**MMRI Python NI-DAQ Program**

**2023-12-14**

**MMRI**

**Ray Lyu**



Table of Contents

[1 Introduction 3](#_Toc153453359)

[2 Prerequisites 3](#_Toc153453360)

[2.1 Python NiDAQ MAX driver 3](#_Toc153453361)

[2.2 After downloaded and connect your sensor with USB open NI-MAX 3](#_Toc153453362)

[2.3 Find the location of exe 4](#_Toc153453363)

[2.4 Prerequisite for the script running 4](#_Toc153453364)

[3 Code overview 5](#_Toc153453365)

[3.1 Parameters 5](#_Toc153453366)

[3.2 Output of the exe 5](#_Toc153453367)

# Introduction

The script is the backbone of python National Instrument Sensor DAQ process. It’s main function is to build connections between the sensor and the python analysis program.

# Prerequisites

## Python NiDAQ MAX driver

<https://www.ni.com/en/support/downloads/drivers/download.ni-daq-mx.html#494676>

## After downloaded and connect your sensor with USB open NI-MAX

A screenshot of a computer

Description automatically generated

Find the name of your device, check it with the name defined on the script

try:

task.ai\_channels.add\_ai\_voltage\_chan("Dev1/ai0")

except:

task.ai\_channels.add\_ai\_voltage\_chan("cDAQ2Mod1/ai0")

The code in the orange box defined 2 type of DAQ Mode, if your device is not between the 2 please revise based on your model.

## Find the location of exe

Server PC location:

D:\OneDrive - McMaster Manufacturing Research Institute\Documents\GitHub\DAQ\dist\ni

A screenshot of a computer

Description automatically generated

Click ni.exe to start the program

The dt folder contains the data collected

A screenshot of a computer

Description automatically generated

testCsv contain the raw signal data

testSummary contain the statistical summary of the signal collected

## Prerequisite for the script running

The above exe package don’t require the python environment but it requires all the files contain in the program folder, if you want to run the python script for quick testing please install according python environment and library.

# Code overview

## Parameters

Directory

directory = os.getcwd() + "\\dt\\testCsv\\"

excel\_directory = os.getcwd() + "\\dt\\testSummary\\"

Current directory involved in the executable mainly specify the location of the data saved

Other parameters involved in the collecting process

sampling\_frequency = 5e3

mag\_threshold = 0.0012

time\_elapse = 4

duration = 25

The sampling frequency determine how many data point collected every second

Mag\_threshold determine the trigger level vibration magnitude, the program will start to collect data after the magnitude over the threshold

Time\_elapse define the time length segment of the data collected. For the data saved it will contain 4s of data based on parameters above

Duration define the amount of time it collected across, right now the program will totally run 25s, it doesn’t depend on when the program is triggered. (no matter the program triggered at 1s or 5s it will only run 25s)

## Output of the exe

The output for the program contains 2 part, one is the raw signal data, which contains the magnitude of the signal as well as the time stamp

The magnitude data saved on: datas list

The time stamp saved on: measureTime list

current\_time\_str = datetime.now().strftime('%Y-%m-%d-%H-%M-%S')

filename = f"{directory}{current\_time\_str}-trial{trial\_number}.csv"

df = pd.DataFrame({'time': measureTime, 'magnitude': datas}).to\_csv(filename)

The file name saved at moment contain the date and time information as well as the Trial number

rms\_value = calculate\_rms(wdt)

max\_value = max(wdt)

min\_value = min(wdt)

today\_date = datetime.now().strftime('%Y-%m-%d') # Get today's date

excel\_filename = excel\_directory + 'signal\_data.xlsx'

df = pd.read\_excel(excel\_filename)

new\_row = {'Trial': trial\_number, 'RMS': rms\_value, 'Max': max\_value, 'Min': min\_value, 'Date': today\_date}

df = df.append(new\_row, ignore\_index=True)

df.to\_excel(excel\_filename, index=False)

The time domain statistic are saved in “signal\_data.xlsx”, right now it will record the RMS, Max and Min value for the whole signal which saved in wdt list.