In [ ]:

```
import serial
import matplotlib.pyplot as plt
from drawnow import *
import atexit
from time import sleep
import pandas as pd
import os
from lightgbm import LGBMRegressor
from sklearn.model_selection import train_test_split
import os
print(os.environ['PATH'])
```

Подключаем Ардуино

Открытие порта

In [ ]:

```
serialArduino = serial.Serial("/dev/ttyUSB0", baudrate=115200 ,timeout=0.1)
```

Определяем функцию, которая будет выполняться при закрытии сессии

In [ ]:

```
def exit_function():
    serialArduino.close()
    print("Закрываем Serial сессию")
    print("serialArduino.isOpen() = " + str(serialArduino.isOpen()))
```

Подготовка ардуино к считыванию данных

In [ ]:

```
sleep(3)

coeff_soprot = 52.3
atexit.register(exit_function);
```

Коэффициенты битовых преобразований

In [ ]:

```
base_coeff = 255/5
collector_coeff = 4095/5
```

Ввод данных

In [ ]:

```
fixed = input("Что фиксируем?:")
fixed_voltage = float(input("Введите фиксированное напряжение(0-5В):"))
start = float(input("Введите начальное значение(0-5В):"))
stop = float(input("Введите конечное значеие(0-5В):"))
```

Обработка введенных данных

In [ ]:

```
## База : 256
## Коллектор : 4095
columns = ['U', 'Ib', 'Ic']

if fixed[0].lower() == "b":
    is_base_fixed = True
else:
    is_base_fixed = False
is_collector_fixed = not is_base_fixed

if is_base_fixed:
    fixed_voltage = fixed_voltage*base_coeff
    start = start * collector_coeff
    stop = stop * collector_coeff
    what_fixed = "B"
if is_collector_fixed:
    fixed_voltage = fixed_voltage*collector_coeff
    start = start * base_coeff
    stop = stop * base_coeff
    what_fixed = "K"
```

Генерация команы для ардуино

In [ ]:

```
request = what_fixed + str(int(fixed_voltage)) + ","+ str(int(start))+ "," + str
(int(stop))+"Q"
request
```

Проверка готовности ардуино к работе

In [ ]:

```
print("Готовность к  = " + str(serialArduino.isOpen()))
```

Отправляем команду на ардуино и считываем ответ

In [ ]:

```
serialArduino.write(request.encode("ASCII"))

serialArduino.flush()

values = list()

exit_fl = False

while not exit_fl:
    valueRead = serialArduino.readline()
    values.append(str(valueRead))
    if b">" in valueRead:
        exit_fl = True
```

Обработка полученных данных

In [ ]:

```
# Удаляем последний приянятый элемент и мусор из первого принятого
values[0] = values[0][len("b'" + what_fixed+ str(fixed) + "," + str(int(start))+
 "," + str(int(stop))+"Q"):]
values = values[:-1]

# Считаем кол-во элементов
count_of_values = len(values)

# Разделяем значения
for i in range(count_of_values):
    values[i] = values[i][:-5]
    values[i] = values[i][2:]
    values[i] = values[i].split(';')
```

Создаем dataframe и конвертируем данные в float64

In [ ]:

```
df = pd.DataFrame(values)
df[0][0] = float(start)
df = df.astype('float64')
```

Преобразуем из битов в вольты

In [ ]:

```
if is_base_fixed:
    df[0] = df[0] * 1/collector_coeff/coeff_soprot

if is_collector_fixed:
    df[0] = df[0] * 1/base_coeff/coeff_soprot
```

Готовим данные к визуализации

In [ ]:

```
df.columns = ['j', 'Ibe', 'Ice', 'Ub', 'Uc']
if is_base_fixed:
    df ['Ibe'] = df['Ibe'].mean()
df.head(20)
```

Построение графиков

Ток базы от напряжения базы

$$f(U_b) = I_b$$

¶

In [ ]:

```
plt.plot(df['Ub'], df['Ibe'])
plt.title("Ток базы от напряжения базы")
plt.xlabel("U")
plt.ylabel("Ib")
# plt.xlim([0, 0.001])
```

Ток базы от напряжения коллектора

$$f(U_c) = I_b$$

In [ ]:

```
plt.plot(df['Uc'], df['Ibe'])
plt.title("Ток базы от напряжения коллектора")
plt.xlabel("Ub")
plt.ylabel("Ic")
```

Ток коллектора от напряжения коллектора

$$f(U_c) = I_c$$

In [ ]:

```
plt.plot(df['Uc'], df['Ice'])
plt.title("Ток коллектора от напряжения коллектора")
plt.xlabel('Uc')
plt.ylabel('Ice')
```

Ток Коллектора от напряжения Базы

$$f(U_b) = I_c$$

In [ ]:

```
plt.plot(df['Ub'], df['Ice'])
plt.title("Ток коллектора от напряжения базы")
plt.xlabel('Ub')
plt.ylabel('Ice')
```

In [ ]:

```

```