In [2]:
```
import numpy as np
import pylab as pl

data_9952=np.fromfile('sec1_9952_sensor1', dtype=np.complex64) #sensor 1
```

In [3]:
```
import numpy as np
import pylab as pl

data_992B=np.fromfile('sec1_992B_sensor2', dtype=np.complex64) #sensor 2
```

SENSOR 1 SPECTROGRAM - data_9952

```
I have decided to use 2Mhz sampling rate as an input and take spectrogram accordingly
```

In [4]:
```
data_9952
```

Out[4]:
```
array([-0.00012207-2.7466269e-04j, -0.00085451-2.5024824e-03j,
       -0.00048829-3.1738800e-03j, ..., -0.00140383+6.1036153e-05j,
       -0.00122072-8.8502420e-04j, -0.00198367+3.0518076e-04j],
      dtype=complex64)
```

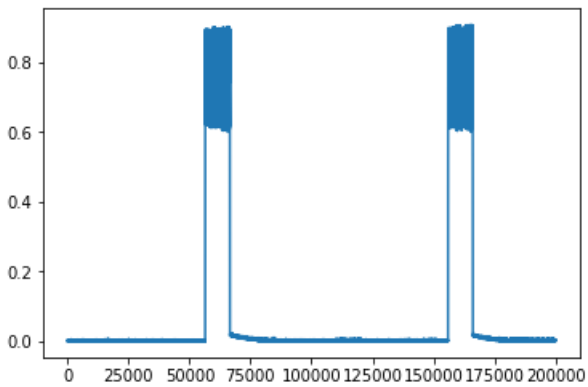
In [68]:
```
len(data_9952)
```

Out[68]:
```
26214400
```

In [79]:
```
audio=data_9952[20000:220000]
```

In [80]:
```
import pylab as pl
pl.plot(np.abs(audio))
```

Out[80]:
```
[<matplotlib.lines.Line2D at 0x1c63cf5e80>]
```



In [66]:
```
## it repeats each 20'000 - 220'000, 220'00 - 420'000 etc....
```

In [71]:
```
len(audio) ##so  26214400/ 200'000= 131, I will have 131 images
```

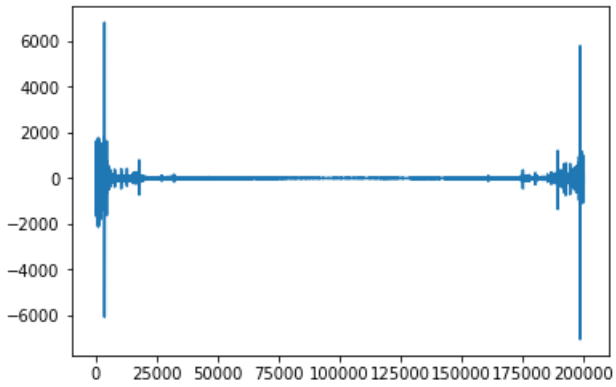
Out[71]:
```
200000
```

In [72]:
```
from scipy.fftpack import fft, rfft
from scipy.fftpack import fftfreq
#complex=fft
#real value=rfft
a=fft(audio)
```

In [73]:
```
pl.plot(a)
```

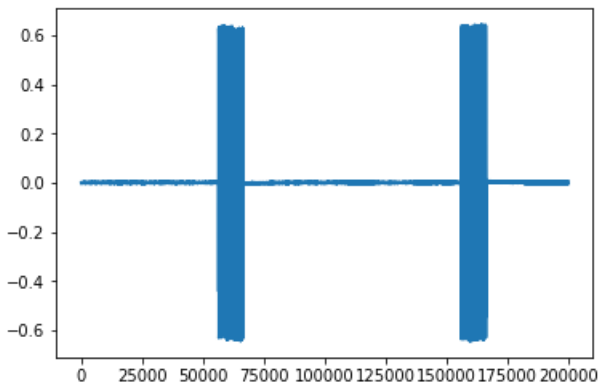
```
/anaconda3/lib/python3.7/site-packages/numpy/core/numeric.py:501: ComplexWarning: Casting co
mplex values to real discards the imaginary part
  return array(a, dtype, copy=False, order=order)
```

Out[73]:
```
[<matplotlib.lines.Line2D at 0x1c535cc3c8>]
```



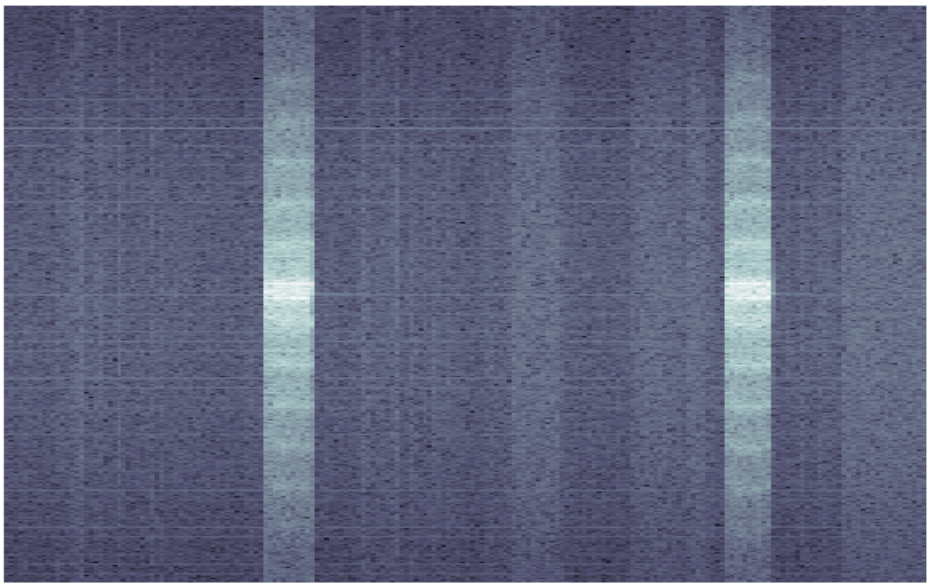
In [74]:
```
pl.plot(audio)
```

Out[74]:
```
[<matplotlib.lines.Line2D at 0x1c64054828>]
```



In [ ]:
```
'''
#APPLYING FILTER
from scipy.signal import lfilter
import matplotlib.pyplot as plt

y=audio
b, a = signal.butter(3, 0.05)
yy = lfilter(b,a,y)

x = np.arange(0, 20000, 0.1)
plt.plot(yy)

'''
```

In [81]:
```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

from __future__ import division
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.ticker import EngFormatter
from scipy import signal #for spectogram

samples=audio
samplingFrequency=2e6
center_freq=1e6

#formatter0 = EngFormatter(unit='Hz')
fig = plt.figure(figsize=(8,5))


ax = fig.add_subplot(111)
#ax.yaxis.set_major_formatter(formatter0)
Pxx, freqs,bins,im,= ax.specgram(samples, NFFT=1024, Fs=samplingFrequency,noverlap=5, mode='ps

d=8
plt.axis('off')
ax.axis('off')
ax.axis('tight')
plt.subplots_adjust(left=0, bottom=0, right=1, top=1, wspace=0, hspace=0)
plt.savefig('plot' + str(d) + '.png')
```



In [87]:
```
## the above is just to see how it looks, now we will process all the images
```

In [ ]:
```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

from __future__ import division
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.ticker import EngFormatter
from scipy import signal #for spectogram
from scipy.signal import lfilter
import matplotlib.pyplot as plt
import numpy as np
import pylab as pl


d=1
i=20000
criteria=len(data_9952) #lenghth of the data
samplingFrequency=2e6
center_freq=1e6

while i<criteria:
    data1=data_9952[i:i+200000]

    samples=data1

    fig = plt.figure(figsize=(5,5))

    ax = fig.add_subplot(111)

    Pxx, freqs,bins,im,= ax.specgram(samples, NFFT=1024, Fs=samplingFrequency,noverlap=5, mode


    plt.axis('off')
    ax.axis('off')
    ax.axis('tight')
    plt.subplots_adjust(left=0, bottom=0, right=1, top=1, wspace=0, hspace=0)
    plt.savefig('plot_200_sample' + str(d) + '.png')

    d=d+1
    i=i+200000
```

In [ ]:

SENSOR 2 SPECTROGRAM - data_992B

In [89]:
```
#I have decided to use 2Mhz sampling rate as an input and take spectrogram accordingly
```

In [90]:
```
data_992B
```

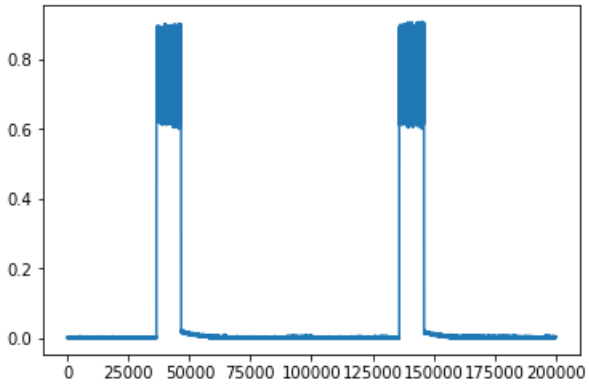
Out[90]:
```
array([-0.00027466+0.0003357j , -0.00241093+0.00375372j,
       -0.00479134+0.00784315j, ..., -0.00057984-0.00021363j,
        0.00039674+0.00039674j,  0.0028687 +0.00106813j], dtype=complex64)
```

In [91]:
```
len(data_992B)
```

Out[91]:
```
26214400
```

In [111]:
```
import pylab as pl
audio2=data_9952[40000:240000]
pl.plot(np.abs(audio2))
```

Out[111]:
```
[<matplotlib.lines.Line2D at 0x1c7a01b080>]
```

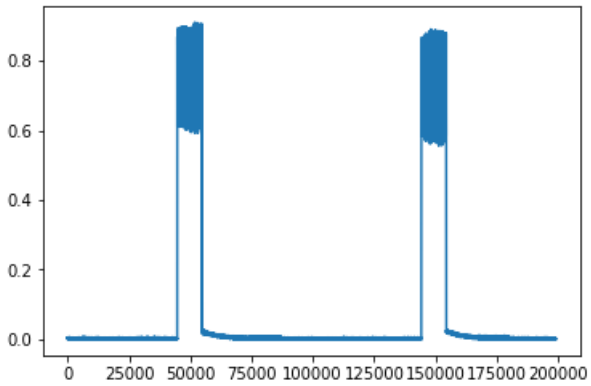


In [112]:
```
## it repeats each 20'000 - 220'000, 220'00 - 420'000 etc....
##so  26214400/ 200'000= 131, I will have 131 images
```

In [182]:
```
#BUT I SHOULD AS IF THE TWO JUMPS ALWAYS STAYS AT THE SENTER
```

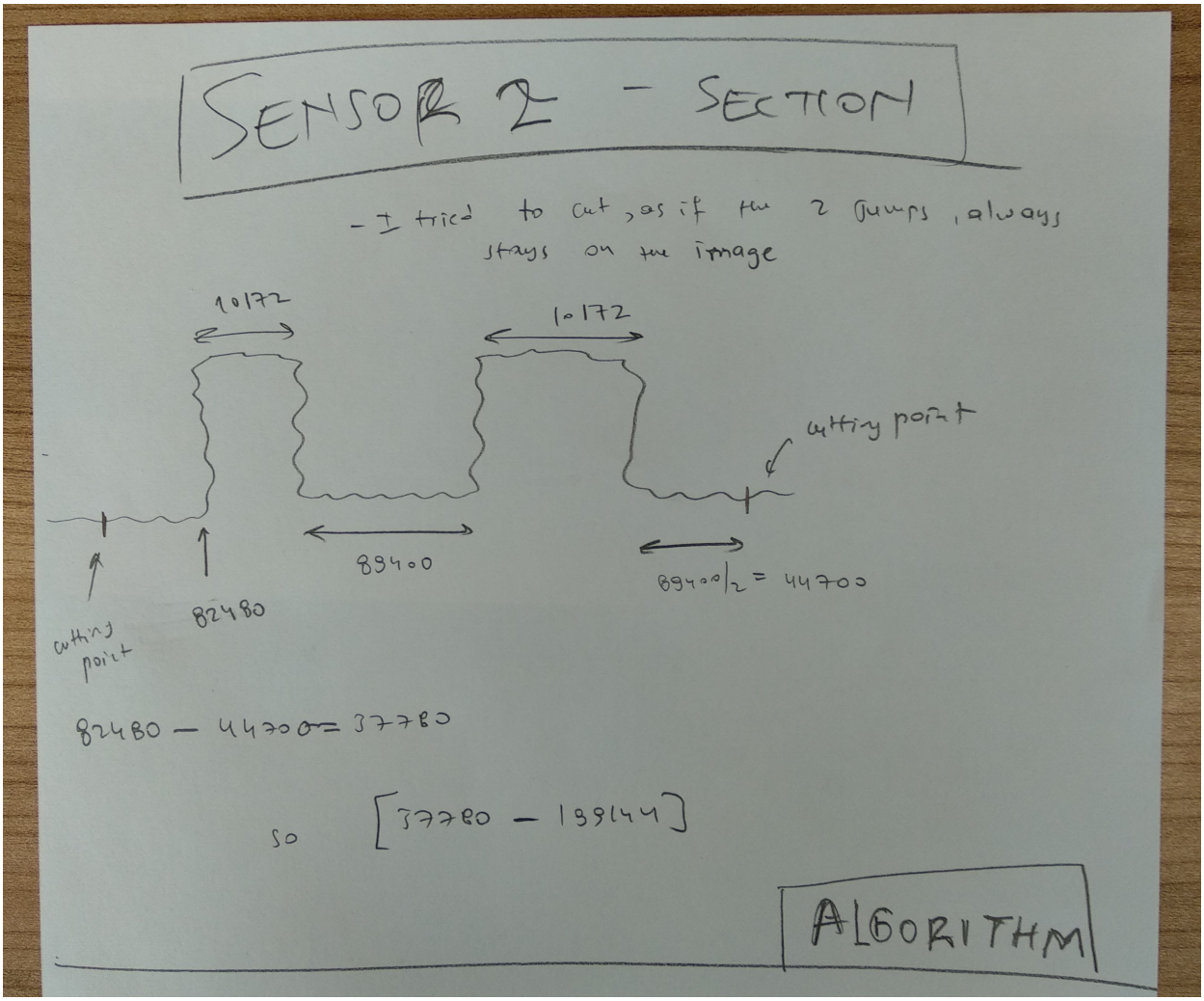
In [179]:
```
pl.plot(np.abs(data_992B[37780:37780+199144]))
```

Out[179]:
```
[<matplotlib.lines.Line2D at 0x1cda618f60>]
```



In [187]:
```
from IPython.display import Image
Image("image.jpg")
```

Out[187]:



In [ ]:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

from __future__ import division
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.ticker import EngFormatter
from scipy import signal #for spectogram
from scipy.signal import lfilter
import matplotlib.pyplot as plt
import numpy as np
import pylab as pl


d=1
i=37780 #starting point
criteria=len(data_992B) #lenghth of the data
samplingFrequency=2e6
center_freq=1e6

while i<criteria:
    data2=data_992B[i:i+199144]

    samples=data2

    fig = plt.figure(figsize=(5,5))

    ax = fig.add_subplot(111)

    Pxx, freqs,bins,im,= ax.specgram(samples, NFFT=1024, Fs=samplingFrequency,noverlap=5, mode


    plt.axis('off')
    ax.axis('off')
    ax.axis('tight')
    plt.subplots_adjust(left=0, bottom=0, right=1, top=1, wspace=0, hspace=0)
    plt.savefig('plot_sectionSensor2_' + str(d) + '.png')

    d=d+1
    i=i+199144
```

In [ ]: