Design Document

Jingyi Lu U5932987

Overview

In part 2 of assignment 2, a collection of sound wave with amplitude envelope is implemented. To be more specific, sine, sawtooth and square sound waves are given, and they are all implemented with the same type of amplitude envelope, that is ADSR. This report will cover the implementation and discussion regarding these things.

Introduction

Amplitude envelope is an envelope for all the values of amplitude of the sound. That is to say, it regulates the shape of the sound wave in amplitude. And in nature, all the sound waves have their own envelopes. In the assignment, a model called ADSR is chosen to simulate the sound of drum.

ADSR amplitude envelope can be divided into four parts, which are attack, decay, sustain, and release separately. And this model generally applies for all musical instruments, like piano, drum, etc.

Implementation

Firstly, reusing the code from assignment 1 part 2 and assignment 2 part 1, sawtooth and sine wave can be implemented. And for square wave, it is implemented similarly. One thing needs mentioning is that, it is convenient to change duration, amplitude and frequency of the wave by using three separate registers to record the values for these characteristics.

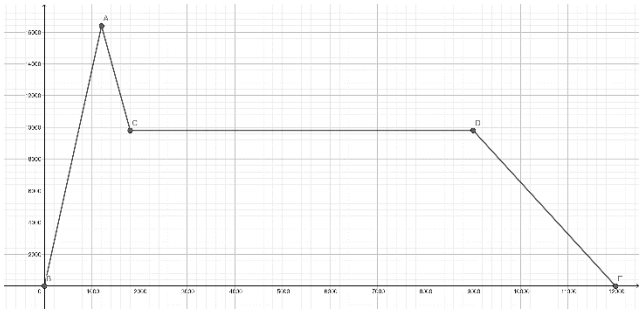
For frequency, it remains unchanged in one piece of note. For duration, it changes every time that the BSP function is called. Due to the fact that the discoboard cost rather little time to operate most operations in ARM. It is hard to record the duration. However, it is known that BSP function has a fixed frequency, that is 48kHz, then the duration can be calculated by calculating the times that BSP function is called. For example, if we want the discoboard to be silent in a duration of 0.5s, then we can minus one for the variable that contains duration originally and if the duration finishes, the function will return and go back. For amplitude, it changes over time and the value of original amplitude is stored in the memory address. In this case, the value of amplitude is the peak of the amplitude envelope. And the peak value is stored for further calculation.

What’s more, to realize the ADSR amplitude envelop, which is talked about in introduction part, we change the shape of every single note of the melody so that it has four phases, which are attack, decay, sustain and release. In order to change the shape, we simply change the amplitude for every circle in the note according to the variable which is used to indicate to termination of duration. To be more specific, the variable that mentioned above for duration

can indicate the times that BSP function is called. Then by minus the original duration by the variable, we can get the number of times that BSP is called. Suppose that x represents that number, and t represents the time that passed for the note. We can get a formula.

$$\frac{x}{48k} = \frac{t}{1s}$$

With this formula, a graph can be generated for an amplitude envelope. In this case, the peak of the envelope is set to be 0x4000, which is 16384, and the duration of the note is 0.25s, that is a maximum of 12k for x. And for attack phase, when $0 \leq x \leq 1200$, the amplitude rapidly rises to the peak. For decay phase, when $1200 < x \leq 1800$, the amplitude rapidly drops to 6o percent of the peak. For sustain phase, when $1800 < x \leq 9000$, the amplitude remains at 60 percent of the peak. For release phase, when $9000 < x \leq 12000$, the amplitude drops smoothly to 0. In this case, all the specific values are chosen subjectively. And it may vary for different choice of values. The graph below shows the amplitude for one note, which has a duration of 0.25s.



Reflection

Firstly, there are many tricky things in the code. In order to improve the modularity of the code, many functions are constructed using branch operation. When it comes to nested functions, push and pop operations are applied to store the value of register lr. What's more, sometimes there are not enough free registers for computation because many registers are used to store values. To solve it, some values that are not frequently used are stored in the memory. For example, the peak value of amplitude is not referred to frequently, hence, we store it in the memory.

Then, although an amplitude envelope is given, it is chosen without referencing any true musical instruments. That is to say, the sound it plays is awful compared to real musical instruments. One potential solution to that looking at the sound wave of one certain type of musical instrument and simulates its amplitude envelope.

Last but not least, in the code, we construct three types of sound waves. However, when we apply the same amplitude envelope for each type of sound wave, the sounds they play are rather different with each other. For sine wave and square wave, the sound becomes awful with amplitude envelope. However, for sawtooth wave, it sounds like drum beats and the sound is significantly different from the original one. Hence, I find that, the sound of notes may change significantly after using amplitude envelope. And for each type of sound wave, the influence of amplitude envelope may differ as well. However, I cannot give an explanation for that because of lack of knowledge in this area.