ChucK Assignment 3

12 April 2012

In this lesson you will explore some of the filters, envelopes, and effects in ChucK, and how to chain them together to produce interesting musical results.

1. You're probably starting to get a little tired of hearing plain oscillators blasting at you for the last couple of weeks. Well, ChucK provides many methods of coloring your sound.

   - **Envelopes** give your sound a dynamic shape so you can control attacks and decays on individual notes.
   - **Filters** amplify and attenuate different frequency ranges to brighten or darken the tone.
   - **Effects** like reverberation, delay, and chorus give you fine control over timbre.

   You need to have your ChucK Manual open for this lesson.

1 ADSR Envelope

2. The ADSR envelope stands for **Attack, Decay, Sustain, Release.** You can find it listed on pages 164-165 in the ChucK Manual.

   - Attack, Decay, and Release are **durations.**
   - Sustain is a **float** value between 0 and 1.

3. To use an envelope, you must place it between the oscillator and the dac:

```
TriOsc myOsc => ADSR myEnv => dac;
```

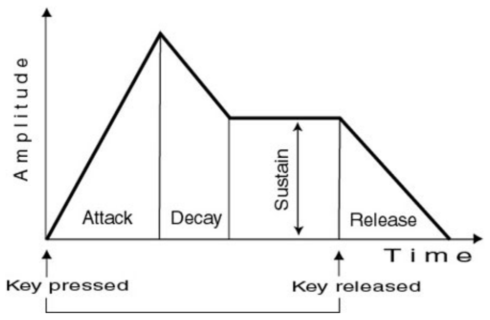

Figure 1: ADSR Envelope

4. Now set up the envelope using the methods on page 165:

```
TriOsc myOsc => ADSR myEnv => dac;
10::ms => myEnv.attackTime; //short attack time
100::ms => myEnv.decayTime;
0.5 => myEnv.sustainLevel;
1000::ms => myEnv.releaseTime; // long release

1 => myEnv.keyOn; // start the attack-decay-sustain
3::second => now;
1 => myEnv.keyOff; // start the release
2::second => now;
```

5. For bell-like or piano-like notes that have no fixed sustain, set the sustainLevel to 0. The release can be arbitrarily short, like 1 millisecond.

```
10::ms => myEnv.attackTime; //short attack time
2000::ms => myEnv.decayTime;
0 => myEnv.sustainLevel;
1::ms => myEnv.releaseTime; // long release

1 => myEnv.keyOn; // start the note
3::second => now;
```

2 Filters

6. Filters change the harmonic spectrum of a sound. They're listed on pages 130-131. They generally have two controls:

   - A main frequency for the filter. For instance, an **LPF** (Low Pass Filter) allows everything below this frequency to be heard, filtering out everything above it.
   - The filter's strength, which is called **Q**.[1] 0 means the filter is off. 1 is moderately strong. Experiment with different Q values to see the effect.

   You can set the frequency and Q separately, or simultaneously with **set**:

```
SawOsc myOsc => ResonZ myFilter => dac;
880 => myFilter.freq;
2 => myFilter.Q;
3::second => now;
myFilter.set(660,10); // 660 Hz, Q=10
3::second => now;
```

[1] No one knows why it's called Q.

3 Effects

7. What sound doesn't benefit from a little reverb? There are a few different reverbs on pages 165-166, each with slightly different sounds.

   - It's most effective to use a reverb on a sound with an envelope.
   - Be sure to allow extra time in your code to allow the reverb to decay

```
SawOsc myOsc => ADSR myEnv => JCRev myReverb => dac;
0.5 => myOsc.gain;
5::ms => myEnv.attackTime;
100::ms => myEnv.decayTime;
0 => myEnv.sustainLevel;
0.2 => myReverb.mix; // Reverb amount
repeat(50)
{
  Std.rand2(400,1200) => myOsc.freq; // random freq between 400 and 1200
  1 => myEnv.keyOn;
  150::ms => now;
}
5000::ms => now; // extra time for reverb
```

8. Experiment with various effects on pages 161-167.

9. Let's try an effect on the sound from your internal microphone instead of an oscillator.

   You should use headphones for this so you don't get feedback.

```
adc => PitShift myPS => dac;
1 => myPS.mix; // all wet sound, no dry
2 => myPS.shift; // up 1 octave (try 0.5 also)
1::minute => now;
```

Try various effects and filters using adc instead of an oscillator.

Assignment 3: Revisit and Revise

Create a short piece that uses an ADSR envelope and either effects, filters, or both. You may revisit the code you used from assignments 1 or 2, or create something new. Save your ChucK code as a .ck or .txt file and upload it to the assignment page on Blackboard by this Tuesday, April 17.