in Figure 11.

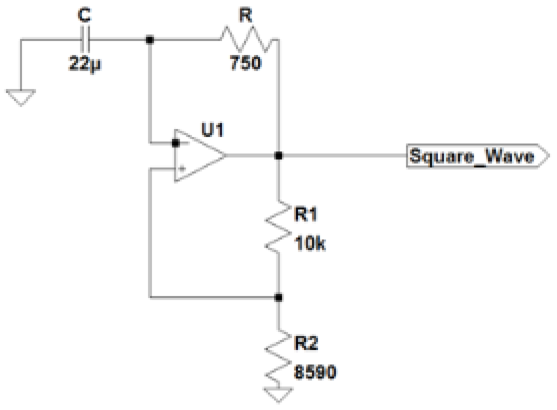

Figure 11: Square Wave Generator: Schematic for the square wave generator that triggered the two LCC110 relays responsible for alternating the outputs to the oscilloscope between the ball and the court

This configuration is basically a Schmitt trigger that takes advantages of a capacitor's charging and discharging characteristics to periodically change its input, resulting in a square wave generator. As we learned in 6.101, a Schmitt trigger changes outputs when a threshold voltage is reached at its input. This threshold is given by: $\pm V_{th} = \pm V_{CC}\frac{R_2}{R_1+R_2}$. Of course, our main concern with this circuit was getting a frequency to drive our relays that allowed a player to see a single, cohesive display. We calculated the frequency of our oscillator to be: $f = \frac{1}{2RC*ln(2\frac{R_2}{R_1}+1)}$. In order to simplify our equation, we opted to eliminate the natural logarithm by setting $ln(2\frac{R_2}{R_1} + 1) = 1 \Rightarrow 2\frac{R_2}{R_1} + 1 = e \Rightarrow \frac{R_2}{R_1} = \frac{e-1}{2} \Rightarrow \frac{R_2}{R_1} \approx \frac{(2.178-1)}{2} \approx .859 \Rightarrow R_2 \approx .859 * R_1$ thus defining the relative values of R_1 and R_2. We arbitrarily chose to make $R_1 = 10k\Omega$ and consequently set $R_2 = 8590k\Omega$. With those values being set, the equation for the frequency of our square wave generator simplified to $f = \frac{1}{2RC}$. Selecting a frequency was done by experimenting through trial and error. Due to the framerate of the human eye being about 30 frames per second, we knew that the minimum frequency we could use to perceive a complete image was 30 Hz; with higher frequencies tentatively looking more fluid. We discovered that at higher frequencies, the oscilloscope showed a lot of artifacts from the transition between displaying the ball, leading to lines of varying intensity between the ball and different parts of the tennis court. We ultimately decided that the best compromise between maximizing fluidity and minimizing artifacts was 30 Hz. Going back to our equation for frequency, this gave us: $30Hz = \frac{1}{2RC}$. We arbitrarily picked the capacitor value to be $22\mu F$ which meant: $R = \frac{1}{2*30Hz*22\mu F} = 757.6\Omega$. Using a 750Ω resistor gave us a frequency of 30.3 Hz which was more than adequate for our purposes.

When outputting a high voltage, the relays both output the respective X and Y coordinates of the static tennis court display and when outputting a low voltage output the respective coordinates of the ball shape.

4.3-Sound Effects

While the original Tennis for Two implementation had no sound effects (at least not intentional ones), we opted to add a few sound effects if only to give players a more aesthetically pleasing experience. The sound effects are a direct tribute to another early video game with a similar theme: Pong. Through some research, we were able to find characterizations of the sounds that can be heard in Pong. Being that Pong was digitally implemented, we felt that it was fair game to try to implement their sound effects through analog circuitry. The two main sound effects that we wanted to integrate to our video game were for the ball bouncing off the floor and the ball changing direction due to a player hit. In Pong, these are analogous to the ball bouncing off one of the walls and a hit (the ball bouncing off of one of the paddles), respectively. Bouncing off a wall triggers a tone given by a square wave with a frequency of 491 Hz for 16ms, while a hit triggers a tone given by a square wave with a frequency of 246 Hz for 16ms.

For our implementation, each of these sounds is triggered by a delta pulse from different parts of the analog computer, thus necessitating that they be two separate systems. They are nearly identical systems, so I will only go into depth for the design of the bounce sound waveform, while providing the schematics for both. The schematic for the sound waveform generator for a hit is shown below in Figure 12.

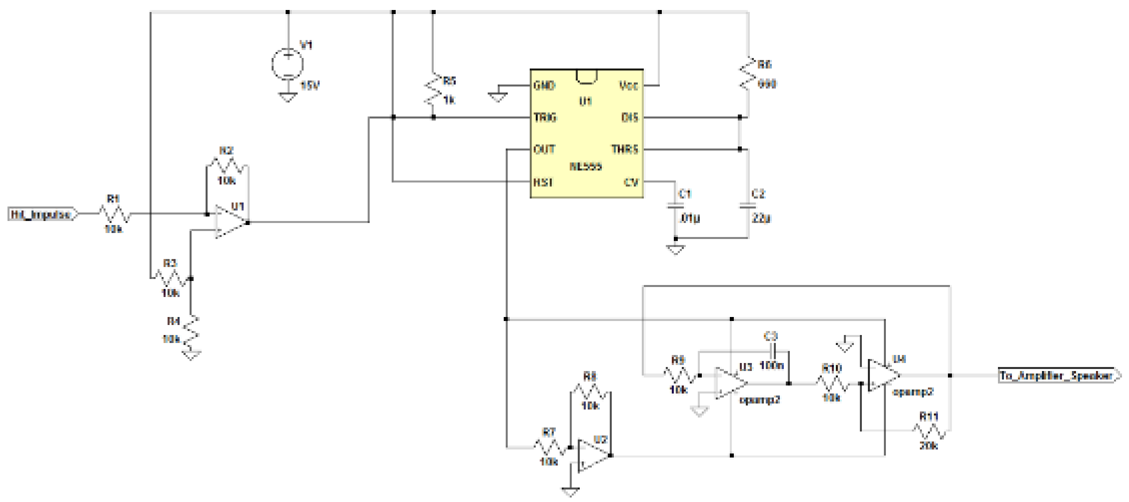

Figure 12: Hit Waveform Generator. Produces a square wave at 491 Hz for 16ms when an impulse is received from the analog computer

To produce the desired waveform for the hit sound, there were two specifications that needed to be met: tone and duration. The first thing we addressed was the duration. We decided that the best way to get a short pulse of sound was by using a 555 timer in monostable configuration. The next thing that needed to be addressed was reconciling the differences between the type of trigger that a 555 timer requires (a low voltage) and the trigger that the analog computer would provide (a pulse of high voltage). This was resolved through the use of an OpAmp in subtractor configuration as shown in Figure 12 by OpAmp U_1. Because $R_1 = R_2 = R_3 = R_4$, the subtractor has unit gain meaning that the output of U_1 is given by $V_{out} = 15V - V_{HitImpulse}$. This means that the trigger of the 555 timer is set at 15V and drops to a low voltage (between 0v and 5V) when the there's a high voltage (10V to 15V) impulse at the input of the whole system. Having figured out the triggering, we set up the the 555 timer in monostable configuration as done in 6.101 for Lab 6. The timing constraints for the 555 timer are determined by R_6 and C_2 and are defined by $t = 1.1 * R_6C_2 \Rightarrow .016s = 1.1 * R_6C_2$ We arbitrarily picked $C_2 = 22\mu F \Rightarrow R_6 = \frac{.016}{1.1*22\mu F} \approx 661.16\Omega$. We used $R_6 = 660\Omega$.

With these timing parameters in effect the output of the 555 timer is V_{CC} for 16ms when triggered. In order to turn this voltage into a square wave, we looked back to the oscillator use to produce the triangle wave and square wave for the tennis court. By setting the appropriate resistors and capacitor in the same way as previously demonstrated, we designed an oscillator to generate the 491 Hz square wave. We tied this to the 555 timer by using the output of the 555 timer as the positive voltage supply for the OpAmps in the oscillator. U_2 is an inverting amplifier with unit gain that inverts the output of the 555 timer to, similarly, provide the negative voltage supply to the OpAmps. This system causes the oscillator to only have power when the 555 timer is triggered, thus creating the precise waveform we require. The waveform generator for bounces is identical with the exception of requiring a different frequency square wave. The schematic for it is shown below in Figure 13.

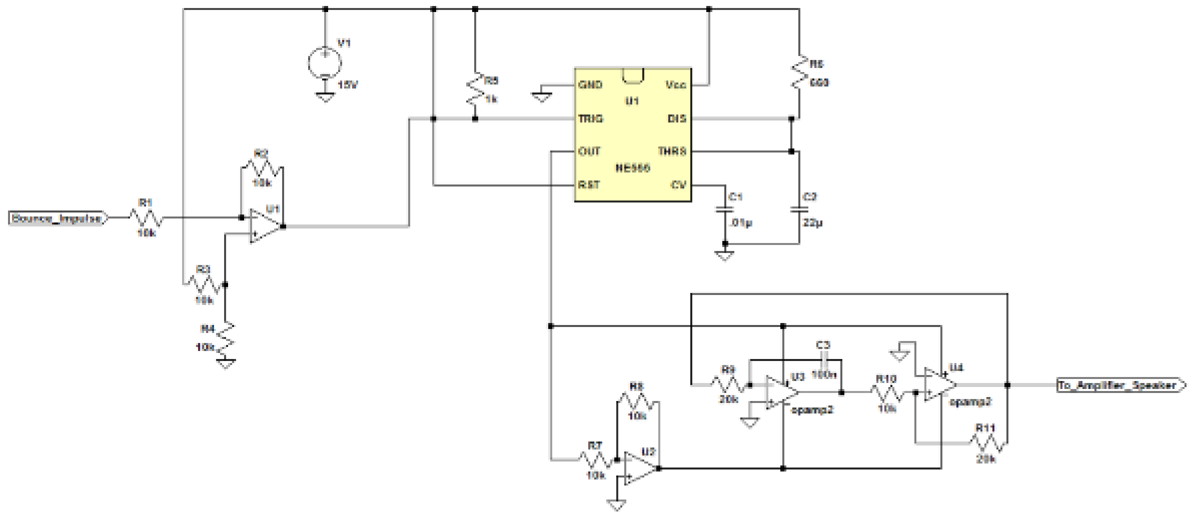

Figure 13: Bounce Waveform Generator.Produces a square wave at 246 Hz for 16ms when an impulse is received from the analog computer

As it turns out, 246 Hz is roughly half of the 491 Hz for the Hit sound. This facilitated the fine tuning of the system. By doubling the value of R_9, we made the frequency of the waveform half of its counterpart giving us our desired waveform for our bounce sound.

Finally, the outputs of both waveform generators were both fed to a two-stage Class A amplifier to finally be turned into our desired audible sounds by a speaker as shown below in Figure 14.

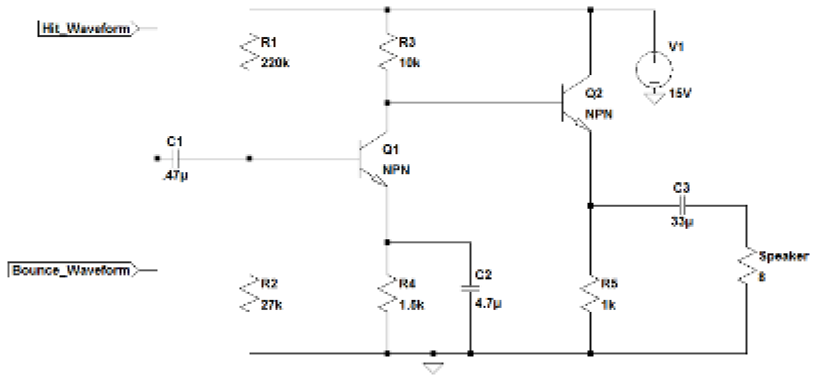

Figure 14: Amplifier/Speaker. Two-Stage Class A Amplifier and speaker setup to emit sound effects

5-Results

Overall we are pretty happy with how the project turned out. We were able to reach our expected goals, and although we were unable to complete all of our stretch goals, it was enjoyable to be able to work towards them. Our final project, while fun to play around with, isn't a serious game, as there is no tangible goal. While analog designs are enjoyable, the strengths of digital are shown in binary conditions like having win/lose states and collision detection that require large complicated arrays of comparators in an analog system. However, the continuous nature of analog has it's own advantages, since it doesn't share the clipping and information loss caused by the discrete nature of digital systems. It was very fun being able to design our own analog system and throughout the course of this project we were able to learn an incredible amount about the design of real world analog circuits.

6-Acknowledgements

We would like to thank Daniel Beierl and Mubarik Mohamoud from the Spring 2015 class whose projects provided valuable insights into displaying pictures on an oscilloscope and analog computing. We would also like to thank Gim Hom and the 6.101 TAs for the incredible amount of work they put into helping us throughout the semester, as well as Amelia Herb and David Custer for their help with the CIM process.

7-Works Cited

D. Beierel, "Alphabet Soup: Displaying Text with an Analog Waveform Generator," MIT, Cambridge, MA, 2015

M. Mohamoud, "The Bouncing Ball Analog Computer," MIT, Cambridge, MA, 2015

H. Holden, 2013, "Atari Pong E Circuit Analysis and Lawn Tennis: Building a Digital video Game with 74 Series TTL IC's." [Online]. Available: http://www.pong-story.com/LAWN_TENNIS.pdf