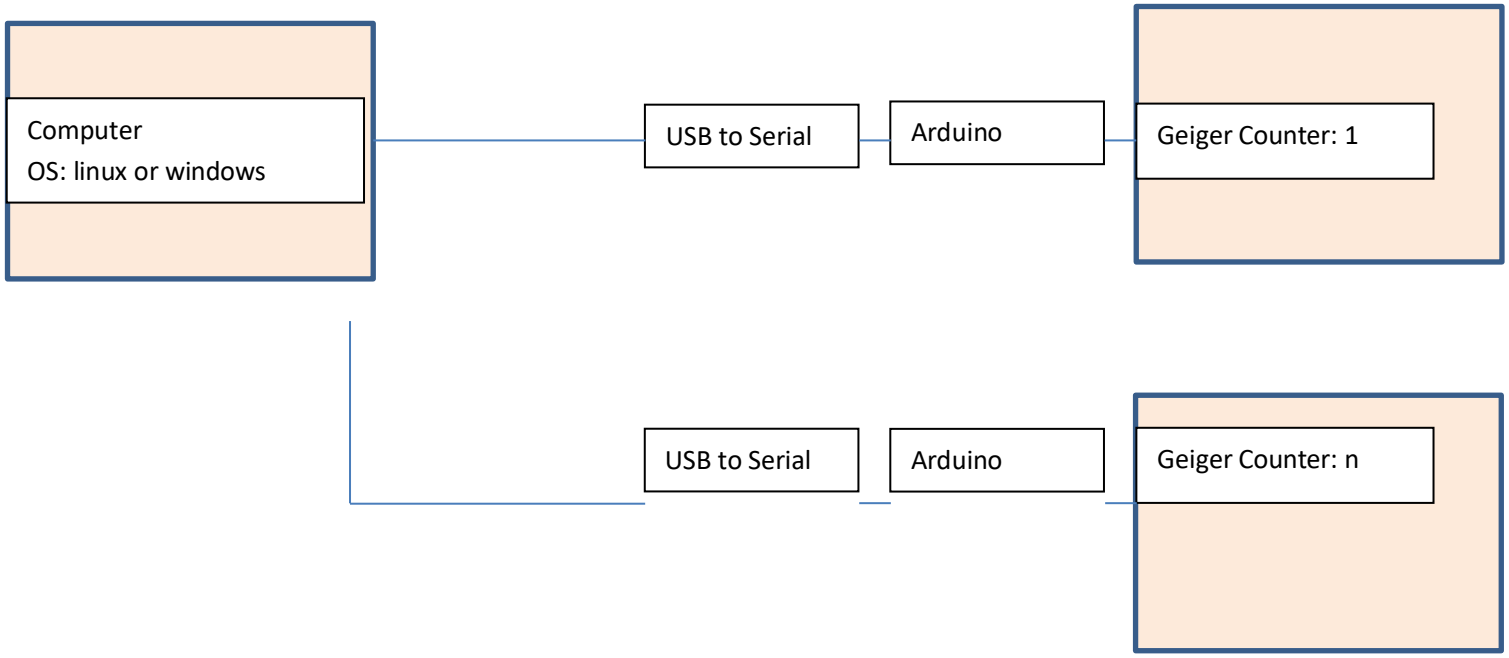
Geiger Counter Local Server—Serial Port Communication Solution



The Geiger counter calculates the radiation intensity and sends the data through the serial port, and the upper computer (PC or Raspberry Pi) runs the python program to read the serial port data and draw it in the graph. Communication through the computer's USB port. This solution can be used when the number of nuclear radiation sensors (Geiger counters) is small, such as 1 to 5, and the data processing requirements could be also relatively simple. When the number of sensors is relatively large, you can consider using the form of 485 bus (using Modbus-RTU protocol). This Modbus-RTU communication will be described in another article.

Material list:

1) Geiger counter (with Arduino nano)

2) Computer (or Raspberry Pi)

Figure 1

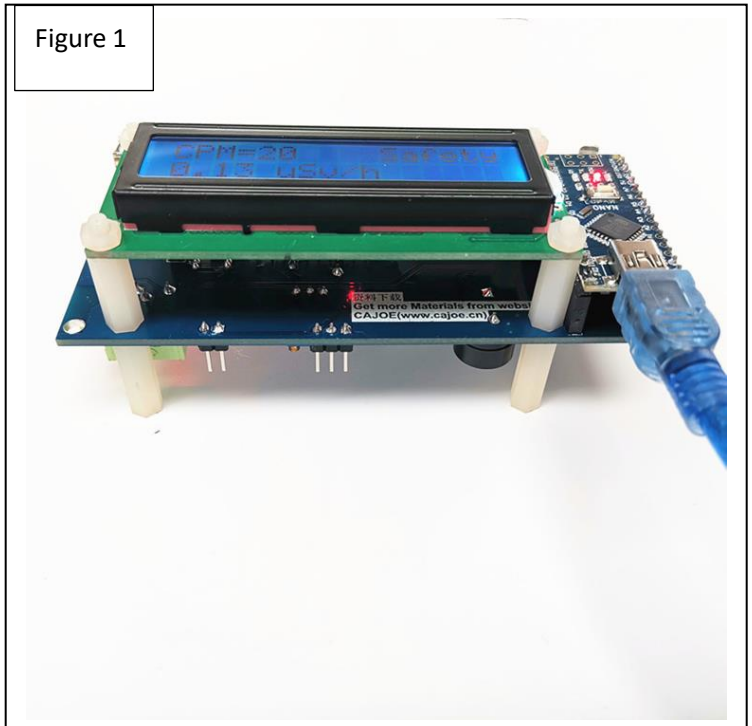


Figure 2

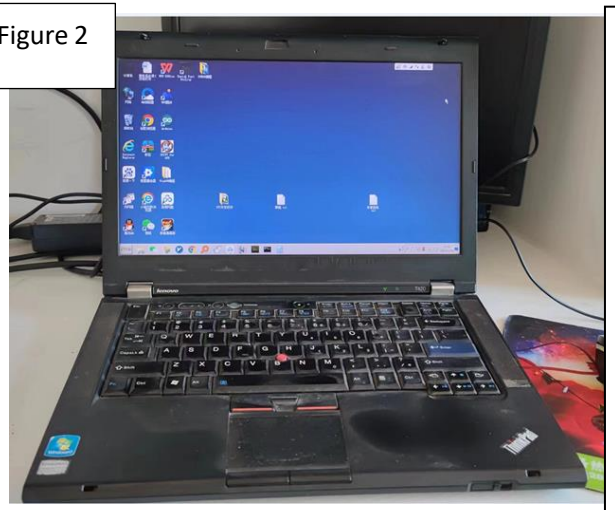


Figure 3

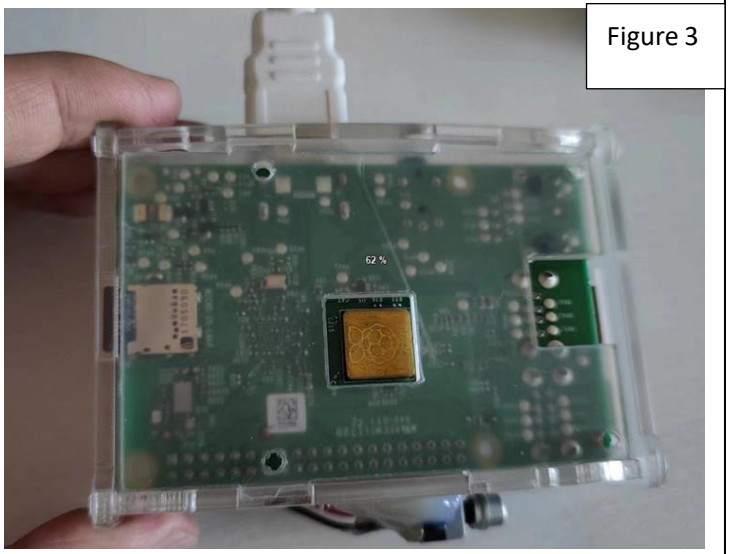


Figure 4

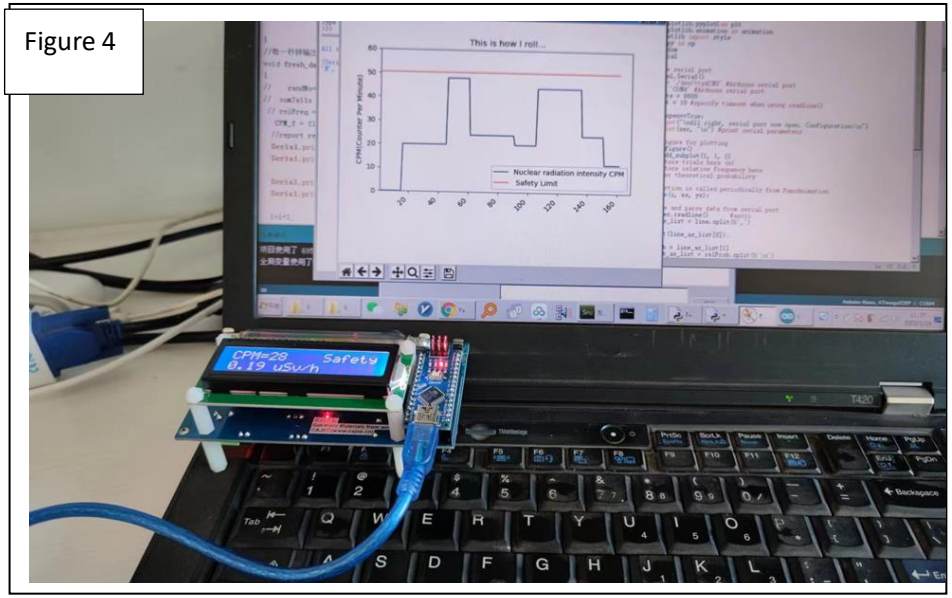


Figure 5

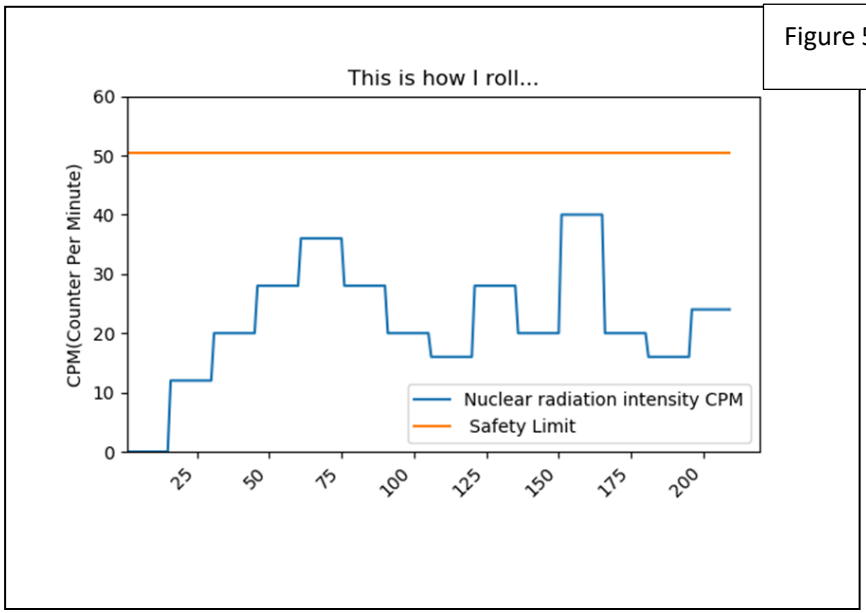
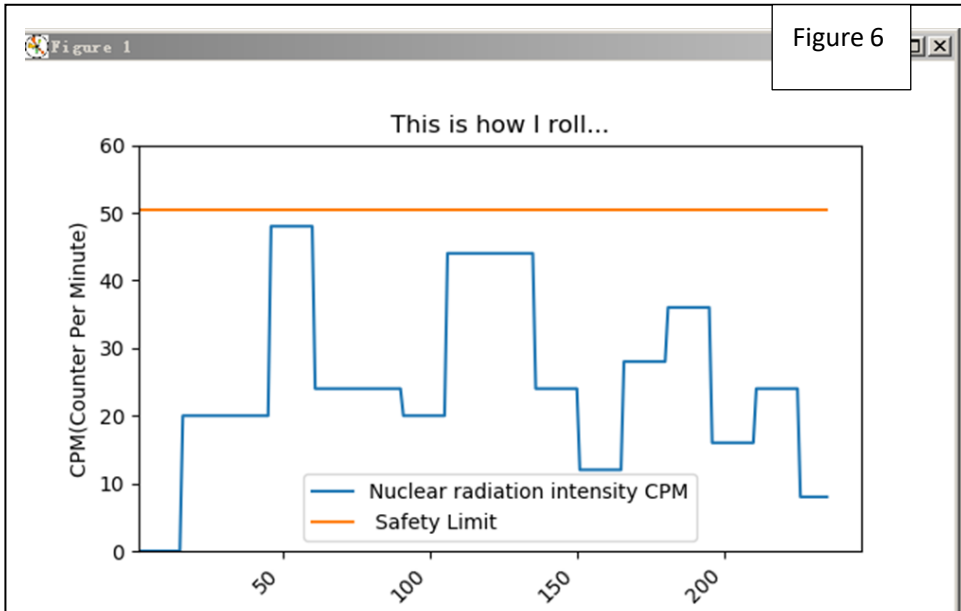
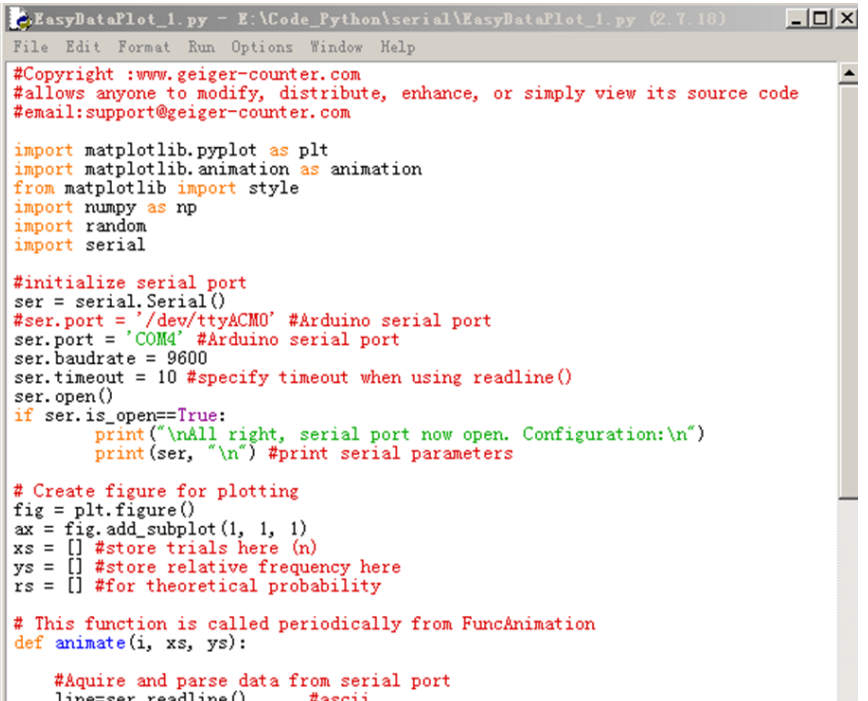


Figure 6





```
EasyDataPlot_1.py - E:\Code_Python\serial\EasyDataPlot_1.py (2.7.18)
File  Edit  Format  Run  Options  Window  Help
#Copyright :www.geiger-counter.com
#allows anyone to modify, distribute, enhance, or simply view its source code
#email:support@geiger-counter.com

import matplotlib.pyplot as plt
import matplotlib.animation as animation
from matplotlib import style
import numpy as np
import random
import serial

#initialize serial port
ser = serial.Serial()
#ser.port = '/dev/ttyACM0' #Arduino serial port
ser.port = 'COM4' #Arduino serial port
ser.baudrate = 9600
ser.timeout = 10 #specify timeout when using readline()
ser.open()
if ser.is_open==True:
        print("\nAll right, serial port now open. Configuration:\n")
        print(ser, "\n") #print serial parameters

# Create figure for plotting
fig = plt.figure()
ax = fig.add_subplot(1, 1, 1)
xs = [] #store trials here (n)
ys = [] #store relative frequency here
rs = [] #for theoretical probability

# This function is called periodically from FuncAnimation
def animate(i, xs, ys):

    #Aquire and parse data from serial port
    line=ser.readline()      #ascii
```

Arduino Code: Geiger_Counter_serial.ino
Python Code: GeigerCounterSerialDataPlot.py
Step 1:
Download Arduino Code: Geiger_Counter_serial.ino

In this application we are using lib LiquidCrystal_I2C and MsTimer2 , which could be installed through Arduino Lib.

Step 2:

Running GeigerCounterSerialDataPlot.py on upper computer, the libs which should be installed properly before the scripts could run

Such as: matplotlib , numpy , serial

Then you could see as figure 6 shows.