



รายงาน

เรื่อง

Interactive Musical Instruments / Synthesis with Python

จัดทำโดย

ณัฐกฤตย์ จตุภักดิ์ดิษฐ์	6101012610037
เดชมนต์ แจ่มจิตต์	6101012610061
เชษฐา สุภโตษะ	6101012620091
วศวัตต์ นนท์ธีระวิทยา	6101012620148

เสนอ

อาจารย์เรวัต ศิริโกคาภิรมย์

รายงานฉบับนี้เป็นส่วนหนึ่งของวิชา Introduction to Signal and Systems

สัญญาณเบื้องต้น รหัสวิชา 010123106

ภาคเรียนที่ 1 ปีการศึกษา 2563

มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ

คำนำ

รายงานฉบับนี้เป็นส่วนหนึ่งของวิชา Intro to Signals & Systems ในระดับชั้นปริญญาตรีชั้นปีที่ 3 โดยมีจุดประสงค์เพื่อการศึกษาความรู้ที่ได้จากเรื่องระบบและสัญญาณเบื้องต้นรวมไปถึงการประยุกต์ใช้กับภาษาไพธอนทั้งนี้ในรายงานฉบับนี้มีเนื้อหาซึ่งประกอบด้วยความรู้เกี่ยวกับสัญญาณคลื่นความถี่ของเสียงและมาตรฐานการประสานเครื่องดนตรีแบบดิจิทัลหรือมิติที่นำมาประยุกต์ใช้เป็นซอฟต์แวร์เสียงสังเคราะห์ ตามหัวข้อ Interactive Musical Instruments / Synthesis with Python

ผู้จัดทำหวังว่ารายงานเล่มนี้จะเป็นประโยชน์ในในเรื่องการสร้างแอปพลิเคชันเสียงสังเคราะห์จากภาษาไพธอน หวังว่ารายงานฉบับนี้จะให้ความรู้ และเป็นประโยชน์แก่ผู้อ่านทุก ๆ ท่าน หากมีข้อเสนอแนะประการใด คณะผู้จัดทำขอรับไว้ด้วยความขอบพระคุณยิ่ง

ขอแสดงความนับถือ

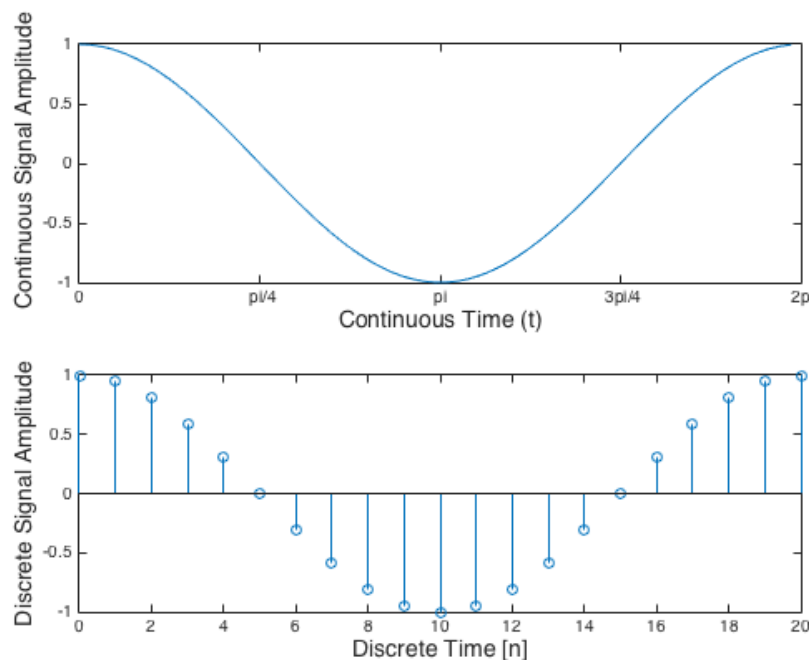
คณะผู้จัดทำ

GitHub Project https://github.com/best2000/synth_key

การสังเคราะห์เสียงด้วย Python + NumPy and Pyo Library

ในการที่จะสร้างเสียงให้เกิดขึ้นมาได้นั้นในทางธรรมชาติเสียงจะเกิดขึ้นได้จากการการที่มีคลื่นเสียงเกิดขึ้น ซึ่งคลื่นเสียงนั้นจะมีลักษณะเป็นคลื่น Sine ซึ่งเป็นคลื่นแบบ continuous signal หรือเป็นสัญญาณที่มีความต่อเนื่องทางเวลา ซึ่งเป็นสัญญาณแบบอนาล็อก

แต่ถ้าในอุปกรณ์ดิจิทัลนั้น การที่จะจัดเก็บคลื่นสัญญาณแบบ continuous นั้น จะไม่สามารถจัดเก็บในรูปแบบของสัญญาณต้นฉบับได้ เนื่องจากการทำงานของอุปกรณ์ดิจิทัล จะสามารถเก็บข้อมูลสัญญาณได้ในแบบ Discrete Signal ซึ่งเป็นการเก็บค่าแอมพลิจูดที่ตำแหน่งเวลา โดยมีช่วงระยะห่างที่เรียกว่า Sampling Rate



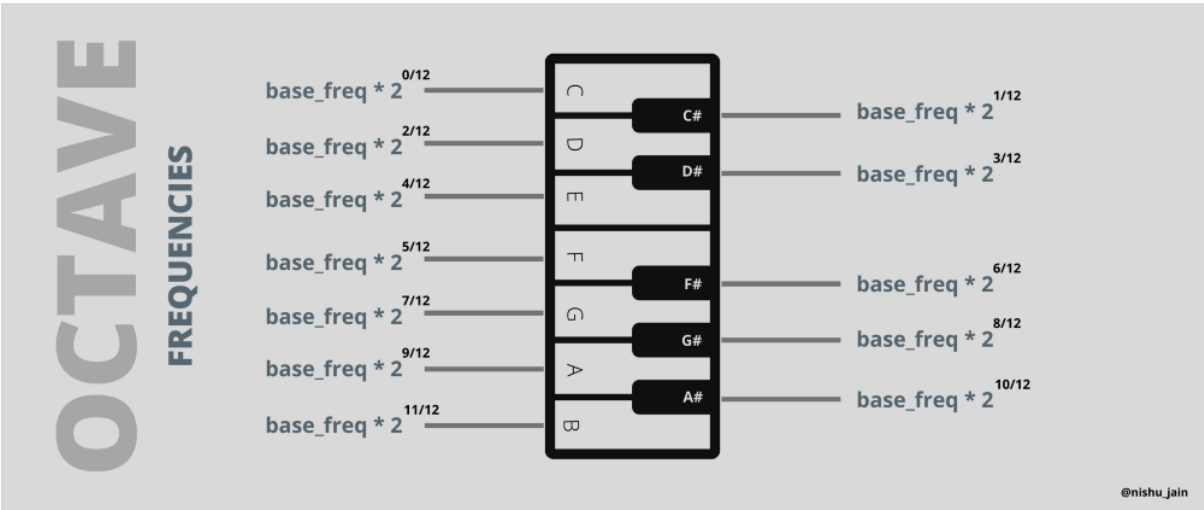
จากภาพ จะเห็นว่าการจัดเก็บสัญญาณในรูปแบบ Discrete Time นั้น จะใช้การเก็บ Amplitude ที่ตำแหน่งเวลาซึ่งการที่อุปกรณ์ดิจิทัลจะสามารถส่งสัญญาณที่เป็นอนาล็อกออกมาได้นั้นจะต้องใช้อุปกรณ์ที่เรียกว่า Digital to Analog Converter (DAC) เพื่อสร้างสัญญาณอนาล็อกขึ้นมาใหม่

ดังนั้น ในการสังเคราะห์สัญญาณเสียงด้วย Python เราจะต้องทำการสร้างชุดข้อมูลแบบ Discrete ขึ้นมา โดยใช้ NumPy ตามวิธีการคำนวณด้านล่าง และนำ Array ของ Amplitude ที่ทำการคำนวณด้วย NumPy แล้ว มาสังเคราะห์เป็นเสียงด้วย Pyo

หลักการคำนวณความถี่ของแต่ละ key

เราสามารถหาความถี่เริ่มต้นได้จากตารางความถี่ของโน้ตแต่ละตัว

Scientific designation	Helmholtz designation	Octave name	Frequency (Hz)	Other names	Audio
C ₋₁	C _{...} or ...C or CCCC	Octocontra	8.176		Play (help·info)
C ₀	C _{..} or ..C or CCC	Subcontra	16.352		Play (help·info)
C ₁	C _. or .C or CC	Contra	32.703		Play (help·info)
C ₂	C	Great	65.406	Low C, cello C, 8' C (see organ pipe length)	Play (help·info)
C ₃	c	Small	130.813	4' C or tenor C (organ), viola C	Play (help·info)
C ₄	c'	One-lined	261.626	Middle C	Play (help·info)
C ₅	c''	Two-lined	523.251	Treble C, high C (written an octave higher for tenor voices) ^[4]	Play (help·info)
C ₆	c'''	Three-lined	1,046.502	High C (soprano)	Play (help·info)
C ₇	c''''	Four-lined	2,093.005	Double high C ^[citation needed]	Play (help·info)
C ₈	c'''''	Five-lined	4,186.009	Eighth octave C, triple high C	Play (help·info)
C ₉	c''''''	Six-lined	8,372.018	Quadruple high C	Play (help·info)
C ₁₀	c'''''''	Seven-lined	16,744.036	Quintuple high C	Play (help·info)



(base_freq)*[(2)^(note_seq/12)]

โดยโน้ตตัวต่อไปจะนำ $\text{base_freq} * 2$ ยกกำลังตามตำแหน่งโน้ตส่วนโน้ตทั้งหมด (12) เรียงจากตัวแรกเริ่ม 0 ถึงตัวสุดท้ายคือ 12

```
Samplerate = 44100

20 def get_wave(freq, duration=0.5):
21     amplitude = 4096
22     t = np.linspace(0, duration, int(samplerate * duration))
23     wave = amplitude * np.sin(2 * np.pi * freq * t)
24
25     return wave
26
27
28 def get_song_data(music_notes):
29     note_freqs = get_piano_notes()
30     song = [get_wave(note_freqs[note]) for note in music_notes.split('-')]
31     song = np.concatenate(song)
32     return song.astype(np.int16)
--
```

จากนั้น เมื่อคำนวณได้ความถี่ของแต่ละโน้ตมาแล้ว จะต้องทำการคำนวณค่า Amplitude โดยจะใช้

Function np.sin ใน NumPy Library

```
--
20 def get_wave(freq, duration=0.5):
21     amplitude = 4096
22     t = np.linspace(0, duration, int(samplerate * duration))
23     wave = amplitude * np.sin(2 * np.pi * freq * t)
24
25     return wave
26
27
28 def get_song_data(music_notes):
29     note_freqs = get_piano_notes()
30     song = [get_wave(note_freqs[note]) for note in music_notes.split('-')]
31     song = np.concatenate(song)
32     return song.astype(np.int16)
--
```

Function นี้จะสร้าง Array ของคลื่น Sine ตามความถี่และระยะเวลาที่กำหนด โดยจะกำหนดจำนวน Sample ตาม Sample Rate * duration ดังนั้น ถ้ากำหนด duration = 0.5 จะหมายถึงการสร้าง array ของสัญญาณแบบ Discrete ที่มีระยะเวลา 0.5 วินาที โดยใน Array จะมี $0.5 * 44100 = 22050$ samples จาก $t = \text{np.linspace}(0, \text{duration}, \text{int}(\text{samplerate} * \text{duration}))$

เราใช้ ฟังก์ชัน np.linspace ในการสร้างตัวแปร t จากการกำหนดค่าระยะจาก 0 ถึง duration โดยมีจำนวนข้อมูลตามจำนวน sample

หลังจากนั้นจะทำการสร้างคลื่นจากสมการ $\text{Wave} = A \sin(2\pi f t)$ โดย A คือ amplitude

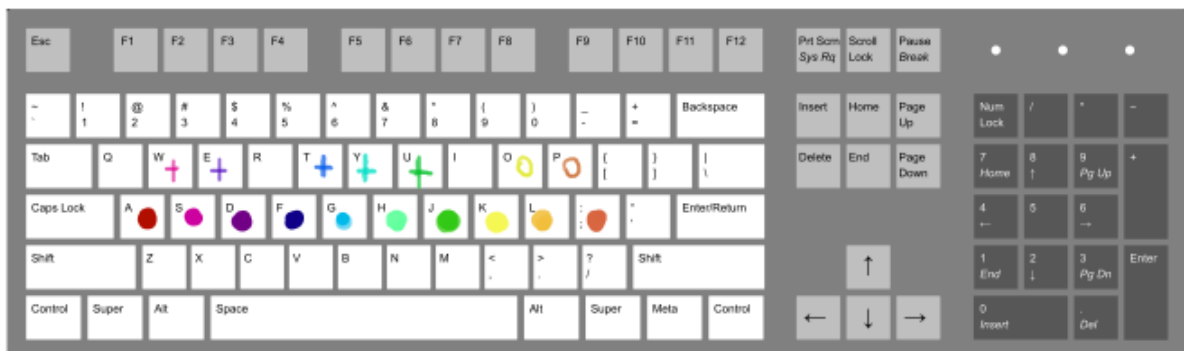
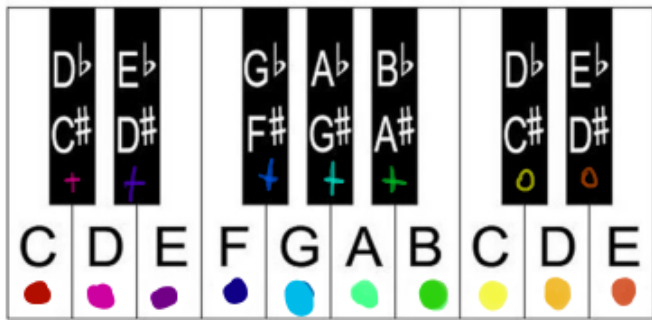
Python Synth Keyboard มี feature ดังนี้

- เล่นเสียง synth จากคอมพิวเตอร์
- ปรับ Octave Key ได้
- เปลี่ยนรูปภาพ คลื่นเสียง หรือ อินพุตเพื่อได้เอาท์พุตตามต้องการได้

How it is made เราได้ใช้ 2 libraries หลักๆคือ

- pyo มีหน้าที่ generate เสียงสังเคราะห์
- pynput มีหน้าที่ รับ input จาก keyboard

แผนผัง key mapping ของโปรแกรม



ปุ่มโหมดต่างๆ

ปุ่ม Z : Sine wave

ปุ่ม X : Supersaw

ปุ่ม C : Supersaw mixing

ปุ่ม V : Triangle

ปุ่ม B : Square

ปุ่มในการกำหนด Octave

ปุ่ม “0” = ห้องเสียงที่มีความถี่เริ่มต้น 16.35 Hz

ปุ่ม “1” = ห้องเสียงที่มีความถี่เริ่มต้น 32.7 Hz

ปุ่ม “2” = ห้องเสียงที่มีความถี่เริ่มต้น 65.41 Hz

ปุ่ม “3” = ห้องเสียงที่มีความถี่เริ่มต้น 130.81 Hz

ปุ่ม “4” = ห้องเสียงที่มีความถี่เริ่มต้น 261.63 Hz

ปุ่ม “5” = ห้องเสียงที่มีความถี่เริ่มต้น 523.25 Hz

ปุ่ม “6” = ห้องเสียงที่มีความถี่เริ่มต้น 1046.50 Hz

ปุ่ม “7” = ห้องเสียงที่มีความถี่เริ่มต้น 2093 Hz

ปุ่ม “8” = ห้องเสียงที่มีความถี่เริ่มต้น 4186.01 Hz

ปุ่ม “9” = ห้องเสียงที่มีความถี่เริ่มต้น 8372.019 Hz

โดยมีหลักการดังนี้

1. Generate คลื่นเสียงขึ้นมา
2. รับ Keyboard Input
3. Input แต่ละตัวจะมีค่าความถี่ที่แตกต่างกัน
4. Output ออกมาเป็นเสียงโน้ตดนตรี

รายการอุปกรณ์ (ซอฟต์แวร์และฮาร์ดแวร์) ที่จะนำมาใช้งาน

Software

ภาษา Python Version 3.8 โดยมี Library ดังนี้

- pyo version 1.0.3
- pynput

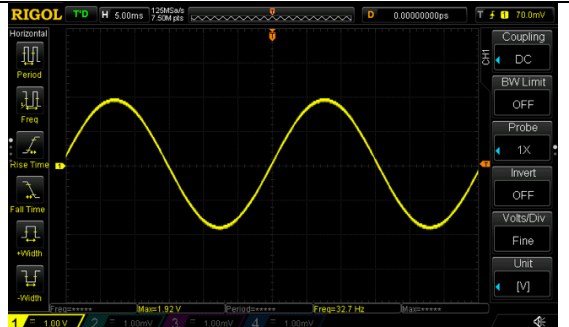

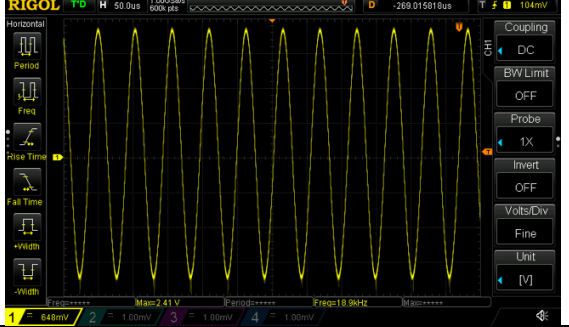

Hardware

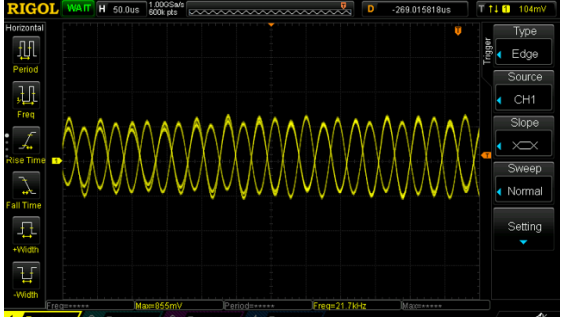
- คอมพิวเตอร์และโน้ตบุค
- computer keyboard

การทดลองวัดสัญญาณเสียงจาก Oscilloscope

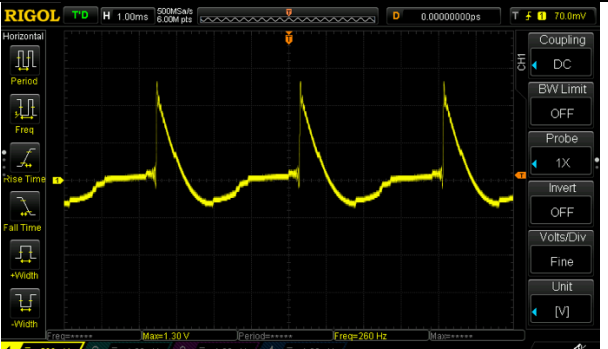

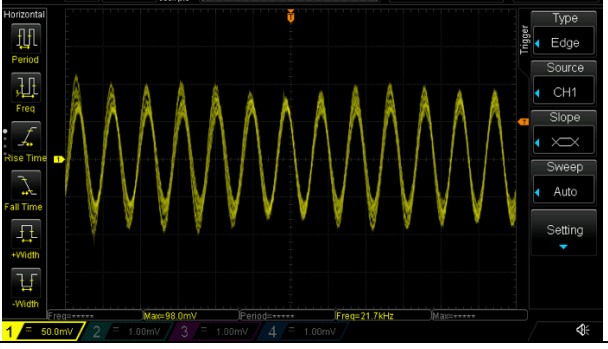
เป็นการวัดสัญญาณเสียง output จากทางช่อง headphone jack output ของคอมพิวเตอร์ โดย การปรับ amplitude จะปรับจากการเพิ่มลดเสียงของคอมพิวเตอร์ และการปรับความถี่จะปรับจาก การกด โน้ตดนตรีในแต่ละโน้ต ของโปรแกรมที่พวกเราได้สร้างขึ้นมาจากภาษา python ซึ่งมีความถี่ที่แตกต่างกัน

สัญญาณ Sine Wave

Frequency	Volume	Graph
32.7 Hz	50 %	
262 Hz	100 %	
18.9 kHz	80%	
21.0 kHz	80%	

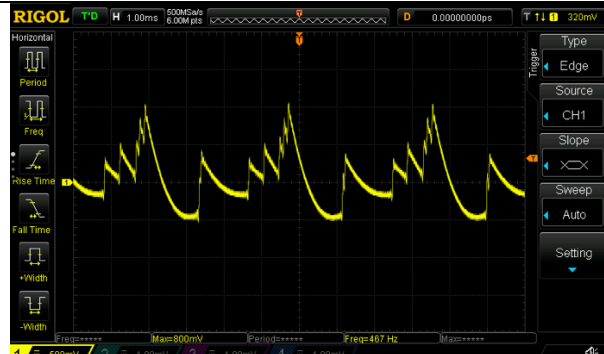
21.7 kHz	80%	
----------	-----	--

สัญญาณ Super Saw Wave

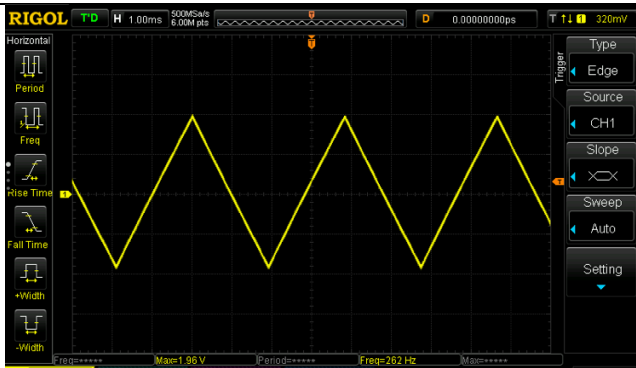
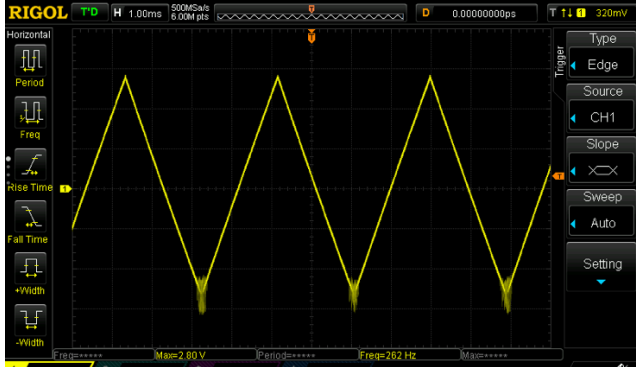
Frequency	Volume	Graph
260 Hz	80 %	
237 Hz	100 %	
21.7 kHz	80%	

สัญญาณ Super Saw + Side Band (Super Saw Mix)

จะเป็นสัญญาณแบบเดียวกับ super saw ด้านบนแต่จะมีสัญญาณอื่นมาผสมด้วย

Frequency	Amplitude	Graph
467 Hz	80 %	

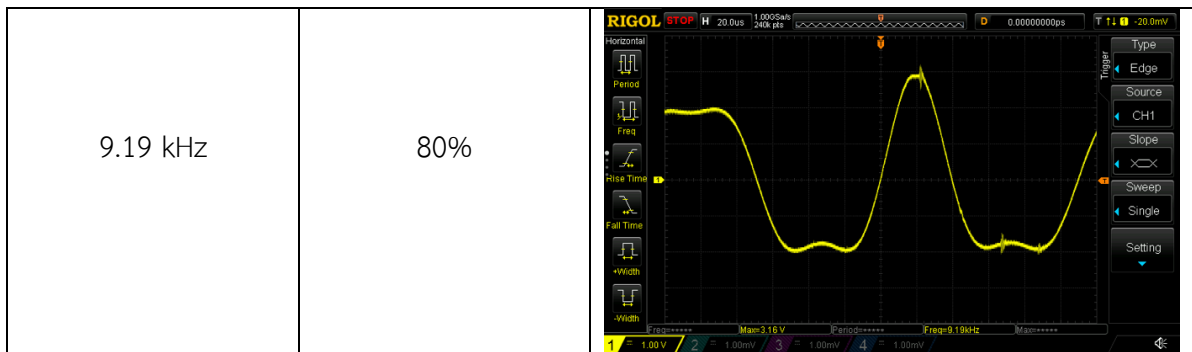
สัญญาณ Triangle Wave

Frequency	Amplitude	Graph
262 Hz	80 %	
262 Hz	100 %	

8.47 kHz	80%	
8.47 kHz	80%	

สัญญาณ Square Wave

Frequency	Amplitude	Graph
262 Hz	80 %	
262 Hz	100 %	



ข้อสรุปจากการสังเกตรูปกราฟ



สัญญาณรบกวนของสัญญาณ Sin เมื่อปรับ Amplitude เป็น 100%

จากการสังเกตรูปกราฟในทุกรูปแบบเมื่อปรับ amplitude เป็น 100% จะทำให้เกิดสัญญาณรบกวนที่กราฟ ซึ่งคาดว่าจะเป็ผลมาจากอุปกรณ์ hardware ปลอ่ยเสียงออกมาได้ไม่สมบูรณ์ ซึ่งเราได้ทำการลองวัดสัญญาณเสียง output จากโปรแกรม Adobe Audition ในคอมพิวเตอร์ด้วยแล้ว แต่ผลที่ออกมา นั้น ไม่มีสัญญาณรบกวน เหมือนที่ทำการวัดด้วย oscilloscope ดังนั้นจึงสรุปได้ว่า เป็นข้อจำกัดของตัว DAC

และสำหรับในเรื่องของ สัญญาณเสียงที่เป็นความถี่สูงเมื่อวัดออกมาจะสังเกตได้ว่า จะมี Amplitude ที่ลดลง และสำหรับสัญญาณ Super saw, Triangle และ Square นั้นจะมีรูปร่างที่ผิดแปลกไป เพราะว่า ในส่วนของเรื่งรูปภาพที่ผิดแปลกไปนั้น คาดว่า เป็นผลเนื่องมาจากอุปกรณ์ DAC ที่อยู่ในคอมพิวเตอร์นั้นไม่สามารถแปลงสัญญาณที่ความถี่สูงมาก ๆ ได้ และการที่ Amplitude ลดลงนั้น เพราะว่าในตัว DAC อาจจะมีตัว High pass Filter ที่จะกรองความถี่สูงกว่าจุด Cutoff ออกไป

ปัญหาที่พบ

ไม่สามารถรับอินพุตพร้อมกันมากกว่าหนึ่งปุ่มพร้อมกันไม่ได้ซึ่งเราทดลองการเขียนโปรแกรมแบบ multiprocessing ซึ่งพบ error จำนวนมากและการใช้งาน thread ให้ประสานกันตรงพอดีนั้นยังไม่สามารถทำได้

อ้างอิง

pyo tutorial:

<http://ajaxsoundstudio.com/pyodoc/>

Chrome music lab:

<https://musiclab.chromeexperiments.com/Oscillators/>

Music Synthesis in Python:

<https://www.youtube.com/watch?v=ROlkhVs15AM&t=1422s>

Note Frequency:

<https://pages.mtu.edu/~suits/notefreqs.html>

How to Play Music Using Mathematics in Python:

<https://towardsdatascience.com/mathematics-of-music-in-python-b7d838c84f72>