Felix Virtanen

Elektroniikka ja sähkötekniikka, 2. vuosikurssi

20.02.2021

Ohjelmoinnin peruskurssi Y2 – Projekti (Oma aihe)

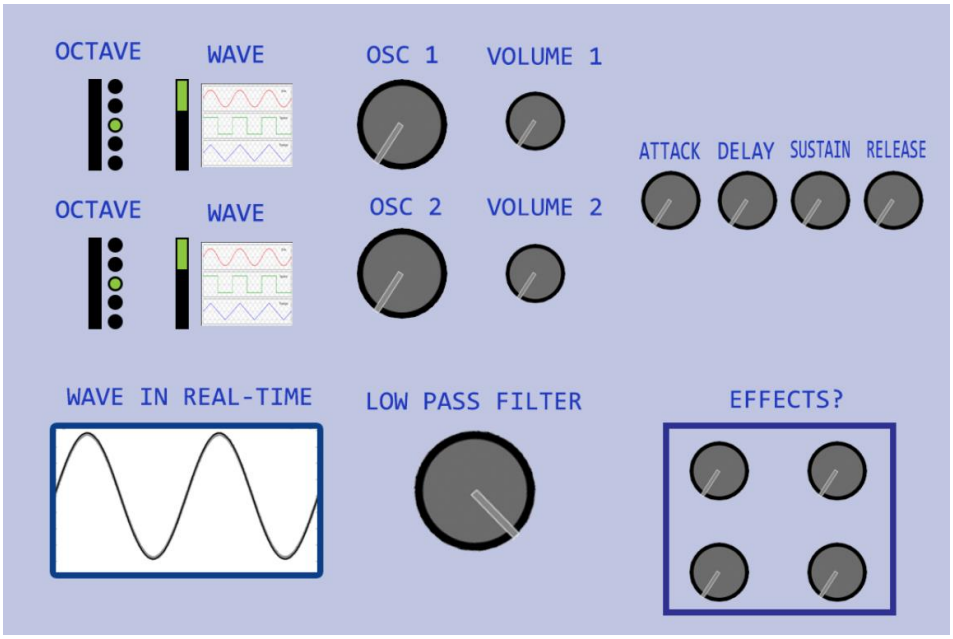
Digitaalinen syntetisaattori (suunnitelma)

2. Yleiskuvaus ja vaikeustaso

Tarkoituksena on tehdä digitaalinen syntetisaattori graafisella liittymällä. Syntetisaattorin olisi tarkoitus sisältää kaksi oskillaattoria (joihin voi valita joko sini, neliö tai kolmio signaalin), ADSR säädöt soinnin äänenvoimakkuuden säätämiseen ajallisesti, oktaavin sekä yleisäänenvoimakkuuden säädöt, visuaalinen ikkuna mikä näyttää signaalin aallon reaaliajassa ja alipäästö suodatin. Oskillaattorit pystyisivät olla samaan aikaan päällä ja kaikkien säätöjen vaikutuksen pystyisi huomaamaan reaaliajassa. Syntetisaattori käyttäisi tietokoneen näppäimistöä koskettimina ja tietty kirjain antaisi tietyn korkuisen äänen. Syntetisaattorin olisi tarkoitus olla monofoninen (polyfoninenkin on mahdollinen, mutta haastavampi). Extrana voisi olla efekti nuppien lisääminen (esim. Kaiku, lfo, särö), myös midilaitteen yhteensopivuus olisi mukava lisätä.

3. Käyttötapauskuvaus ja käyttöliittymän luonnos

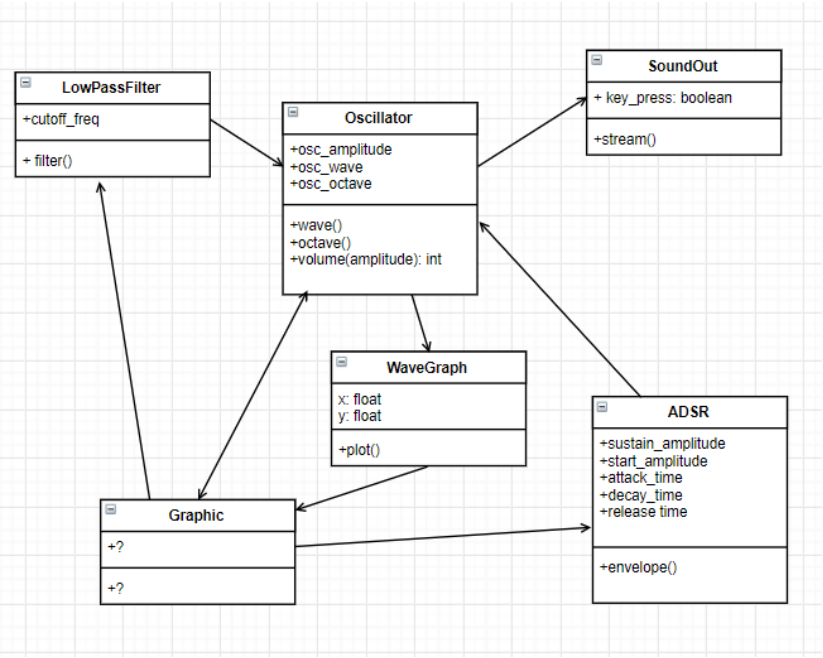
Ohjelma tuottaa tietynkorkeista äänisignaalia painamalla näppäimistössä kirjaimia (a,s,d,f,g,h,j,w,e,t,y,u) (pianon koskettimien asettelun mukaan) ja hiirellä pystyy säätämään erilaisia nuppeja graafisesta käyttöliittymästä, jotka muokkaavat ääntä.



Käyttötapaus:

Käynnistetään syntetisaattori ja “avataan” oskillaattori nostattamalla oskillaattorin signaalin amplitudia. Valitaan oskillaattorin aalto, painetaan näppäimistön näppäintä ja kuullaan signaali kaiuttimista. Voidaan muuttaa eri nuppeja ja kuunnella ääntä painamalla näppäimiä niin kauan kunnes saavutetaan haluttu ääni. ADSR, aallon muoto ja oktaavi ovat “koko ajan päällä”. Kaikki säädöt vaikuttavat lopulta alkuperäiseen signaaliin ja ulostullut ääni on näiden yhdistelmä.

4. Ohjelman rakennesuunnitelma



Oscillator luo signaalit ja sisältää vaihtuvan aallonmuodon, oktaavin sekä amplitudin. ADSR muokkaa signaalin amplitudia ajan mittaan, riippuen painettavasta näppäimistöstä ja Graphic nupeista. WaveGraph piirtää kuvaajan ja Graphic sisältää visuaalisen käyttöjärjestelmän ja säätönupit. LowPassFilter päästää vaan taajuuksia alle tietyn rajan, jonka Graphic nuppi määrittää. SoundOut tuo ulos ääntä streamin avulla.

5. Tietorakenteet

Olis tarkoitus käyttää numpyn taulukkoja (array) säilömään signaalin informaation, koska ne ovat pythonin listoja tehokkaampia.

6. Tiedostot ja tiedostoformaatit

Tietoa ei tarvitse tallentaa.

7. Algoritmit

Alipäästö-suodattimeen tarvitaan todennäköisesti FFT?

8. Testaussuunnitelma

Ohjelmaa on paras testata itse graafisesta käyttöliittymästä vääntelemällä nuppeja ja huomata ero äänessä kaiuttimien kautta.

Olisi hyvä testata, että amplitudeja ei voi nostaa liian korkeaksi. Sen voisi testata siirtämällä nuppi volyyminupin maksimiarvoon ja katsoa onko se koodissa annettu arvo. Pitäisi myös kokeilla onko nupin minimiarvo nolla, että oskillaattorin saa “pois päältä”.

Kuvaajan oikein toimimisen voisi testata visuaalisesti.

9. Kirjastot ja muut työkalut (Ehdotukset)

Pyaudio: Signaalin toistamiseen, (stream)

Scipy.IO: signaalinulostulostamiseen (write), (open)

Matplotlib: Signaalin kuvaajan piirtämiseen.

Numpy: Signaalin informaation säilyttämiseen numpyn taulukoihin (array)

Scipy.signal: (voisi olla hyötyä low pass filterin tekemisessä)

PyQt tai tkinter? : Graafiseen ulkoasuun

Muita harkittavia vaihtoehtoja (Sounddevice, pyo, pyaudiere, Winsound)

10. Aikataulu

- 1. Signaalin muodostaminen ja sen ulostulostaminen kaiuttimista 10h
- 2. Eri taajuiset signaalit aktivoitaviksi tietokoneen näppäimistön avulla 5h
- 3. ADSR toiminnon lisääminen 4h
- 4. Graafinen ulkoasu + siihen eri toimintojen lisääminen + mahdollinen äänen pehmentäminen? 15h
- 5. Kuvaajan muodostaminen graafiseen ulkoasuun 6h
- 6. Alipäästö suodattimen lisääminen 2h
- 7. Hienosäätö ja muiden efektien lisäys? ??

11. Kirjallisuusviitteet ja linkit

https://github.com/Kurene/simple-synthesizer-with-pyqt5-pyaudio/blob/master/main.py

https://github.com/denczo/EardrumBlaster

https://sidparida.com/2017/07/12/python-audio-synthesis-from-scratch-a-tutorial-series-part-2-creating-the-oscillators-and-sound-generation-module/

https://pythonaudiosynthesisbasics.com/

https://stackoverflow.com/questions/9770073/sound-generation-synthesis-with-python

Videot:

https://www.youtube.com/watch?v=dNJ171EsoNQ

https://www.youtube.com/watch?v=DdbYj3o3-3k

https://www.youtube.com/watch?v=tgamhuQnOkM&list=PLrOv9FMX8xJE8NgepZR1etrsU63fDDGxO&index=6

https://www.youtube.com/watch?v=OSCzKOqtgcA&list=PLrOv9FMX8xJE8NgepZR1etrsU63fDDGxO&index=7

Alipäästö suodatin:

https://gist.github.com/junzis/e06eca03747fc194e322

Signals and systems – TIM materiaali

Kirjastot:

https://docs.scipy.org/doc/scipy/reference/signal.html

https://docs.scipy.org/doc/scipy/reference/tutorial/io.html

https://people.csail.mit.edu/hubert/pyaudio/docs/

https://numpy.org/doc/stable/reference/generated/numpy.arange.html

https://matplotlib.org/stable/tutorials/index.html