```
# CHS Avalanchers plot magnitude code
# Made by Raul Alvarado on March 27th, 2026

# Import the necessary libraries
import pynanovna, math, keyboard
import numpy as np
import matplotlib.pyplot as plt

# Initialize the VNA serial link
vna = pynanovna.VNA()

#Sets frequency range and number of points to sweep
vna.set_sweep(850, 900, 101)

# set the dB calibration to -∞
dBCalibrate = -math.inf

# create blank arrays to differentiate the complex64 int parts
real = []
imaginary = []

# initialize VNA data stream and add condition to start
# data analysis
stream = vna.stream()
hasCalibd = False

# enable interactive plot
plt.ion()

# for all collected parameters in the VNA stream
for s11, s21, frequencies in stream:

    # if the C key on the keyboard is pressed, init
    # calibration for the VNA and declare that hasCalibd
    if keyboard.is_pressed('c'):
        # extract real and imaginary number parts
        real = np.real(s21)
        imaginary = np.imag(s21)

        # this is the equivalent of |S21| = sqrt(Re{S21}^2 + Im{S21}^2)
        # unlike a Friis gain equation, this calculates magnitude instead
        # of expected loss.
        dBCalibrate = np.sqrt(np.add(np.square(real), np.square(imaginary)))
        hasCalibd = True

    # once a initial calibration has been taken,
```

```
# start plotting data asap
if hasCalibd:
    # extract real and imaginary number parts
    realAfterCalib = np.real(s21)
    imaginaryAfterCalib = np.imag(s21)

    # equivalent of |S21| = sqrt(Re{S21}^2 + Im{S21}^2)
    dBMeasure = np.sqrt(np.add(np.square(realAfterCalib), np.square(imaginaryAfterCalib)))
    dBDisplay = dBMeasure - dBCalibrate

    # this is a measure of the average amplitude (of magnitude)
    # over the dbDisplay time.
    RMSDisplay = np.sqrt(np.sum(np.square(dBDisplay)))

    # initializes plot
    plt.clf()
    plt.figure(figsize=(10, 6))

    # plots out the magnitude, raw frequency, and s21 relative to different frequencies.
    plt.plot(dBDisplay, 'r', label='|s21| magnitude')
    plt.plot(frequencies/850, 'b-', label='measured s21 (raw freq)')
    plt.plot(frequencies/850, dBDisplay, 'g-', label='measured s21 rel. to display')
    plt.plot(frequencies/850, dBCalibrate, 'y-', label='measured s21 rel. to calibrate')
    plt.plot(frequencies/850, dBMeasure, 'm-', label='measured s21 rel. to last measure')
    plt.xlabel("Frequency (MHz)")
    plt.ylabel("Magnitude (dB)")
    plt.title("Average magnitude (dB): " RMSDisplay)

    # draws the plot
    plt.draw()
    plt.pause(0.01)
```